EpitopePredictions vs MHCnuggets

How they work and how they differ.

Jasper Bakker, Hidde Nauta, Owen Moorman

Introduction

We have two programs (MHCnuggets and Epitope Predictions) which both predict how likely a certain epitope is to be shown on the outside of the cell membrane with different haplotypes. This is important in the recognition of one's own body cells. If a cell presents an epitope that is foreign to the immune system, it will be killed, which is beneficial for the body, as the cell could be malfunctioning or infected by a virus. This presentation of epitopes also plays a key role in vaccine development. In order for a vaccine to be effective you want the epitopes to be presented as often and in as many cases as possible, therefore the vaccine has to contain epitopes for all immune types. Thus the immune system can quickly detect it and develop antibodies against the pathogen. Also, this can be used to make sure that this happens on every different haplotype, so people with different immune systems all get immune quickly. However, these programs output some very different results and thus it is unknown if the given predictions are trustworthy or not. In this paper we discuss the differences between these programs, why they are caused, and how this affects the usefulness of the results.

Hypothesis

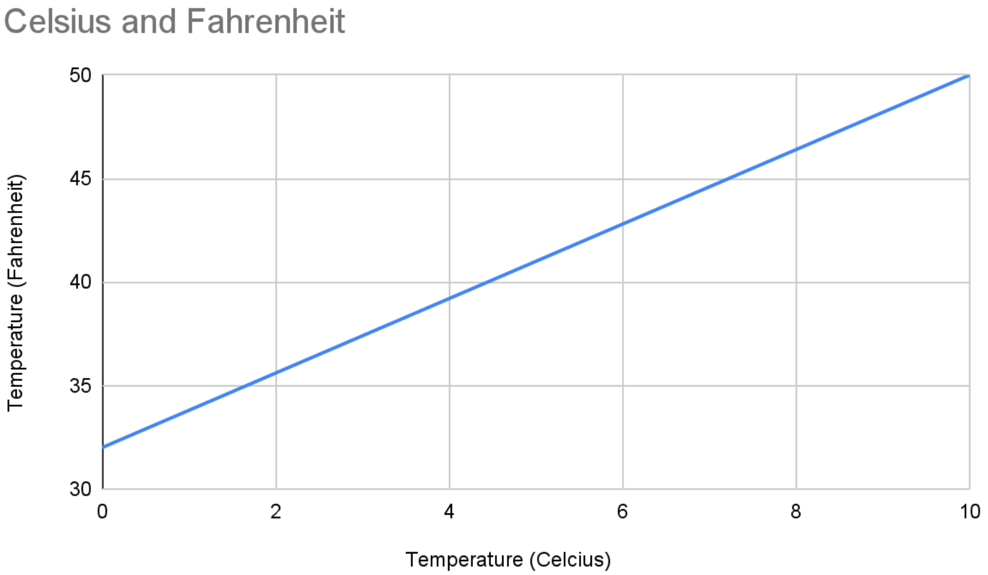
Since both programs are trying to predict the same thing, we expected the results to be equal, or at least very similar. We expected the results to be fairly close together. However, as will be shown in the results section of this paper, both programs output different results, they are not always equal nor are they always fairly close together. We knew this could be caused by the use of different scales for both programs, just like thermometers can give temperature in celsius or fahrenheit and give thus different outputs. In this case the numbers differ, but the temperature stays the same. We thought the same might be happening here. If this is happening, we expect the highest results of EpitopePredictions to also be the highest result from MHCnuggets, and vice versa.

