Understanding Adaptive Immune Response with Deep Learning

Olga Lyudovyk
Dept. of Computational Oncology
Memorial Sloan Kettering Cancer Center
New York, NY 10065
lyudovyo@mskcc.org

Artem Streltsov
SC Johnson College of Business
Cornell University
Ithaca, NY 14853
as3923@cornell.edu

Alvin Qu
Cornell Tech
Cornell University
New York, NY 10044
aq38@cornell.edu

Edmond Lu
Cornell Tech
Cornell University
New York, NY 10044
el535@cornell.edu

Shuo Han
Cornell Tech
Cornell University
New York, NY 10044
sh2439@cornell.edu

Abstract

Recognition of antigens by T-cells and B-cells is at the core of the adaptive immune system that protects against viruses, bacteria, and even cancer. Large experimental datasets of T-cell receptors (TCR) and the antigens they recognize have recently become available, but the principles or recognition motifs remain unknown. We apply Transformer and Graph Convolutional Network models to predict antigens that a given TCR will bind and to produce antigen recognition motifs. Our BERT-based solution significantly outperforms the best-in-class model for this task with a test set Average Precision and ROC AUC of 0.925 and 0.911 respectively. This work can have wide practical applications in cell therapy and vaccine development.

1 Introduction

Harnessing capabilities of the immune system is a new and promising paradigm for prevention and treatment of many diseases including cancer, AIDS, and COVID-19. However, elucidating components and mechanisms of the immune system is still an area of active research. In particular, how the immune system recognizes pathogens and distinguishes them from peptides derived from one's own cells remains poorly understood.

The human immune system consists of innate and adaptive immunity. Innate immunity is the nonspecific defense mechanism that is activated immediately or within hours of an antigen's (foreign invader's) appearance in the body. The adaptive immunity is more targeted and evolves in response to previous exposure to a foreign antigen. It emerges as T-cells and B-cells (key components of the immune system) encounter, recognize, and respond to a novel pathogen.

Vesicles within eukaryotic cells called endosomes continuously break down proteins produced by cells or microbes into shorter peptides: segments of 9-17 amino acids in length. Some of these peptides are then presented to the immune system cells via helper molecules called Major Histocompatibility Complex (MHC). Peptides that can bind to a person's specific MHC molecules are referred to as antigens. Cells with MHC molecules on their membrane then present these antigens to T-cells and B-cells, and portions of these antigens 5-6 amino acids in length that are recognized by T-cells or B-cells are referred to as epitopes. (MHC class molecules' ability to bind and present these antigens is out of scope of this project.)

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

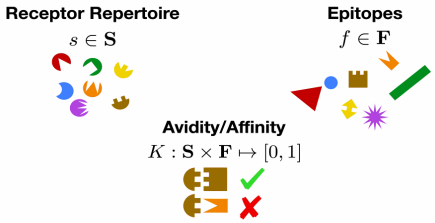


Figure 1: Schematic of matching T-cell receptors and epitopes (Borrowed from Sethna 2021).

T-cells and B-cells whose receptors are able to find a "target" epitope replicate and take up a more significant portion of a person's immune cell repertoire. Those T- and B-cells whose receptors happen to recognize epitopes from one's own cells are removed to avoid unwanted auto-immune reaction (Olson 2018). In this paper we focus on the ability of T-cells to recognize their cognate epitopes and attempt to learn and predict this T-cell to epitope pairing. Figure 1 illustrates T-cell receptor (TCR) and epitope pairings. This pairing is not one-to-one: this ability of the same TCR to recognize several epitopes and for the same epitope to be recognized by several TCRs is referred to as cross-reactivity.

2 Background

T-cells and B-cells evolve to be able to bind and recognize novel epitopes because the DNA sequence that encodes their T-cell Receptors (TCRs) and B-cell Receptors (BCRs) undergoes continuous mutations and re-combinations: a variety of randomly recombined TCR sequences are constantly produced via the process of V(D)J recombination. The key element produced by the V(D)J recombination is the hyper-variable region named complimentary determining region 3 (CDR3) which arises via nucleotide insertions, deletions, or point mutations and is directly responsible for specificity of epitope recognition. Flanking molecules produced by genes referred to as V-gene and J-gene might also affect a TCR's epitope specificity to an unknown degree. There is a limited set of V-genes and J-genes in the human population, and unlike the CDR3 region they do not undergo hyper-mutation.

High-throughput sequencing of TCRs elucidated the sequences and the generation process of these critical components of the immune system, but our understanding of how T-cells actually recognize their epitopes has remained elusive. Prior work includes clustering T-cells based on their sequence similarity to understand preferential expansion in response to an epitope (Huang 2020, Sant 2018) and generative models that estimate the generation and selection factors in each individual that govern the recombination and expansion processes (Olson 2018, Sethna 2020, Isacchini 2020, Sethna 2019).

