Zürcher Hochschule
für Angewandte Wissenschaften

School of Engineering

# Project thesis (Data Science)

# Deep Learning for T-cell epitope binding affinity prediction

**Authors** Yves Bähler
Gérôme Meyer

**Supervisor** Dr. Jasmina Bogojeska

**Datum** 22.12.2023

# Contents

# Abstract

This project thesis explores TCR to epitope binding affinity prediction with Deep Neural Networks and evaluates the use of different layers of Large Language Models (LLMs) pre-trained on proteins as embeddings for this task. Using Meta's Evolutionary Scale Model 2 (ESM2), embeddings for a data set of TCR and epitope amino acid sequences were generated. Then, different Neural Network architectures were trained and evaluated on their ability to generalize out to unseen data in three different scenarios of data availability. We were able to confirm previous work which showed that using different internal layers of embedding models can have significant impact on model performance, suggesting that different structures are learned per layer by these models. We have also found indications that deep learning architectures may be able to solve the more difficult task of predicting completely unknown TCR-Epitope binding pairs.

Zurich University
of Applied Sciences

School of
Engineering

# DECLARATION OF ORIGINALITY

## Project Work at the School of Engineering

By submitting this project work, the undersigned student confirm that this work is his/her own work and was written without the help of a third party. (Group works: the performance of the other group members are not considered as third party).

The student declares that all sources in the text (including Internet pages) and appendices have been correctly disclosed. This means that there has been no plagiarism, i.e. no sections of the project work have been partially or wholly taken from other texts and represented as the student's own work or included without being correctly referenced.

Any misconduct will be dealt with according to paragraphs 39 and 40 of the General Academic Regulations for Bachelor's and Master's Degree courses at the Zurich University of Applied Sciences (Rahmenprüfungsordnung ZHAW (RPO)) and subject to the provisions for disciplinary action stipulated in the University regulations.

City, Date:

Winterthur, 22.12.2023

Signature:

Gerome Meyer

Yves Bähler

The original signed and dated document (no copies) must be included after the title sheet in the ZHAW version of all project works submitted.

Zurich University
of Applied Sciences

# 1 Introduction

T-Cells are a core part of the human immune system's defense against all kinds of diseases. Together with other lymphocytes they serve both as a first line of defence as well as an integral part of the adaptive immune response. The T-Cell receptor (TCR) plays a key role in this, determining the epitopes of foreign cells a T-Cell can bind to and therefore attack. A large variety of TCRs is necessary in order to defend against the similarly large variety of different pathogens that we may encounter throughout our lifetime. In TCRs this variety is achieved through stochastic genetic recombination which theoretically allows for up to $10^{15}$ - $10^{20}$ different receptors [1]. The amount of different possiblities makes it very difficult to reliably predict the binding capabilities of specific TCRs. At the same time, if there was a reliable way to predict which TCRs would bind to a specific epitope much more personalized treatments would be made possible as well as potentially being able to read a patients immunologic history. Knowing this history could amongst other things empower the application of therapeutic T-cells for cancer treatment [2]. Further, because of the large pool of possible bindings, clinical studies in this field are expensive and time-intensive. Even a tool which is not perfectly reliable could empower scientist to conduct these studies more efficiently by only looking at potential candidate cells.

Due to the large amount of possibilities between TCRs and Epitopes, and the intricate workings of their bindings not being fully explored, our approach is to combine state-of-the-art machine learning methods with available data to discover if predicting TCR-Epitope bindings is reliably possible with Deep Learning techniques.

Also, this project thesis explores how embeddings generated by language models specifically pre-trained on protein sequences can be used for the task of TCR-Epitope binding prediction, as well as different model architectures for the classification task.

## 1.1 Related Work

This chapter will provide an overview of existing literature and previous work around TCR-Epitope prediction.

### 1.1.1 T-cell receptor specificity prediction with bimodal attention networks (TITAN)

Serving as main inspiration for our work, the TITAN model by Anna Weber, Jannis Born and Maria Martinez leverages transfer learning through the use of the SMILES

embedding for epitopes and uses a Bimodal Attention Network to improve prediction performance [3]. They established a baseline with a $k$-nearest-neighbor (K-NN) classifier using Levenshtein distance, where they predicted binding by choosing a fixed distance threshold [4, 2] and achieved an AUC-ROC of 0.78. They then compared this baseline against their newly proposed architecture, which separately encodes the SMILES annotation of the epitope and BLOSUM62 matrix of the TCR sequence with a set of convolutions. The encoding is then used in a set of context attention layers before being used for classification in linear feed forward networks. Using this architecture, they achieved a score of 0.87 ROC-AUC [2]. You can find more detailed explanations of the Levenshtein distance and the ROC-AUC score in Section 3.3.

An important difference from the TITAN paper to ours, is that in it the CDR3 region of the TCR sequence was used for training some of the models whereas we decided to use the full TCR sequence for all of our training data, since they showed that using the full sequence boosts model performance. Also noteably, they embedded epitopes with the SMILES representation and TCRs with the BLOSUM62 matrix, whereas we chose to embed both with Meta's ESM2 model in order to evaluate its performance for the task of TCR-Epitope binding prediction [2].

### 1.1.2 Evolutionary-scale prediction of atomic-level protein structure with a language model (ESM2)

In this Paper, Zeming Lin et al. leverage the capabilities of large language models (LLMs) to learn representations of complex structures when training on a simple task at a large scale of data and compute. Their models specifically encode evolutionary information to predict the protein structure at a resolution of the atoms [5]. They demonstrate direct inference of full atomic level structure being extracted using a large language model. Their pre-trained models are available online[1] and can easily be used for inference through their Python Package.

### 1.1.3 VDJdb

In the database VDJdb, a curated database of T-Cell receptors is provided for use. The mission of the VDJdb is to aggregate the scarce TCR specificity information and provide a curated repository to store the data. The data set we used by the TITAN paper partially contains data from this database [6].

[1] You can find the ESM models here: https://github.com/facebookresearch/esm

### 1.1.4 CEDAR - Cancer Epitope Database and Analysis Resource

The CEDAR database holds experimental data on antibody and T-Cell epitopes from humans and animals in regarding the context of cancer [7]. The Website under cedar.iedb.org also hosts tools to assist and predict, aswell as analysing cancer epitopes. This database could be used to expand on the data used in TITAN and increase the data diversity.

## 1.2 Objectives

This section describes the initial objectives of this project thesis. The exact project description (written in German) can be found in the appendix (see Chapter 6.1).

### 1.2.1 Analyse data and discover problems

We have to analyse the data to discover and address problems in the available data. With our approach, we aim to discover where data diversity is missing and if it is possible to remedy it. Furthermore, we provide an extensive oversight over the data used for our experiments. We also aim to analyse how the TCRs and Epitopes differ in between each dataset.