Methods

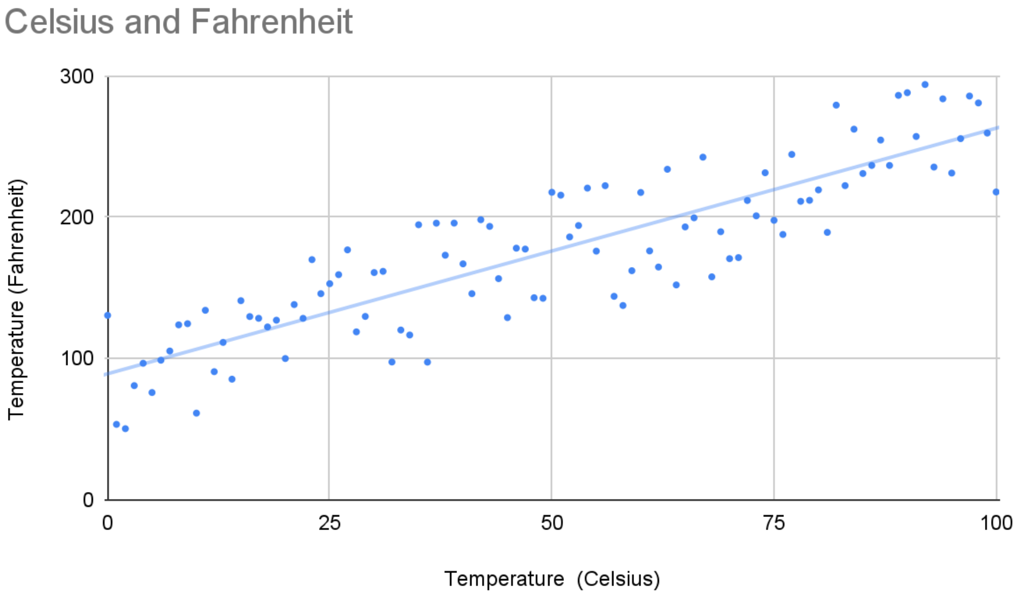
Correlation

Here we will be showing our methods for determining whether or not there is a correlation between the predicted results from both programs. We demonstrate this using some examples.

In this table we have the temperature in two different units: Celsius and Fahrenheit. If we were to plot this table, we would get the graph below. The x-axis shows the temperature in Celsius and the y-axis shows the temperature in Fahrenheit. In this example it is easy to see the correlation: as we increase the value of the x-axis, the y-axis follows. While such a correlation is easy to spot, it becomes more difficult when there is noise involved. For this example we expect a trendline with the following formula: Temperature (°F) = (Temperature(°C) × 9/5) + 32. If we look at the graph we see it starts at 32, which results in the +32. The slope of the graph is equal to $\frac{18}{10} = (\frac{\Delta T(°F)}{\Delta T(°C)}) = \frac{9}{5}$, which corresponds to the expected slope.

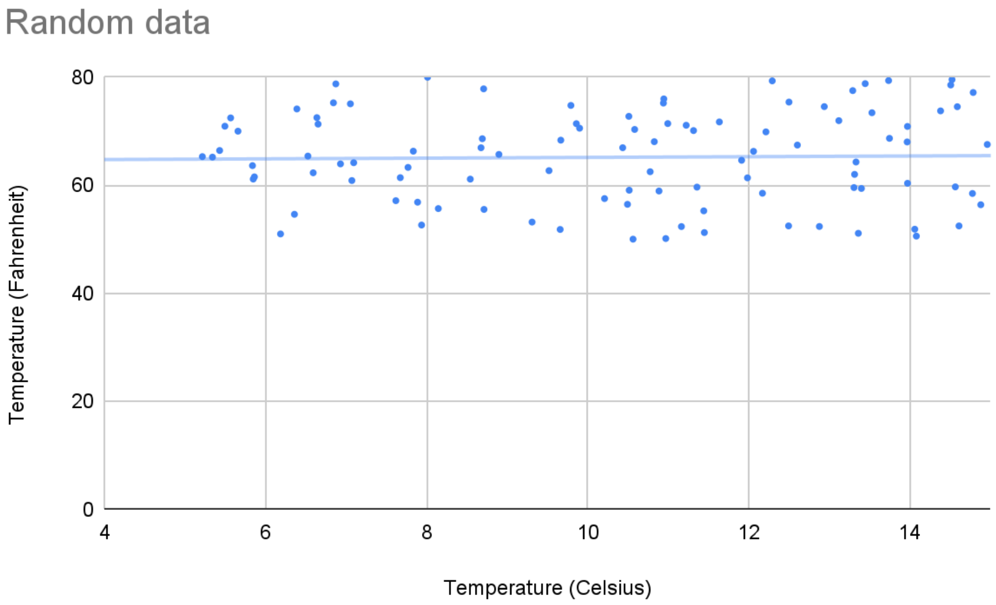


The example below is exactly the same as the previous example, but we added some noise to the measurements. Though, at first glance, it might not seem like there is a clear correlation between the two, a trendline shows that this assumption is false. We still have the same correlation between the temperature in Celsius and the temperature in Fahrenheit. Here, the trendline has an equation of $y = 1,7391x + 89,124$. This slope (1.7391) is very similar to the expected slope of $\frac{9}{5} = 1.8$



For our last example we have a graph with random data. Here we see there is no correlation: as we increase the value on the x-axis, the y-axis stays the same, this can be seen using the trendline, which has an equation of $y = 0,0036x + 65$. This slope is very close to 0, indicating there is no correlation.

We also have a way to quantify this correlation, using the Pearson Correlation Coefficient.