Recently, two teams proposed deep learning approaches to understanding T-cell specificity and cross-reactivity. One approach (Bravi 2020) applies a Restricted Boltzmann Machines-based probabilistic model to learn features of TCRs that differentially expand in response to in vitro stimulation with a set of patient-specific antigens. This model can distinguish between specific and non-specific expansion and establish a connection between epitope specificity and TCR sequence motif. However, this model requires TCR sequencing data collected at two time points: pre- and post-stimulation with a set of antigens. In contrast, DeepTCR (Sidhom et al, 2019) builds a Convolutional Neural Network model to "featurize" TCR sequences in both an unsupervised manner (on a large set of unlabeled TCR sequences) and in a supervised manner (a small set of experimentally obtained paired TCRs and epitopes). Sidhom et al (2019) show that this CNN-derived featurization can be used in a number of downstreat TCR repertoire analysis tasks. More recently, Springer at al (2020) developed ERGO (pEptide tcR matchinG predictiOn), a Deep Learning model based on Natural Language Processing (NLP) to predict TCR-epitope binding. To the best of our knowledge, this is the state of the art (SOTA) general TCR-epitope prediction model. We replicated the architecture in our LSTM model as one of our baseline models, but trained it on a larger dataset as described in Datasets and Evaluation section.

The devastating global COVID-19 pandemic spurred a collective effort of the global scientific community to solve this challenge. Specifically, a collaboration between Adaptive Biotechnologies and Microsoft has produced an unprecedented dataset of high-throughput TCR sequencing data of COVID-19 patients and computational and experimental in-vitro identification of putative SARS-CoV-2-specific TCRs and the SARS-CoV-2 viral epitopes that they are able to recognize (Nolan

2020). This dataset and previous smaller experimental datasets pairing TCRs and epitopes from various infectious diseases collated as part of VDJdb (Shugay 2018) present an opportunity to learn patterns of TCR-epitope recognition using deep learning approaches. To our knowledge, this study has not been undertaken on this combined dataset to date.

3 Datasets and Evaluation

We use three publicly available datasets for this project: Adaptive Biotechnologies' ImmuneCODE database containing putative SARS-CoV-2-specific TCR sequences and their corresponding epitopes (Nolan 2020); VDJdb: a collection of paired TCR sequences and epitopes from previously published studies, largely in infectious diseases (Shugay 2018); and McPAS-TCR, a manually curated database of TCR sequences associated with various pathologies and antigens (Tickotsky 2017).

ImmuneCODE (Nolan 2020) contains 143,938 unique CDR3 regions from TCRs of over 1,400 subjects exposed to or infected with SARS-CoV-2 virus. In a series of multiplexed experiments based on MIRA assay technology, these TCRs were co-cultured with various segments of SARS-CoV-2 viral peptidome and were found to pair with 363 unique SARS-CoV-2 epitopes. This dataset is published in three distinct files, with antigens of different length used in the experimental setup, from the length of a putative epitope itself to a "minigene" sequence of 30-40 amino acids.

VDJdb (Shugay 2018) is a compilation of 51,257 sequences of TCR CDR3s and 227 antigens known to be recognized by these TCRs. The longest CDR3 sequence is 36 amino acids long, and the longest epitope sequence is 18 amino acids long. The dataset contains duplicate pairs due to redundancy in other data points, such as MHC class alleles. This dataset was deduplicated and processed by restricting TCRs to human TCRs only. McPAS (Tickotsky 2017) is similar to VDJdb but somewhat smaller, containing over 20,000 TCR sequences matching over 300 unique epitopes. There is some overlap between these two datasets.

TCR	V-gene	J-gene	Epitope	Antigen Species
CASSAQGTGDRGYTF	TCRBV27	TCRBJ01-02	ADAGFIKQY	SARS-CoV-2
CASSKGTVSGLSG	TCRBV21	TCRBJ02-07	LADAGFIKQY	SARS-CoV-2
CASSYLPGQGDHYSNQPQHF	TCRBV1301	TCRBJ1-501	FLKEKGGL	InfluenzaA

Table 1: Example of input data after pre-processing.

Examples of input data after pre-processing is shown in Table 1. The described datasets only contain positive (i.e. valid) TCR-antigen pairs. The experimental high-throughput setup of the ImmuneCODE dataset where combinations of all peptides and TCRs were tested in some sub-experiment allows us to conclude that a permutation of TCR and epitope sequences not listed in that dataset is an invalid match. This enabled us to use a data augmentation technique where we generated a large number of negative examples simply by permuting the TCR sequence in each entry, allowing us to train our models with a much larger dataset. While the same cannot be said with confidence regarding the other two datasets, Springer et al (2020) used the same technique on the VDJdb, McPAS, and another dataset which we did not include. We experimented restricting the training set permutations to only sequences from ImmuneCODE and including all sequences from the three datasets and did not find a meaningful difference in terms of performance or generalization. Generally, in the universe of 10^{15} to 10^{20} TCR sequences that can be created via the VDJ recombination (Laydon 2015), only up to 10^4 have been shown to bind at any particular peptide. So any randomly chosen permutation of TCR sequence and epitope is likely an invalid pair.