### 1.2.2 Evaluating ESM2 Emebddings for downstream task

This project thesis specifically evaluates the capabilities of Meta's ESM2 model for the task of TCR-Epitope binding prediction. Using the data originally used by the TITAN paper, we create an Embedding using ESM2, specifically the 650 Million parameter and the 3 Billion parameter variants [5, 2].

### 1.2.3 Establishing a simple MLP baseline

We aim to establish a baseline Model using the ESM Embedding, using a multi layer perceptron (MLP) classifier with 512 neurons and the ReLU and Sigmoid activation functions [8]. This allows us to compare if model complexity is able to improve the prediction for the TCR-Epitope binding prediction problem.

### 1.2.4 Evaluating deep learning model architectures

We aim to discover if a increase in Model complexity helps solve the TCR-Epitope binding prediction problem and aim to find a model that can generalize to unknown TCR-Epitope bindings without any prior knowledge about their other bindings and their specific interaction. Using the MLP baseline as a starting point, we evaluate

the change in prediction performance through use of Deep Neural Networks as well as Transformers.

## 1.3 Problem Definition

In current literature, the Epitope part of the TCR-Epitope binding prediction problem is not given a lot of weight due to lack of available data. We wish to discover through various experiments if the Epitope data sparsity can be addressed or circumvented through use of deep learning methods such as Deep Neural Networks and Transformers. Due to the nature of the data, normal data augmentation approaches don't reliably work - the new generated data points would never be observed in nature.

The Epitopes and TCRs are encoded in the genetic code format. This allows us to use NLP approaches. We aim to see if recent advancements in Natural Language processing, specifically pretrained Large Language Models (LLM), can help us solve this problem with the available data.

Being able to predict whether a TCR and an Epitope bind can be essential for Cancer treatment, while also allowing the attending doctor to view the patients entire immunohistory. This allows for a targeted treatment of the cancer - while saving a lot of money and time in finding a proper medication to combat specific cancer epitopes.
Inspired by TITAN and its approach to discovering which TCRs bind to which Epitopes [2]. We aim to discover if the use of Deep Learning can improve upon the TITAN score for TCR binding prediction, aswell as discovering if it is possible to predict if they bind. Going one step further to confirm whether the Model is able to generalize well enough to predict a completely unseen TCR-Epitope pair.

The secondary goal of our project is to discover whether the possibility exists to predict an unseen epitope with the existing data - or if there is a need to increase the data diversity to solve this problem.

## 1.4 Required Knowledge and Helpful Sources

We expect the reader to have a basic understanding of Data Science and Analysis. If you wish to reproduce our results, a strong understanding of Python or another programming language with deep learning libraries (such as R or Julia) will be necessary. We personally used Python with the PyTorch and PyTorch Ignite Frameworks to write our training and evaluation pipelines and which interface nicely with the ESM2 models [9, 10].

An understanding of deep learning is not strictly necessary, but will obviously be beneficial in understanding our work. Ian Goodfellow's book "Deep Learning"

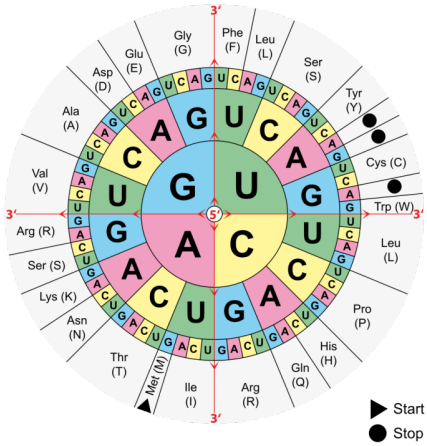



Figure 1: Visualisation of amino acids that make up the Genetic Code [13]

can provide you with all of the information you may need for this project thesis and more [11]. We will explain the Transformer Architecture in a bit of detail in the next section, but if you find our explanation lacking we recommend you have a look at Harvard's "The Annotated Transformer"[2] [12].

# 2 Theoretical Foundations

## 2.1 Genetic Code

In 1956, the Genetic code was discovered and formalized. Brittanica describes the genetic code as follows: "The Genetic code is composed of four nucleotides: adenine (A), guanine (G), cytosine (C) and uracil (U). Out of three of these building blocks, a single sequence is built. As an example, the nucleotides AUG is the amino acid methionine. This amino acid is also found at the start of every mRNA and indicates the start of a protein." All amino acids are made out of these four building blocks, and in image 1, the building blocks of the Genetic Code is shown. Both TCRs and Epitopes are made out of these basic buildings blocks [13].

## 2.2 Transformer Neural Network Architecture

Transformers have become the new standard architecture for many natural language processing tasks. The original paper "Attention is all you need" proposed use of the

[2]Available online under: http://nlp.seas.harvard.edu/annotated-transformer/

attention mechanism devised earlier for Recurrent Neural Networks in an Encoder-Decoder structure as a replacement for recurrent neural networks. This encoder-decoder structure works especially well for sequence-to-sequence tasks such as text-to-text or speech-to-text. Both modules can however be used separately for tasks that do not deal with sequence-to-sequence data. An advantage of the transformer architecture over Recurrent Neural Networks is its ability to process input in parallel, therefore being able to take advantage of modern hardware parallelism during training and inference. This allows transformer models to train much faster and scale to large numbers of parameters and larger quantities of input data. Also, the transformer's attention mechanism, which gave the original paper its name, allows the architecture to "focus" on certain features given some context improving model performance significantly.

As mentioned before, transformers are split into an encoder and a decoder. The former encodes an input sequence into an internal representation, whereas the latter uses the internal representation as well as the original input to predict a (new) sequence.

Figure 2 visualises the transformer architecture. Since we are only interested in a sequence-to-one classification, we only use the encoder section of the architecture.

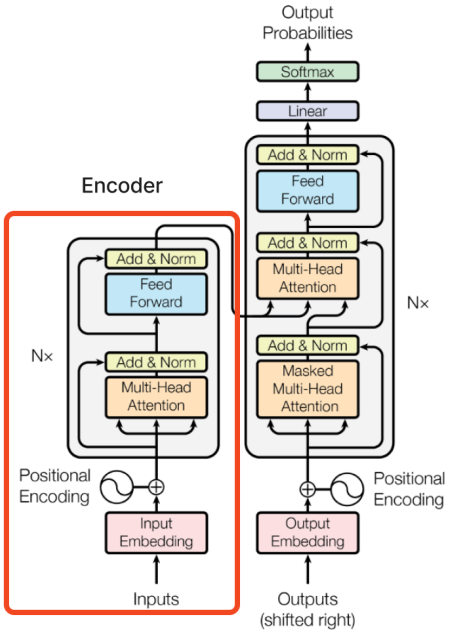



Figure 2: Visualization of the Transformer Architecture by Polosukhin et al. [14]

# 3 Methods

This section describes the different steps that were applied to achieve the results described in the next section.