Pearson Correlation

Using the Pearson Correlation Coefficient we are able to find a correlation between the results of MHCnuggets and EpitopePredictions for each different haplotype. See the formula down below.

$$r = \frac{\Sigma(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\Sigma(x_i - \bar{x})^2 \Sigma(y_i - \bar{y})^2}}$$

r = correlation coefficient

x_i = values of the x-variable in a sample

$\bar{x}$ = mean of the values of the x-variable

y_i= values of the y-variable in a sample

$\bar{y}$= mean of the values of the y-variable

The Pearson correlation always has an output ranging from -1 to 1. A value of -1 meaning a fully negative correlation, for example an equation with a slope -1, -0,342 or -(3,5*10^{34}). The opposite is true for a slope of 1, this means a fully positive correlation, for example an equation with a slope of 1, 0,342 or 3,5*10^{34}.

If we were to calculate the r values of above graphs we would get the following results (here we used the values of celsius from 0 to 100 instead of 0 to 10 to have more data points):

- Celsius and Fahrenheit: 1
- Celsius and Fahrenheit with noise: 0.8744
- Noise: 0.0227

As we can see here the values from the first graph have a perfect positive correlation, the slope between 2 values is the same for every value. The second graph, with noise, has a lower correlation due to the many fluctuations in it's slope. The third graph that is supposed to be random has a tiny positive correlation, this is probably due to the graph being not completely random and due to noise as the trendline also has a slight positive slope (0,0036).

Spearman Correlation

While it seems as if this coefficient could provide solid evidence as to whether there is a correlation or not, the Pearson correlation coefficient has some drawbacks. To illustrate this we use the following graphs:

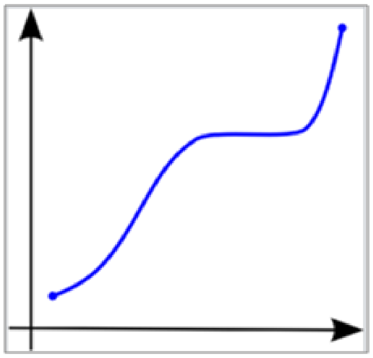


Figure 1 - A monotonically increasing function

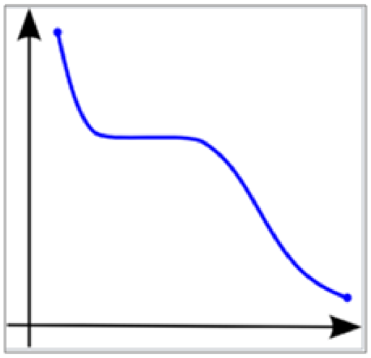


Figure 2 - A monotonically decreasing function

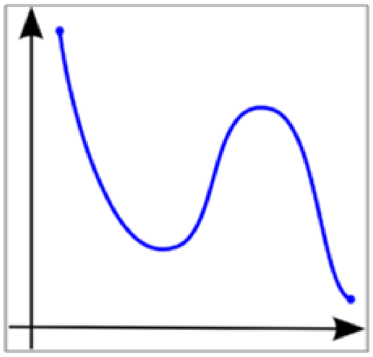
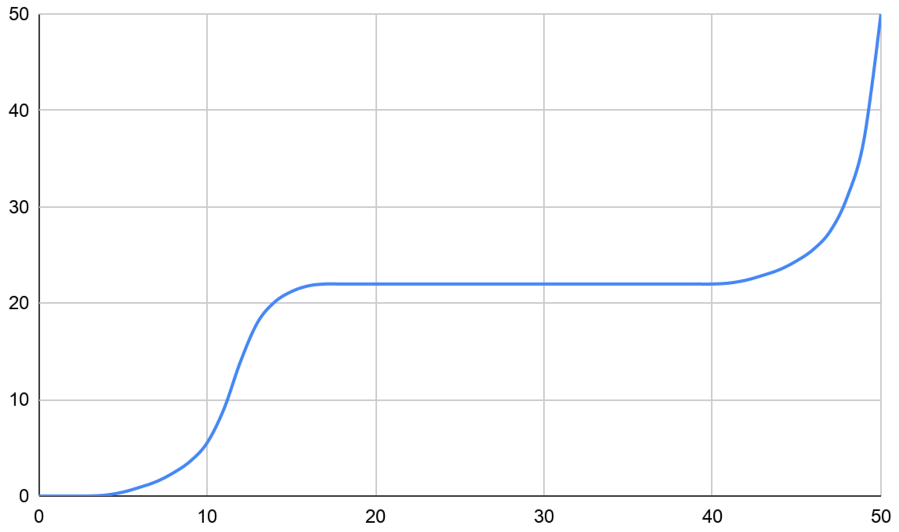


Figure 3 - A function that is not monotonic

It is clear there is a correlation between the variables, but this correlation is non-linear. If we were to input these values into the Pearson formula, we get a correlation of 0.843, which is still considered quite strong, but the Pearson Correlation Coefficient is not meant to be used on a non-linear graph like this one. To work around this problem, we will also be testing if there is a non-linear correlation. For this we would need a coefficient to determine correlation between monotonic, or monotone, sets of data. Monotonic meaning they both tend to go up, or down, together.