Instead of using the traditional approach to split the dataset into training and test set randomly, we recognized the need to evaluate how well our models generalize not only to unseen pairs, but also to unseen TCRs and unseen epitopes. We therefore withheld a random selection of TCRs and epitopes as a test set, taking care that it represents all three datasets and 4 different scenarios: unknown pair of known TCR/epitope; unknown TCR with known epitope; known TCR with unknown epitope; and unknown TCR paired with an unknown epitope. The resulting test set included 2% of TCRs and their pairs withheld from each dataset; 2% of epitopes and their pairs; and 5% of pairs of TCRs and epitopes which themselves were not withheld. The resulting test set included 19.8% of all available data, leaving in the training set before data augmentation 507548 unique valid pairs, 150422 CDR3s, and 1394 epitopes. We then randomly split the training set into training and validation and use the

validation set to evaluate the classification accuracy during training and to help us choose the best hyperparameters for each model.

We used Area under the Receiver-Operator Curve (ROC) and Average Precision (Precision-Recall Curve) as the main metrics to evaluate performance of each model on the validation and test sets. Both summarize the model performance under different classification threshold values and from different perspectives. The advantage of the Precision-Recall curve is that it focuses on the trade-off between precision and recall, and in our problem set up with a vastly larger set of possible negative examples, we particularly wanted to understand the model's precision at different classification thresholds.

To the best of our knowledge, the ERGO model (Springer 2020) is the best in class for this problem. However, the original model was trained on smaller datasets. Our LSTM implementation closely resembles the ERGO model and serves here as the comparison with the best in class benchmark.

4 Methods

Our proposed approach to understanding patterns of binding between TCRs and epitopes is to borrow attention-based models from the Natural Language Processing domain. TCR sequences and epitope sequences are combined together and a transformer is used to encode the resulting sequence. The output representation from the transformer is then used to classify the sequence as a valid or an invalid pair. We also consider alternative implementations based on the LSTM architecture with a separate LSTM as an encoder for the TCR and epitope sequences. The encoded outputs are further concatenated and classified as before. As another approach, patterns or motifs of recognition can be learned directly by implementing a graph convolutional neural network. Given a TCR and epitope sequences, the desired outputs of the model is an estimate of the probability that this TCR will recognize this epitope or the TCR-Epitope pair is valid.

In total, we implemented a baseline Multilayer Perceptron (MLP) model, an LSTM-based model, and 2 BERT Transformer models and evaluated them using metrics described in section 3, in order to find the best-performing approach. The detailed model architecture and experimental setup is shown in Table 2. We also attempted to apply bipartite graph neural network, inspired by an analogy with a recommender system. We describe our efforts and faced obstacles with this method in section 6.

Model	Architecture	Hyperparam. values	Tokenizer	N epochs
Baseline MLP	Embedding layer FC layer with ReLU FC layer with Sigmoid Binary Cross-Entropy (BCE) loss	learning rate=5e-05 embedding dims=64 hidden dims=512 weight decay=1e-05 optimizer=Adam dropout rate=0.1	3-gram tokenizer, separate for TCR and PEP	100
LSTM	Embedding layer TCR LSTM stack 2 PEP LSTM stack 2 Dropout FC layer with Sigmoid BCE loss	learning rate=1e-4 embeddings dims=10 hidden dims=500 weight decay=1e-05 optimizer=Adam	char-level, separate for TCR and PEP	100
BERT-base	BERT-base encoder FC layer with Sigmoid BCE loss	learning rate=1e-4 number of heads=12 hidden dims=768 weight decay=1e-05 optimizer=Adam	char-level on sequences of form '[CLS] T C R [SEP] P E P [SEP]'	100
BERT-mini	BERT-mini encoder FC layer with Sigmoid BCE loss	learning rate=1e-4 number of heads=4 hidden dims=768 weight decay=1e-05 optimizer=Adam	char-level on sequences of form '[CLS] T C R [SEP] P E P [SEP]'	100

Table 2: Model Architecture and Experimentation Setup

4.1 Baseline: N-gram MLP model

As a baseline, we created a simple 2-layer N-gram MLP model. The N-gram tokenizer splits both TCR and epitope sequences into n-grams of size $n = 3$. The n-grams seen in the training set are converted into a vocabulary, where the vocabulary of TCR sequence n-grams is treated separately from vocabulary of epitope sequence n-grams. Using these vocabularies, each TCR and epitope sequence is featurized as a vector of n-gram vocabulary values, and the two are then concatenated with a separator sequence in-between. The sequence is left-padded or limited to 40 elements, since 99% of CDR3 sequences and 98% of epitope sequences in our combined dataset are under 20 amino acids in length. A "fake" pair set is created by randomly pairing an epitope with a CDR3 sequence and removing those that actually exist in the combined dataset. The MLP model is then trained on the featurized true and fake paired sequences to predict the probability that the CDR3-epitope sequence is true.