## 3.1 Amino Acids Embeddings

The data used contains amino acids in a string format, which cannot be used directly for training our machine learning models. They therefore need to be transformed into a numerical representation suitable for Neural Networks.

The evolutionary scale models (ESM) are a group of large language models which are pre-trained on predicting masked amino acid sequences. While these models are only directly trained on predicting missing amino acids, they create complex internal numerical representations. These can be adapted for use in downstream tasks such as the TCR-Epitope binding prediction. This approach originally stems from the Natural Language Processing domain and takes away the need for extensively training a complex model, in order to learn these representations in the model itself. Instead, the ESM2 inference can be used to create the representation which can then directly be used as the input for a downstream model [5].

This approach significantly reduces the model complexity and therefore the time and cost required to train a downstream model.

The second version (ESM2) comes in six variants from eight million to 15 billion parameters. Amongst other things they include positional encodings, and showed significant improvements to their prediction capabilities. The authors observed that, "the ESM-2 model with 150M parameters performs better than the ESM-1b model with 650M parameters" [5].

### 3.1.1 Model selection

We have evaluated the two variants of the ESM2 model with 650 Million and 3 Billion parameters. The authors of the ESM2 model claim that embeddings from 3 Billion and 15 Billion parameter models show the best performance on downstream tasks, however smaller embeddings, for example from the 650 Million parameter model, require less compute power and models based on them can be trained more quickly.

### 3.1.2 Layer Selection

Previous work has shown that early layers in general-purpose models encode domain-specific information and seemed to, at least historically, produce better

results than later layers [15]. In order to verify this claim, we selected different layers from throughout the two ESM2 models.

### 3.2 Data Modification and Information

The authors of the TITAN paper used heterogeneous data from two sources, namely VDJdb and ImmunoCODE [2, 6, 16]. This dataset is unbalanced, with 191 unique Epitopes, while having 22'885 unique CDR3 TCR sequences. If the full TCR sequence is regarded, the number of unique TCRs climbs up to 23'139. In the TITAN Paper, all Epitopes with over 400 connections to TCRs are downsampled to this limit, while any Epitope having less than 15 TCR connections is excluded from the dataset. This approach is used to mitigate the effects of an unbalanced dataset - but it is unclear which effect this has on the model training performance. Deep Learning Models usually benefit from more data from which to extrapolate from. The authors of TITAN have already cleaned this data and therefore there are no empty values.[3]

We approach the problem by making three different train, validation and test data sets. This approach was chosen to explore how well the model is able to generalize under different conditions.

A data set is a set of entries fulfilling certain properties. In this experiment, we have three categories:

- A data set where the epitopes in the test and validation data set are not available to the model during training - which we we call from now on Epitope-unseen.
- A data set where the TCRs in the test and validation data set are not available during training - from now on TCR-unseen.
- And a data set where both TCRs and Epitopes in the test and validation data set are not available during training - from now on Both-unseen.

The first dataset, EPITOPE-Unseen allows us to see how well the model is able to generalize on unseen epitopes. This lets us estimate how well the model would predict other unseen epitopes it encounters in the real world.

The second dataset TCR-unseen, has all epitopes that have a connection with randomly chosen TCRs removed from the training set. This allows us to check how well the model is able to generalize in case the TCR is unseen instead.

[3]The data sets described here by the TITAN paper are available on their Github page: https://github.com/PaccMann/TITAN

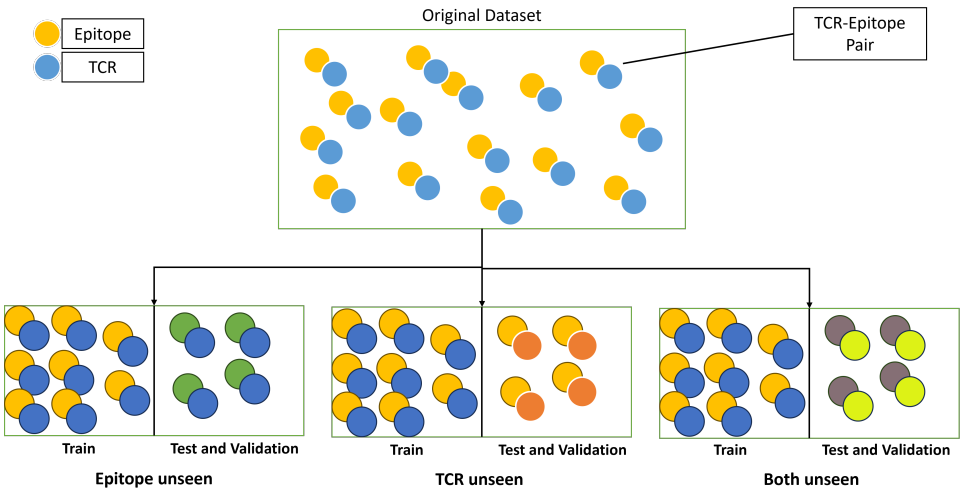



Figure 3: Visualization of the Datasplit

The third dataset, both-unseen, has a random epitope chosen and removed all its TCR connections with the train set. This is the most common encountered problem, where neither TCR nor Epitope is seen or known and creating a model that could accurately predict for this case would be the end goal.

All three data sets are used to check where more data needs to be acquired to properly predict a binding. It also helps see whether the low amount of variety in epitopes is a problem for the model training.

| **Dataset** | **Percent Binding** |
|---|---|
| epitope-train | 49.54% |
| epitope-test | 54.17% |
| epitope-validation | 49.29% |
| tcr-unseen-train | 50.00% |
| tcr-unseen-test | 50.00% |
| tcr-unseen-validation | 50.00% |
| both-unseen-train | 50.00% |
| both-unseen-test | 52.90% |
| both-unseen-validation | 58.82% |

Table 1: Comparison of binding versus non-binding entries in each data set

In Table 1 we see how the distribution of the data is after our split. The optimal case would be if the TCR-Epitope pair that binds reaches exactly 50.00%. Our

data split fulfills this requirement well, meaning our results with the models are not skewed too far in any direction. The 58.82% in the validation dataset of both-unseen is concerning, but can be taken into consideration for the evaluation of our experiments. We decided not to remove more data due it already being scarce. Due to the massive gap in diversity between Epitopes and TCRs, a perfect split is made possible in the data sets tcr-unseen-train, tcr-unseen-test and tcr-unseen-validation. This was not the case for Epitopes, which is why they are not balanced, with a 50% binding percentage.

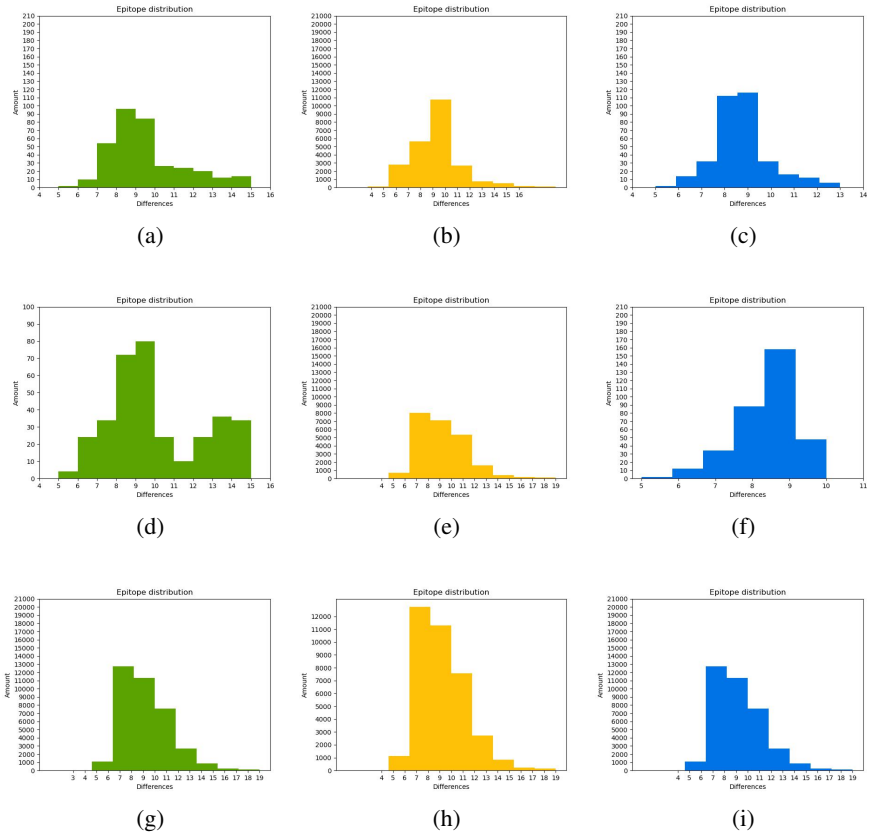



Figure 4: Shows the Levenshtein Distribution of different data sets
(a) both-unseen-test (b) both-unseen-train (c) both-unseen-validation
(d) epitope-unseen-test (e) epitope-unseen-train (f) epitope-unseen-validation
(g) tcr-unseen-test (h) tcr-unseen-train (i) tcr-unseen-validation

In Figure 4, the Distribution of the differences between the Epitopes in our data sets. This is done to check how the Epitope differences are distributed across our data sets. We chose the Levenshtein Metric [4] due to it being used in other papers and serving well to see how varied the Epitopes are. In 2, more information is provided for the Epitope Distribution. The data sets are all equal in distribution of differences.

The values where the Levenshtein difference is 0 are the cases where the same Epitope is compared to one another, or with itself. No Epitopes have a difference of two, three or four letters in genetic code. Starting at a difference of five, a slight

elevation is noticeable, exploding from six different letters on wards and increasing until reaching ten, when a steep decline is noticeable. The graphs in Figure 4 all look very similar due to the same dataset being used.