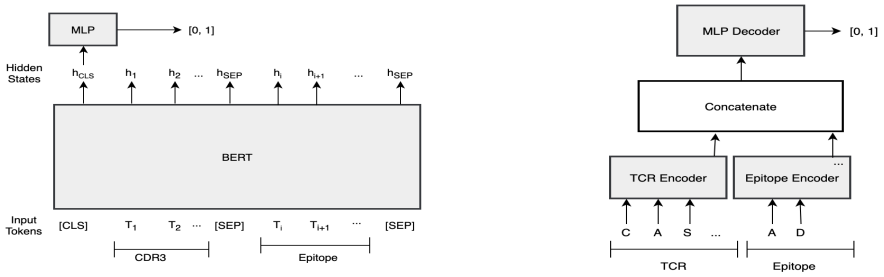
4.2 BERT

Our BERT model consists of two major parts: a BERT encoder and a Multilayer Perceptron (MLP). We explore two different BERT encoders: BERT-base and BERT-mini. The model architecture is shown in Figure 2(a). For each CDR3-epitope pair we prepend a [CLS] token and concatenate the two sequences together adding a [SEP] token in-between and at the end. The sequences are padded on the right to maximum length within each batch.

BERT Encoder: The BERT encoder takes in the tokenized sequence $\langle t_0, t_1, t_2, \ldots t_T \rangle$ (T denotes the index of the last token in the sequence), and generates the hidden representation from the last layer for each token of the input sequence : $f_E : t_i \rightarrow h_i, \forall i \in \{0, 1, 2 \ldots T\}$. We then take the hidden representation corresponding to the [CLS] token and pass it to the MLP classifier. We consider two encoders - a BERT-base (12 attention heads and 768 hidden size) and a BERT-mini (4 attention heads and 768 hidden size).

MLP Classifier: An MLP classifier with just the classification layer is built on top of the BERT to produce the final prediction. The MLP takes in the hidden state from the [CLS] token h_{CLS} and makes a prediction $f_D : h_{CLS} \rightarrow [0, 1]$. By default, the hidden layer has a dimension of 768. A weighted binary cross-entropy loss is then computed on the predictions.

Transformers generally require a decent amount of data to generalize. To this end, we pretrain the BERT encoders on the combined COVID, McPAS, VDJdb dataset in a self-supervised way on the Masked Language Modeling (MLM) task. We follow the MLM approach described in Devlin et al (2018) and for each input sequence mask each token with probability 15%, but then each masked token has a 10% probability of being replaced with a random token instead and another 10% probability of being replaced with the original token (while still being considered 'masked'). We then take the hidden representations from the last layer of the BERT encoder, pass them through a single fully-connected layer predicting the token that is masked with the cross-entropy loss ignoring any non-masked predictions.



(a) BERT Architecture

(b) LSTM Architecture

Figure 2: BERT and LSTM Model Architecture

4.3 LSTM

The LSTM model consists of three main parts: a TCR sequence LSTM encoder, an Epitope sequence LSTM encoder, and an MLP Decoder. The model architecture is shown in Figure 2(b). Our TCR sequence encoder $f_{TCR} : t \rightarrow Z_t$ first takes in a TCR sequence $t = \langle t_1, ..., t_l \rangle$ of character length l and outputs latent character embeddings $Z_{t_1}, ..., Z_{t_l}$ for each of l characters. Our Epitope sequence encoder $f_{EP} : p \rightarrow Z_p$ takes an input of Epitope sequence p of size n and creates the latent character embeddings $Z_{p_1}, ..., Z_{p_n}$ for each sequence character. The decoder $f_D : Z_D \rightarrow [0, 1]$ then takes in the concatenated TCR and Epitope latent character embeddings corresponding to the last token each $Z_D = [Z_{t_l}, Z_{e_n}]$ and outputs a probability that the TCR-Epitope pair is valid. Note that this model replicates the architecture of the ERGO model (Springer et al, 2020) as a point of comparison. More specifically, for this model we use an identical architecture, tokenizer and the set of hyperparameters as in the ERGO paper.

TCR Sequence Encoder: The TCR encoder takes in the tokenized input sequence and passes it through an embedding layer of dimension 10. The embedded TCR sequence then is fed into an LSTM encoder with a hidden size of 500 which generates hidden representations of each input character $\langle Z_{t_1}, ..., Z_{t_l} \rangle$. We further extract the representation of the last non-padded character of the TCR sequence and pass to the concatenation operator.

Epitope Sequence Encoder: Similarly to the TCR encoder, the Epitope encoder first creates embeddings for each character in the sequence through an embedding layer of size 10. These embeddings are then fed into a 500-unit LSTM layer, which outputs the latent embeddings $\langle Z_{p_1}, ..., Z_{p_n} \rangle$ and once again we keep the representation of the last valid character.

Decoder: The Decoder is a Multilayer Perceptron with one hidden layer of size 1000. Dropout and LeakyReLu are applied on top of the hidden layer and the output is passed to the final sigmoid classifier. As before, the output of this model is a probability that the TCR-Epitope pair is valid.