**Occurrences of Epitopes in the data sets**

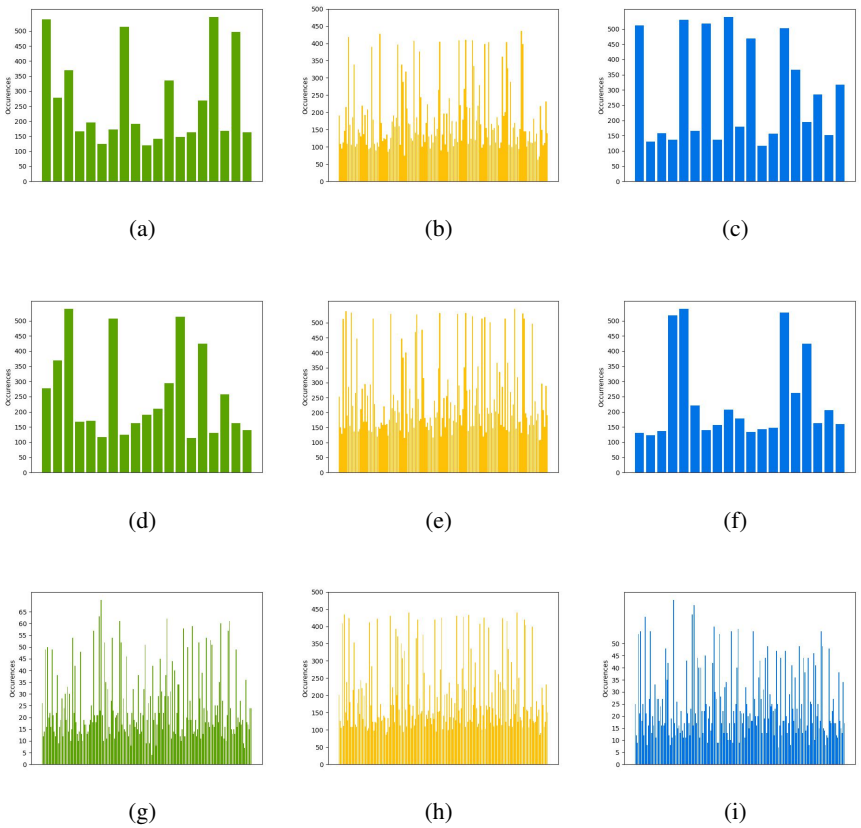


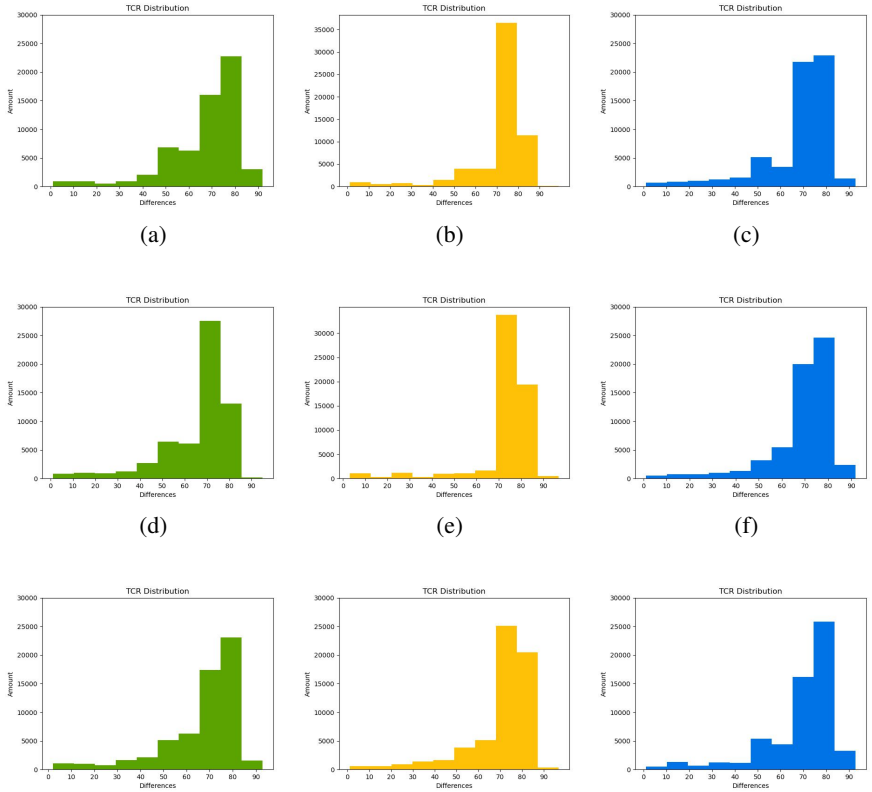

Figure 5: Shows how often a given Epitope is in the dataset
(a) both-unseen-test (b) both-unseen-train (c) both-unseen-validation
(d) epitope-unseen-test (e) epitope-unseen-train (f) epitope-unseen-validation
(g) tcr-unseen-test (h) tcr-unseen-train (i) tcr-unseen-validation

In Figure 5, the Occurrences of each Epitope in the data sets is shown. The distributions are vastly different in between the data sets and the subsets, consisting of training, validation and test. Most notably is the difference in Figure 5(g), which does not behave the same way as 5(a) and 5(b). This is due to the dataset being TCR unseen - where no importance was given to the way Epitopes are split. The three

training sets represented in 5(b,e,h) all appear similar in distribution. However there are less Epitopes in 5(b, e) than in 5(h). In the training set all 192 unique Epitopes are represented.

The difference in Epitope distribution is also mirrored in 5(c,f,i). In 5(c,f), less Epitopes are represented compared to 5(i).

**Levenshtein Difference of TCRs**




(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure 6: Shows the Levenshtein Distribution of TCRs
(a) both-unseen-test (b) both-unseen-train (c) both-unseen-validation
(d) epitope-unseen-test (e) epitope-unseen-train (f) epitope-unseen-validation
(g) tcr-unseen-test (h) tcr-unseen-train (i) tcr-unseen-validation

In Figure 6 the difference of TCRs in the different data sets are represented. There is a consistent uptick in the Levenshtein[4] difference in the 70 - 90 range in the genetic code. This is consistent over all TCRs, and is the CDR3 part of the TCR.

CDR3 is the most variable part of each TCR, and is the part thought responsible for binding [1]. This makes sure that the test, train and validation data sets are similar and represent a range of TCRs. Them not being dissimilar allows us to train and test a model and expect representative results that mirror real world use cases.

| Dataset | Count | Unique | Min/ | Max Occurrence | Levenshtein mean |
|---|---|---|---|---|---|
| epitope-train | 37803 | 154 | 142 | 546 | 9.142 |
| epitope-test | 4875 | 19 | 295 | 539 | 9.102 |
| epitope-validation | 4512 | 19 | 206 | 539 | 8.105 |
| tcr-train | 37766 | 192 | 111 | 440 | 9.133 |
| tcr-test | 4724 | 192 | 17 | 70 | 9.133 |
| tcr-validation | 4700 | 192 | 14 | 68 | 9.133 |
| both-train | 27664 | 154 | 106 | 436 | 9.181 |
| both-test | 5098 | 19 | 278 | 546 | 8.559 |
| both-validation | 5567 | 19 | 136 | 539 | 8.244 |

Table 2: General statistics for the Epitopes in the data sets

| Dataset | Count | Unique | Min/ | Max Occurrence | Levenshtein mean |
|---|---|---|---|---|---|
| epitope-train | 37803 | 22253 | 1 | 14 | 67.192 |
| epitope-test | 4875 | 4639 | 2 | 3 | 66.978 |
| epitope-validation | 4512 | 4278 | 1 | 3 | 67.485 |
| tcr-train | 37766 | 18613 | 2 | 18 | 67.228 |
| tcr-test | 4724 | 2314 | 2 | 12 | 67.136 |
| tcr-validation | 4700 | 2314 | 2 | 8 | 67.041 |
| both-train | 27664 | 13781 | 2 | 8 | 67.160 |
| both-test | 5098 | 4794 | 1 | 4 | 67.360 |
| both-validation | 5567 | 5235 | 1 | 4 | 67.004 |

Table 3: General statistics for the Full TCRs in the data sets

In Table 2 and 3 general information over the Epitopes and TCRs are represented, respectively.

### 3.2.1 Statistics for Epitopes in the data sets

In Table 2, the Count, the unique Occurrences, the Minimal Occurences, Maximum Occurences and the Mean of the Levenshtein [4] distance in each data sets.

With a much higher Count than unique Occurrences, very few unqiue Epitopes are known. [2] The highest amount of Epitopes are in the training data sets across

all data sets, excluding the tcr-unseen variation, where Epitopes are seen in training, compared to the other two. As control of similary between the epitopes in the different data sets, we use the Levenshtein [4] mean. With a Levenshtein mean range of 8.105 to 9.181, the Epitopes are equally different in the data sets. The Outliers are epitope-validation, both-test and both-validation, where very few Epitopes are represented, with 19 different epitopes. There being few Epitopes explains the difference in Levenshtein distance.

#### 3.2.2 Statistics for TCRs in the data sets

In Table 3, the Count, unique Occurrences, Minimal and maximal Occurrences and the Levenshtein mean between TCRs in the different data sets. The train, test and validation data sets are all relatively similar to their counterparts. Due to there being so many more TCRs than Epitopes, the unique numbers are across the board higher than in 2. This is also represented in the maximum Occurrence of TCRs across all data sets being only 18, compared to the 546 in Epitopes.

With the Levenshtein distance range being between 66.978 and 67.485, the TCR distribution across the data sets are all very similar.

### 3.3 Metrics

#### 3.3.1 Levenshtein Distance

In 1966, V.I. Levenshtein published the paper "Binary codes capable of correcting deletions, insertions and reversals". In this paper, the Levenshtein distance was introduced. Based off on Hamming Distance, that goes:

$$D(x, y) = \text{Number of coordinates where } x_i \neq y_i$$

The difference of the Hamming Distance to the Levenshtein Distance is the possibility of inserting new characters in certain spots. This extends the use of the Hamming distance significantly, allowing for a more general approach to handling Strings. To describe the difference between the TCRs and Epitopes, we use the Levenshtein distance [4].

#### 3.3.2 Receiver Operating Characteristic (ROC) curve

The receiver operating characteristic (ROC) is a metric that measures the true positive rate (TPR), also known as recall, against the False Positive Rate (FPR) for a given decision threshold of a classification model. The ROC curve then plots the TPR and FPR for different classification thresholds, see figure 7 [17].

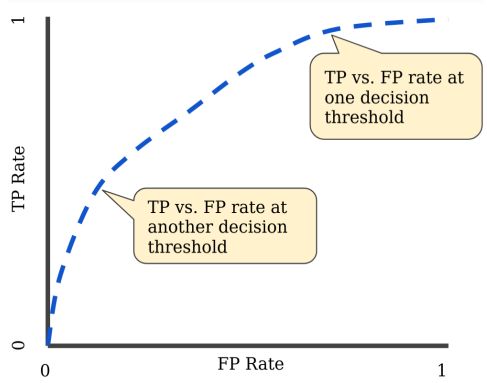



Figure 7: Visualization of an example ROC curve [17]

**True Positive Rate (TPR)** measures the ratio of True Positive (TP) classifications over the TP and False Negative (FN) classifications.

$$TPR = \frac{TP}{TP + FN}$$

**False Positive Rate (FPR)** measures the ratio of False Positive (FP) classifications over FP and True Negative (TN) classifications.

$$FPR = \frac{FP}{FP + TN}$$

### 3.3.3 Area under ROC Curve (ROC-AUC)

The area under the ROC curve is a metric used to compare different ROC curves. As the name suggests, it measures the area under the ROC curve and would be 1.0 for a model that gets all classifications correct and 0.0 for a model that gets all classifications incorrect. The ROC-AUC metric is commonly used in related work regarding TCR-Epitope binding prediction and serves to evaluate the performance of different models [2, 18, 15].

## 3.4 Model

### 3.4.1 Multi Layer Perceptron Baseline

Our baseline model for the TCR-Epitope binding affinity prediction is a shallow Neural Network classifier with a single hidden layer consisting of 512 neurons and using a rectified linear unit activation (ReLU) function.

### 3.4.2 Deep Neural Networks

For our second model architecture, we expand on the baseline Neural Network model by adding more Hidden layers. We expand the Model by up to three Hidden Layers and scale the size of the Model up to a maximum of 4096 Neurons. We use the ReLU function for these experiments, and the ADAM as optimizer for the model [19, 20]. These experiments allow us to check whether a bigger model is better able to encompass the information available in the Embeddings [5].

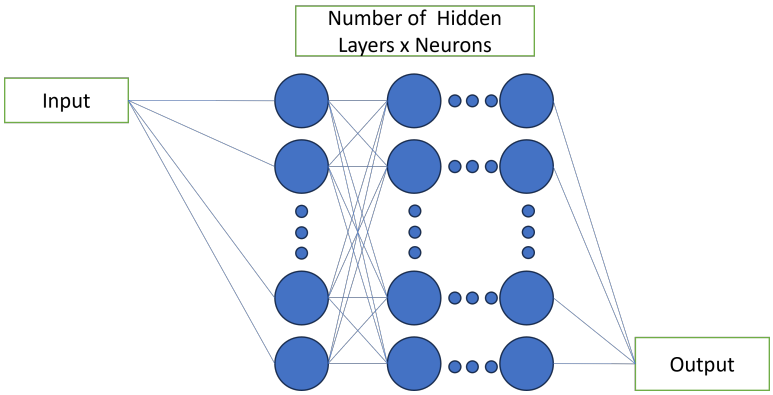



Figure 8: Neural Network

### 3.4.3 Transformer Encoder

Finally, we implemented a Transformer encoder architecture based on the Transformer Architecture proposed by Vasvani et al. in 2017 [14]. Figure 9 visualizes the following architecture: The amino acid sequences are embedded using one of the ESM models. Then a linear feed-forward layer projects the input to a smaller representation to deal with memory constraints, since the Transformer's model complexity scales quadratically with increasing size. Then, the encoder structure of the Transformer architecture is applied on the down-sampled output. Finally, a linear feed forward layer uses the internal representations learned by the transformer encoder to classify the samples into binding or non-binding. During training, we use Binary Cross Entropy as the target loss function.

### 3.4.4 Implementation

Our experiments were implemented in the Python programming language. The neural network architectures were created using PyTorch and training and evaluation of the models was facilitated by the PyTorch Ignite library [9, 10].

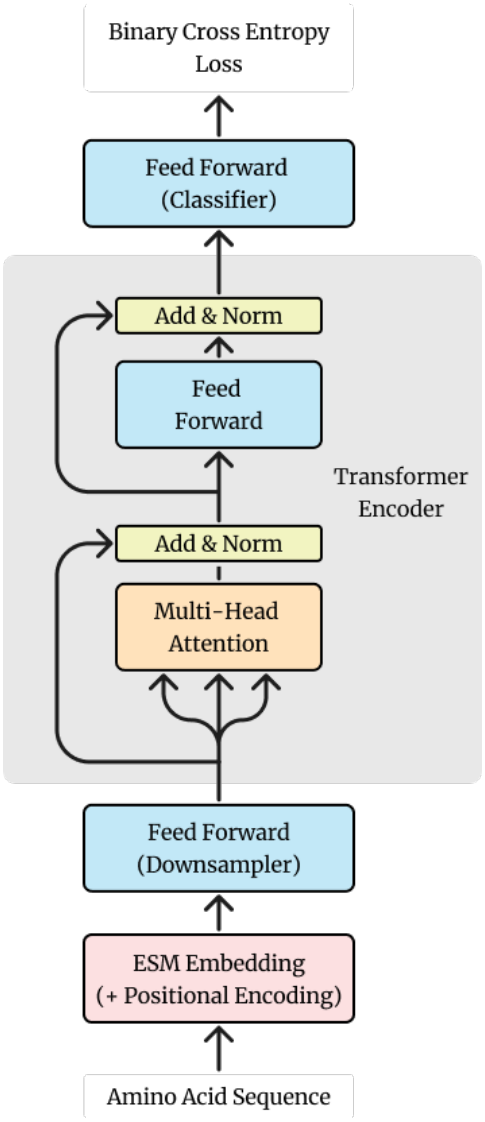



Figure 9: Adapted Transformer model architecture

### 3.5 Experiments

Multiple models were trained with distinct parameter configurations, as detailed below. However, the subsequent variables remained consistent across all models. The target function employed was binary cross entropy loss, and the optimizer function utilized was ADAM [19]. Evaluation metrics encompassed accuracy, ROC-AUC, and loss. During each training iteration the two most proficient models were preserved based on the ROC-AUC metric.

#### 3.5.1 Baseline Training

The multi layer perceptron baseline (described in section 3.4.1) was trained separately on embeddings that were extracted from the layers 6, 14, 21 and 33 in the 650 million parameters model[4]. Further, the training was split between the three types of data sets of TCR unseen, Epitope unseen and Both unseen.

#### 3.5.2 Transformer Encoder Training

We have attempted to train our Transformer Encoder with both the 650 Million and 3 Billion parameter variants of the ESM2 model. Furthermore we have used different 2, 4 and 6 heads and downsampling output sizes of 4500 and 5500 neurons. However, it seems that this specific architecture does not converge, meaning that it is unable to learn any patterns in the data.