4.4 Bipartite Graph Neural Network

The Bipartite Graph Neural Network (BGNN) is formulated as a recommendation system problem. Specifically we attempt to use the recommendation system BGNN model architecture modeled in [15] and apply it in the context of CDRs and Epitopes. Each CDR is represented as a "user", each Epitope is represented as an "item", and the prediction of whether the CDR-Epitope pair exists is represented as a "rating." Given an input of CDR and Epitope pairs formatted in a bipartite graph structure B, our GNN $f_{GNN} : B \rightarrow M_B$ predicts the rating matrix M_B representing the probability of an edge occurring between all possible CDR-Epitope pairs. If we predict an edge occurs between a CDR-Epitope pair, it indicates that our BGNN predicts that said pair is valid. Our GNN consists of two major parts: a graph autoencoder and a bilinear decoder. The training is performed using negative log likelihood on the predicted ratings matrix M_B.

Graph Autoencoder: The Graph Autoencoder takes in the CDR-Epitope bipartite graph's feature matrix X and adjacency matrix A. The feature matrix is CDR learned embeddings, which we take from our BERT model's learned embeddings. The adjacency matrix is a matrix that represents whether a CDR-Epitope pair edge exists in the bipartite graph, having a value of 1 if the pair exists and 0 if it does not. The autoencoder passes these matrices into a graph convolutional encoder to perform a form of message passing. This is the same message passing as the one typically done in regular Graph Neural Networks where vector messages containing information about the state of the graph are passed and transformed across the graph's edges. The encoder returns separate CDR embeddings U and Epitope embeddings V.

Bilinear Decoder: The decoder is a bilinear function $f_{Bi} : U, V \rightarrow M$ and softmax function that acts on the Graph Autoencoder's embeddings U and V and returns a reconstruction rating matrix M, where each value at entry (i, j) represents the probability distribution between whether the edge is predicted to exist between CDR i and Epitope j.

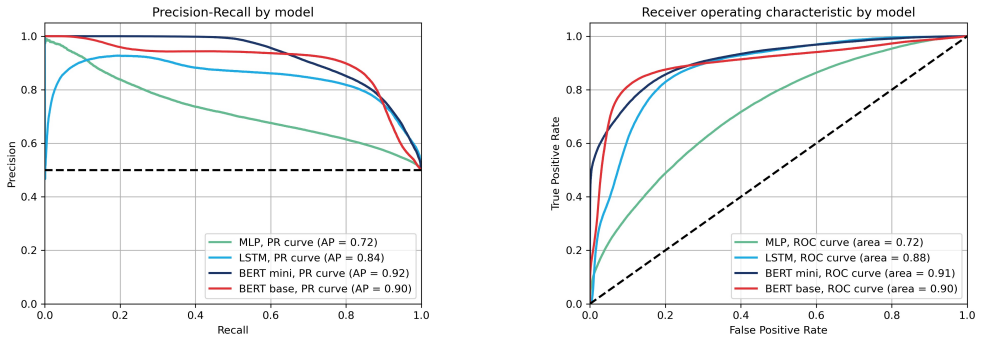
5 Results

The final accuracy, average precision score, and ROC AUC score for the validation and test sets by model can be found in Table 3. Out of the models we created, BERT-mini performed the best overall

with the highest Average Precision score of 0.925 and highest ROC AUC score of 0.911. Both the LSTM and BERT-base models performed fairly well on this task as well. All three NLP-inspired models significantly outperformed the baseline MLP model on all measures, and both BERT-based models beat the current SOTA (our LSTM reimplementation of the ERGO model) on all measures. Although our BERT-mini model works best overall, we found that BERT-base performed the best when it came to accuracy with a value of 0.854.

	Model	Accuracy	ROC AUC	Average Precision
Validation Set	Baseline MLP	0.669	0.773	0.641
	LSTM	0.918	0.979	0.930
	BERT-base	0.963	0.993	0.971
	BERT-mini	0.933	0.987	0.955
Test Set	Baseline MLP	0.651	0.718	0.721
	LSTM	0.802	0.876	0.844
	BERT-base	0.854	0.897	0.904
	BERT-mini	0.804	0.911	0.925

Table 3: Model Results



(a) Precision-Recall Curves (b) ROC Curves

Figure 3: Precision-Recall and ROC curves for the BERT and LSTM

We also evaluated the models specifically on how well they generalize in four scenarios: unseen pairs of seen TCR/epitope sequences, unseen TCRs, unseen epitopes, and unseen pairs where both the TCR and the epitope are withheld from the training data, with precision overall being a more important metric. Both precision and recall were high for unseen TCRs, meaning that the model learned TCR-specific features well that can generalize to unseen TCR sequences. This is perhaps unsurprising given the fact that the training set had a much larger variety of TCRs than epitopes. The model had a higher precision on unseen pairs of seen sequences, but surprisingly a much higher recall on unseen pairs where both sequences were not in the training set.

We also evaluated qualitatively the output of the model on 4 epitopes from the COVID-19 virus. We selected at random 2 epitopes that were included in the training set (klwaqcvql, ylqprtfl) and 2 that

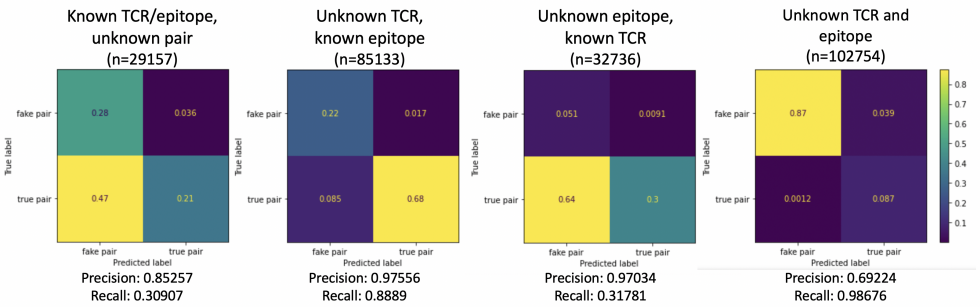


Figure 4: Confusion matrices for evaluating performance on four scenarios of test sequences.

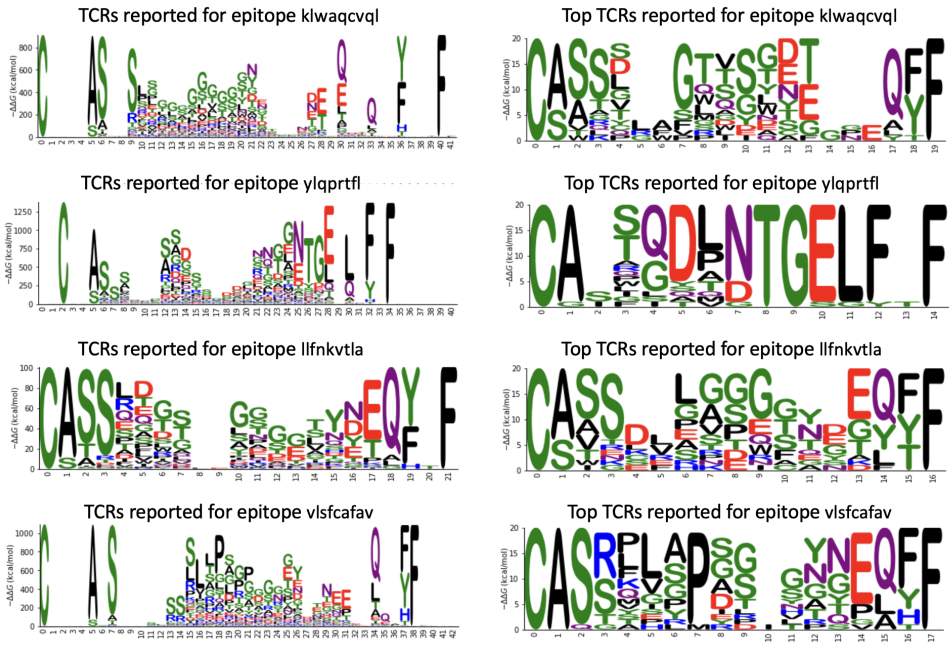


Figure 5: Sequence motifs of TCRs associated with 4 COVID epitopes: shown for all pairs reported in the combined dataset (left) and learned from top predicted TCRs by BERT-Mini model (right).

were excluded (llfnkvtla, vlsfcafav). We aligned all the sequences reported in the combined dataset using BioPython and muscle package (Chapman 2000, Edgar 2004), then calculated the position weight matrices (PWMs) for each consensus sequence and plotted the resulting sequence logos using Logomaker (Tareen, 2020), seen in Figure 6, left. We then selected a subset of 10,000 TCRs at random from the combined dataset and predicted the score of those TCRs paired with these 4 epitopes using BERT-mini. This score can be interpreted as the confidence score that a given TCR recognizes a given epitope. We selected the top 20 TCRs for each epitope, aligned them and generated the sequence PWM and logo (Figure 6, right). Motifs seen in a sequence logo of all known sequences were recovered from the sequences predicted by the model, including "TGELFF" for ylqprtfl and "NDEQF" for llfnkvtla. This was true even for cases where the model did not train on any sequence pairs for that specific epitope.

6 Discussion

In conclusion, BERT based Transformer models consistently performed better than the previous SOTA on predicting the binding of a TCR-Epitope pair. Specifically, a number of important observations can be made from the Receiver Operating Curves (ROC) and the Precision-Recall (PR) curves presented in Figure 3. First, the ROC and PR curves of the BERT-mini model strictly contain those corresponding to the baseline MLP and LSTM models which means that BERT-mini is strictly superior to these baselines. Secondly, BERT-base achieves the best combination of precision and recall (for instance, 0.8 recall with 0.9 precision) as well as true positive and false positive rates (for example, 0.85 true positive rate at a false positive rate of only 0.1) that are infeasible with any other model. Thus, the ultimate decision on the winner between BERT-mini and BERT-base depends on the choice of the trade-off tolerance which is application-specific.