## 4 Results

### 4.1 Baseline

Table 4 shows that the MLP baseline achieves an ROC-AUC score of 0.84 when the model has already seen the epitope during training. Conversely, this architecture is not able to generalize well when facing unseen epitope sequences were present during training.

One notable outlier to this is one model trained on the "both unseen" data set where TCRs and Epitopes from the training were not present in the validation and test data. Trained on the 14th layer of the ESM embeddings, it achieves an ROC-AUC score of 0.65 and an accuracy of 60% while also maintaining a F1-Score of 0.62, unlike the same models trained with the other ESM layers which had much lower F1-Scores. During validation, this model only ever achieved an ROC-AUC of 0.55, which suggests that these results are the result of random chance.

[4]The full model name is esm2_t33_650M_UR50D.

| Dataset | Layer | ROC-AUC | Accuracy | F1-Score |
|---|---|---|---|---|
| TCR-Unseen | 6 | 0.84 | 0.76 | 0.77 |
| TCR-Unseen | 14 | 0.73 | 0.54 | 0.21 |
| TCR-Unseen | 21 | 0.70 | 0.63 | 0.70 |
| Both-Unseen | 14 | 0.65 | 0.60 | 0.62 |
| TCR-Unseen | 33 | 0.63 | 0.60 | 0.55 |
| Both-Unseen | 33 | 0.57 | 0.49 | 0.39 |
| Both-Unseen | 6 | 0.55 | 0.50 | 0.29 |
| Both-Unseen | 21 | 0.51 | 0.51 | 0.55 |
| Epitope-Unseen | 6 | 0.51 | 0.51 | 0.20 |
| Epitope-Unseen | 33 | 0.48 | 0.56 | 0.26 |
| Epitope-Unseen | 21 | 0.47 | 0.48 | 0.31 |
| Epitope-Unseen | 14 | 0.45 | 0.46 | 0.42 |

Table 4: Best baseline models per data set evaluated on the test data and sorted by ROC-AUC score. Marked in red is the outlier that is discussed in the paragraph below.

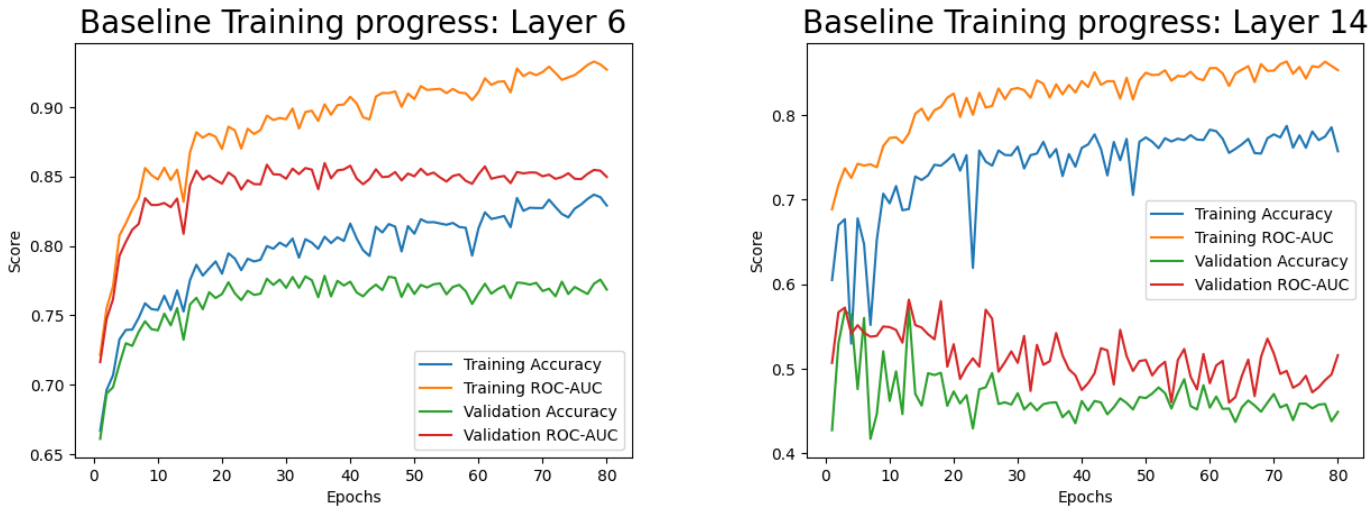
As mentioned Section 3.2 in Table 1, the "Both-unseen" test data set contains an unbalanced 58.82% binding combinations and 41.18% non-binding combinations. This might be the reason for this outlier, if the model had some kind of bias towards classifying more samples as binding than non-binding. Though this is unlikely given that both ROC-AUC and F1-Score are above 0.5.

### 4.1.1 Embedding layer selection

The different layers from the ESM embedding described in Section 3.1.2 seem to have a strong impact on the performance of the downstream model. As seen in table 4 layer 6 of the ESM model seems to capture the most relevant information for predicting the unseen TCR bindings, which is consistent with previous work [15].

Figure 10a shows the training and validation score progress across 80 training epochs. The green and red line visualize the validation scores. After 20 epochs the model seems to no longer improve on these scores, while the training scores keep improving up to 80 epochs.

Figure 10b then illustrates the increased difficulty of trying to predict bindings for unknown data, especially epitopes. While the training scores improve as before, the validation scores now are more volatile over the training epochs and get worse as the model gets better at predicting the training data.




(a) (b)

Figure 10: Training curves of the baseline

### 4.2 Deep Neural Networks

For the Deep Neural Networks, the 3 billion parameter model was used, named esm2_t36_3B_UR50D [5].

In Table 5, the Embedding layers 6 and 36 of each data set were used to train the Neural Network. With 2048 Neurons and three Hidden layers, the model did not improve on the Baseline result. This shows us that going for a deeper and larger architecture does not yield better results. The Baseline predicted the results better.

In Table 6, the layers 6 and 36 of each dataset was used to train the Neural

| Dataset | Layer | ROC-AUC | Accuracy | F1-Score |
|---|---|---|---|---|
| TCR-Unseen | 6 | 0.86 | 0.78 | 0.79 |
| TCR-Unseen | 36 | 0.69 | 0.65 | 0.68 |
| Epitope-Unseen | 6 | 0.45 | 0.47 | 0.16 |
| Epitope-Unseen | 36 | 0.39 | 0.42 | 0.44 |
| Both-Unseen | 6 | 0.63 | 0.55 | 0.40 |
| Both-Unseen | 36 | 0.49 | 0.49 | 0.55 |

Table 5: Experimental Results with 2048 Neurons, 3 Hidden Layers and no Dropout

| Dataset | Layer | ROC-AUC | Accuracy | F1-Score |
|---|---|---|---|---|
| TCR-Unseen | 6 | 0.72 | 0.69 | 0.72 |
| TCR-Unseen | 36 | 0.64 | 0.60 | 0.60 |
| Epitope-Unseen | 6 | 0.47 | 0.49 | 0.32 |
| Epitope-Unseen | 36 | 0.46 | 0.46 | 0.45 |
| Both-Unseen | 6 | 0.54 | 0.56 | 0.66 |
| Both-Unseen | 36 | 0.52 | 0.48 | 0.32 |

Table 6: Experimental Results with 512 Neurons, 3 Hidden Layers and 0.25 Dropout

Network. However, an increase in model and the addition of a Dropout layer did not help the model to generalize. The baseline model was better in predicting TCRs, Epitopes and Both. Again, the Baseline, with no Dropout and only 1 Hidden layer was better able to predict the data.