From our experiments, we found that applying our dataset to the proposed BGNN architecture does not work well. We tried modifying our dataset to fit as was proposed in [15], but were unable to produce any meaningful results here. This is mainly because of both the large size of our dataset as well as the difference between the graph structure between our data and the data the BGNN architecture was designed for. We feel that the proposed GNN has promise for predicting whether a CDR binds with and Epitope, but could not be fully explored in this project.

Two important practical applications for our models are identification of molecular (TCR) biomarkers of viral disease or immune response, such as response to vaccination, and identification of tumor-specific TCRs for potential CAR-T cell therapy. CAR-T cell therapy is a procedure where T-cells are recovered from a patient, modified to be more effective at targeting the tumor, expanded in vitro, and reintroduced into the patient. In the first scenario of the infectious disease response, likely epitopes are predicted from the viral sequence; blood samples are collected and TCR repertoires are sequenced for a population that is known to be exposed to the condition (infectious disease or vaccine) and compared to TCR repertoires from a naive population. Unfortunately simply comparing TCR sequences found in one population and not in another suffers from a vast number of false positives, and in vitro experiments to confirm which of those TCRs actually bind to the epitopes of interest are costly and often impractical. Predictions by a model such as the ones we propose can help narrow down a list of potentially millions of TCR sequences to a set of likely candidates to be further investigated experimentally. In the second scenario, neoantigens are predicted from the mutations that arose in a patient's specific tumor. Blood samples are taken from this patient and TCR repertoires are sequenced, often on the order of several million TCR sequences. No negative population to compare these TCRs exists, and in vitro validation of candidate TCRs against the tumor neoantigens may not be practical due to time constrains. Our models can help with precise and timely identification of candidate TCRs that may recognise the tumor neoantigens in order to ensure that the right T-cells are weaponized and expanded for CAR-T cell therapy.

7 Conclusion and Future Work

We consider the case of using Transformer models when trying to predict whether a T-cell receptor binds to an epitope. We show that our BERT models work effectively on the task, generalizing well to previously unseen sequences and outperforming the current SOTA. Our LSTM model replicates the ERGO architecture (Springer 2020) and represents the current SOTA, trained a much larger training dataset to train the LSTM model than was originally done by Springer at al. (2020). Nevertheless, our BERT models consistently outperform the SOTA architecture on all evaluation metrics.

In future work, it would be interesting to leverage the BGNN structure proposed in the model. We found this method promising, but could not fully implement the model given the project's time constraints and challenges in getting a BGNN to work on a real life-sized dataset. For our Transformers, it would also be interesting to explore a more compact and effective Transformer model, DistilBERT, as our Transformer model produces high dimensional embeddings and are computationally time and memory consuming. Another interesting future direction is to apply transfer learning: to pre-train the BERT model in an unsupervised manner on known possible TCR sequences (on the order of over tens of billions of TCR sequences exist in various combined datasets) and on all possible viral and bacterial epitopes separately, then finetune it on TCR-epitope pairs used in this analysis.

8 About the Team

Olga Lyudovyk: PhD student in Computational Biology and Medicine at Weill Cornell / Tri-I program. Olga's research focuses on computational immunology and immuno-oncology.

Artem Streltsov: PhD student in Applied Economics and Management at SC Johnson College of Business. His interests include applications of machine learning in economics and finance.

Alvin Qu: Masters student in Computer Science at Cornell Tech. His interests include software development with a focus in machine learning components that utilize NLP and CV.

Edmond Lu: Masters student in Computer Science at Cornell Tech. His interests include NLP and recommendation systems.

Shuo Han: Masters student in Computer Science at Cornell Tech. His interests include data science and building machine learning models.

References

[1] Bravi B, Balachandran VP, Greenbaum BD, Walczak AM, Mora T, Monasson R, Cocco S. Probing T-cell response by sequence-based probabilistic modeling. BioRxiv, 2020.12.17.423283. https://doi.org/10.1101/2020.12.17.423283

[2] Devlin J, Chang M, Lee K, Toutanova K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. 2018

[3] Huang H, Wang C, Rubelt F, Scriba TJ, Davis MM. Analyzing the Mycobacterium tuberculosis immune response by T-cell receptor clustering with GLIPH2 and genome-wide antigen screening. Nat Biotechnol. 2020 Oct;38(10):1194-1202. doi: 10.1038/s41587-020-0505-4. Epub 2020 Apr 27. PMID: 32341563; PMCID: PMC7541396.

[4] Isacchini G, Sethna Z, Elhanati Y, Nourmohammad A, Walczak AM, Mora T. Generative models of T-cell receptor sequences. Phys Rev E. 2020 Jun;101(6-1):062414. doi: 10.1103/PhysRevE.101.062414. PMID: 32688532.

[5] Nolan S, Vignali M, Klinger M, Dines JN, Kaplan IM, Svejnoha E, Craft T, Boland K, Pesesky M, Gittelman RM, Snyder TM, Gooley CJ, Semprini S, Cerchione C, Mazza M, Delmonte OM, Dobbs K, Carreño-Tarragona G, Barrio S, Sambri V, Martinelli G, Goldman JD, Heath JR, Notarangelo LD, Carlson JM, Martinez-Lopez J, Robins HS. A large-scale database of T-cell receptor beta (TCR) sequences and binding associations from natural and synthetic exposure to SARS-CoV-2. Res Sq [Preprint]. 2020 Aug 4:rs.3.rs-51964. doi: 10.21203/rs.3.rs-51964/v1. PMID: 32793896; PMCID: PMC7418738.

[6] Olson BJ, Matsen FA 4th. The Bayesian optimist's guide to adaptive immune receptor repertoire analysis. Immunol Rev. 2018 Jul;284(1):148-166. doi: 10.1111/imr.12664. PMID: 29944760; PMCID: PMC6347465.

[7] Raffel C, Shazeer N, Roberts A, Lee K, Narang S, Matena M, Zhou Y, Li W, Liu P. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. Journal of Machine Learning Research 21 (2020).

[8] Sant S, Grzelak L, Wang Z, Pizzolla A, Koutsakos M, Crowe J, Loudovaris T, Mannering SI, Westall GP, Wakim LM, Rossjohn J, Gras S, Richards M, Xu J, Thomas PG, Loh L, Nguyen THO, Kedzierska K. Single-Cell Approach to Influenza-Specific CD8+ T Cell Receptor Repertoires Across Different Age Groups, Tissues, and Following Influenza Virus Infection. Front Immunol. 2018 Jun 27;9:1453. doi: 10.3389/fimmu.2018.01453. PMID: 29997621; PMCID: PMC6030351.

[9] Sethna Z. Modeling epitope cross reactivity: Greenbaum Lab Presentation, 03/19/2021.

[10] Sethna Z, Elhanati Y, Callan CG, Walczak AM, Mora T. OLGA: fast computation of generation probabilities of B- and T-cell receptor amino acid sequences and motifs. Bioinformatics. 2019 Sep 1;35(17):2974-2981. doi: 10.1093/bioinformatics/btz035. PMID: 30657870; PMCID: PMC6735909.

[11] Sethna Z, Isacchini G, Dupic T, Mora T, Walczak AM, Elhanati Y. Population variability in the generation and selection of T-cell repertoires. PLoS Comput Biol. 2020 Dec 9;16(12):e1008394. doi: 10.1371/journal.pcbi.1008394. PMID: 33296360; PMCID: PMC7725366.

[12] Shugay M, Bagaev DV, Zvyagin IV, Vroomans RM, Crawford JC, Dolton G, Komech EA, Sycheva AL, Koneva AE, Egorov ES, Eliseev AV, Van Dyk E, Dash P, Attaf M, Rius C, Ladell K, McLaren JE, Matthews KK, Clemens EB, Douek DC, Luciani F, van Baarle D, Kedzierska K, Kesmir C, Thomas PG, Price DA, Sewell AK, Chudakov DM. VDJdb: a curated database of T-cell receptor sequences with known antigen specificity. Nucleic Acids Res. 2018 Jan 4;46(D1):D419-D427. doi: 10.1093/nar/gkx760. PMID: 28977646; PMCID: PMC5753233.

[13] Sidhom J-W, Larman HB, Ross-MacDonald P, Wind-Rotolo M, Pardoll DM, Baras AS. DeepTCR: a deep learning framework for understanding T-cell receptor sequence signatures within complex T-cell repertoires. BioRxiv, 464107, 2019. https://doi.org/10.1101/464107

[14] Ying R, Lou Z, You J, Wen C, Canedo A, Leskovec J. Neural Subgraph Matching. arXiv:2007.03092 [cs.LG]. 27 Oct 2020.

[15] Rianne van den Berg, Thomas N. Kipf, Max Welling: Graph Convolutional Matrix Completion. arXiv:1706.02263

[16] Ido Springer, Hanan Besser, Nili Tickotsky-Moskovitz, Shirit Dvorkin, Yoram Louzoun: Prediction of specific TCR-peptide binding from large dictionaries of TCR-peptide pairs. BioRxiv, 2020. https://doi.org/10.1101/650861

[17] Ammar Tareen, Justin B Kinney, Logomaker: beautiful sequence logos in Python, Bioinformatics, Volume 36, Issue 7, 1 April 2020, Pages 2272–2274, https://doi.org/10.1093/bioinformatics/btz921

[18] Edgar, R.C. (2004) MUSCLE: multiple sequence alignment with high accuracy and high throughput. Nucleic Acids Res. 32(5):1792-1797

[19] Chapman BA and Chang JT (2000). Biopython: Python tools for computational biology. ACM SIGBIO Newsletter, 20, 15-19

[20] Tickotsky N, Sagiv T, Prilusky J, Shifrut E, Friedman N. McPAS-TCR: a manually curated catalogue of pathology-associated T cell receptor sequences. Bioinformatics. 2017 Sep 15;33(18):2924-2929. doi: 10.1093/bioinformatics/btx286. PMID: 28481982.