# 5 Discussion

The essential conclusion is that there is not enough of variety in epitope sequences where bindings are known. This complicates building models which are able to generalize to a point where they would be able to predict binding with much higher quality. However, a data set containing a larger variety of epitopes is required to make Neural Networks converge and being able to generalize well. A data augmentation to solve the lack of epitope diversity is very difficult due to neither the epitope nor the TCR appearing in the natural world if approached with common data augmentation techniques. It would also be impossible to know if the TCRs and epitopes bind.

## 5.1 Outlook and Further Research

The transformer architecture and attention mechanism is a very promising direction in the field of TCR-Epitope binding prediction. Not only are there advances in foundational models which can be used to create numeric representations, but there is also the possibility of using the Attention mechanism directly for the binding prediction itself. We have attempted to implement this in form of the Transformer Encoder described in Section 3.4.3. Unfortunately, this architecture does not converge. It is likely that the down-sampling step introduced by us for memory limit reasons is the key factor that leads to the model failing to converge. Further research may look into tweaking this approach to work with the smaller ESM2 embeddings in order to avoid having to down-sample. Another promising approach would be to

train an auto-encoder[5] on the larger embeddings to compress them into a less sparse format which effectively means extracting the down-sampling step from inside of the model to a separate model, which could improve the classifier's stability.

# 6 Appendix

## 6.1 Description of Project in Complesis

T-Zellen sind eine Art von weißen Blutkörperchen, die eine entscheidende Rolle bei der adaptiven Immunantwort spielen. Die Fähigkeit der T-Zellen, spezifische Epitope zu erkennen und an sie zu binden, ist entscheidend für die Fähigkeit des Immunsystems, eine angemessene Immunantwort auszulösen. Das Verständnis der Bindungsinteraktionen zwischen T-Zell-Epitopen und TCRs ist von entscheidender Bedeutung für die Untersuchung von Immunreaktionen, die Konzeption von Impfstoffen und die Entwicklung von Immuntherapien für verschiedene Krankheiten, darunter Infektionen, Autoimmunerkrankungen und Krebs. Ziel dieser Arbeit ist es daher, die Fähigkeit modernster Deep-Learning-Ansätze zur Vorhersage der T-Zell-Epitop-Bindungsaffinität zu erforschen und auf diese Weise einen Schritt in Richtung personalisierte Immuntherapie zu machen.

## 6.2 Time table

| Week 01 | Week 02 | Week 03 | Week 04 | Week 05 | Week 06 | Week 07 |
|---|---|---|---|---|---|---|
| Related Work Research | Related Work Research | Related Work Research | Related Work Research | Create embedding pipeline with SMILES (TITAN Paper) (and other related Work architectures) | Create Neural Network Baseline (LLM Solution, using SMILES embedding) | Research the ESM Model, attempt a first implementation on a dummy dataset |
| Week 08 | Week 09 | Week 10 | Week 11 | Week 12 | Week 13 | Week 14 |
| Get a first working MLP as Baseline instead of TITAN, hoping to get a comparable score with TITAN | Attempt to use a Transformer to improve the model design | Experimentation on Transformer architecture | Experimentation on Model | Experiment on Model | Finalize Model & Work on Documentation | Improve & Finalize Documentation |

[5]This was a suggestion by our supervisor, Dr. Jasmina Bogojeska.

# References


[1] L. et al., "Estimating t-cell repertoire diversity: limitations of classical estimators and a new approach," 2015.

[2] R. M. M. Weber A, Born J, "Titan: T-cell receptor specificity prediction with bimodal attention networks." *Bioinformatics*, vol. 37, pp. 237–244, 2021.

[3] G. B. Goh, N. O. Hodas, C. Siegel, and A. Vishnu, "Smiles2vec: An interpretable general-purpose deep neural network for predicting chemical properties," 2018.

[4] D. A. Nauk, "Binary codes capable of correcting deletions, insertions, and reversals," 1965.

[5] Z. L. et al., "Evolutionary-scale prediction of atomic-level protein structure with a language model," *Science 379*, pp. 1123–1130, 2023.

[6] B. et al., "Vdjbase: an adaptive immune receptor genotype and haplotype database," 2019.

[7] K.-Y. Z. et al., "The cancer epitope database and analysis resource: A blueprint for the establishment of a new bioinformatics resource for use by the cancer immunology community," *Frontiersin*, 2021.

[8] S. R. Dubey, S. K. Singh, and B. B. Chaudhuri, "Activation functions in deep learning: A comprehensive survey and benchmark," 2022.

[9] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[10] V. Fomin, J. Anmol, S. Desroziers, J. Kriss, and A. Tejani, "High-level library to help with training neural networks in pytorch," https://github.com/pytorch/ignite, 2020.

[11] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[12] A. Rush, "The annotated transformer," in *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, E. L. Park, M. Hagiwara, D. Milajevs, and L. Tan, Eds. Melbourne, Australia: Association for Computational Linguistics, Jul. 2018, pp. 52–60. [Online]. Available: https://aclanthology.org/W18-2509

[13] jgi.doe.gov, "Proving the genetic codes flexibility," *jgi.doe.gov*.

[14] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[15] N. Deutschmann, A. Pelissier, A. Weber, S. Gao, J. Bogojeska, and M. R. Martínez, "Do domain-specific protein language models outperform general models on immunology-related tasks?" *bioRxiv*, 2023. [Online]. Available: https://www.biorxiv.org/content/early/2023/10/26/2023.10.17.562795

[16] N. S. et al., "A large-scale database of t-cell receptor beta (tcr$\beta$) sequences and binding associations from natural and synthetic exposure to sars-cov-2," 2020.

[17] Google, "Classification: ROC Curve and AUC," 2022. [Online]. Available: https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc

[18] P. Meysman, J. Barton, B. Bravi, L. Cohen-Lavi, V. Karnaukhov, E. Lilleskov, A. Montemurro, M. Nielsen, T. Mora, P. Pereira *et al.*, "Benchmarking solutions to the t-cell receptor epitope prediction problem: Immrep22 workshop report," *ImmunoInformatics*, vol. 9, p. 100024, 2023.

[19] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017.

[20] A. F. Agarap, "Deep learning using rectified linear units (relu)," 2019.