This is where the Spearman Correlation comes into play. Using the following formula, we are able to determine how strong a monotonic increase or decrease is between the two datasets, if there is any.

$$r_s = 1 - \frac{6\,\Sigma\, D_i^{\,2}}{N^3 - N}$$

r_s = correlation coefficient
N = the number of data points
D_i = $rg(X_i) \ - \ rg(Y_i)$

When we input the same graph in the Spearman formula, we get a correlation of 0.946. Which is very close to one, meaning there is a very strong correlation between the variables in the graph. If this graph were less perfect, and predictable, the Pearson Correlation could indicate there is no correlation, or a very weak one, while there may still be a monotonic relation present. That is why we will also be testing every haplotype to see the Spearman correlation, to determine if there is a monotonic relationship.

And if we were to use the Spearman correlation on the same values as we did with Pearson's we get the following results:

- Celsius and Fahrenheit: 1
- Celsius and Fahrenheit with noise: 0.8766
- Noise: 0.0310

As can be seen the values are fairly similar, meaning that Spearman's correlation can still be used reliably on linear graphs.

But this is not where it ends. As there are non-linear graphs that are not monotonic, like a parabola. For this type of correlation we will be plotting the haplotypes as well, to see if there is a different type of correlation visible.

Additionally, the Stabilized Matrix Method article indicated that the regular IC50 value, but also the Log(IC50) value can be used for EpitopePredictions. We will be testing both values to see which one we need to use. Since all results from EpitopePredictions should match a normal distribution we can determine if the Log-value is necessary, or if the normal IC50 should be used. To test for a normal distribution a histogram chart can be used. A histogram is a bar chart which splits data into different sets along the x-axis. The y-axis shows how many data points fit into that set,

Biology

Before we dive into the two models and the results it is first necessary to understand what kind of values these models actually predict and what those values mean in the physical world. Therefore we will dive into the biological side of EP and MCHn.

How are antigens presented to the immune system?

The immune system determines whether a cell in the body is invasive, like a bacteria, and should thus be killed, or a body's own cell, like a blood cell or a liver cell.

This is usually accomplished by checking whether the antigen that is presented on a MHC-I (Major Histocompatibility Complex) or a MHC-II molecule. These two molecules have nearly the same name, but differ a fair amount. MHC-I molecules are found on the surfaces of all nucleated cells, cells with a nucleus. At that place they present epitopes found inside the cell itself. MHC-II molecules are only found on the surfaces of antigen presenting cells (APCs) like macrophages or phagocytes. Thus MHC-I is used for the detection of infected cells by, for example viruses and MHC-II is used for presenting the antigens of a virus or bacterium to other cells in the immune system, for example, T helper cells or B cells.[4]

For viruses this can be determined by verifying the antigens on the virus' viral envelope or when a certain cell is infected and is showing a foreign antigen on one of it's MHC-I proteins. The viral envelope is a defensive shell to protect the DNA inside it.[14] When, for example, a T_c cell detects one of those antigens as presented by MHC-I it will determine whether it is foreign or not, if that is the case the T_c cell will kill the infected cell by disintegrating it's membrane with proteins. It will also start cloning itself rapidly to be able to more quickly notify other cells necessary in the defense against the virus like B-cells or macrophages.

For bacteria this process is fairly similar, the differences stem from the fact that usually a bacterium's antigens aren't directly presented on an infected cell's membrane. But only on the *Bacterium's* membrane, thus a T_c cell can't detect so bacteria can't be detected by the immune system when they are inside of a cell.[16]

How does IC_{50} relate to this?

IC_{50} (half maximal inhibitory concentration) is a biological unit that describes how much of a certain substance is required to induce a certain biological processor component by 50%. This could be, for example, a drug, enzyme or cell. The scale often used for expressing IC_{50} values is molar concentration (mol/L)[15].

In our case the IC_{50} value corresponds with how well a certain epitope binds to MHC-I. IC_{50} refers to the amount of a certain substance, so a lower IC_{50} value corresponds with an epitope being more likely to be presented.

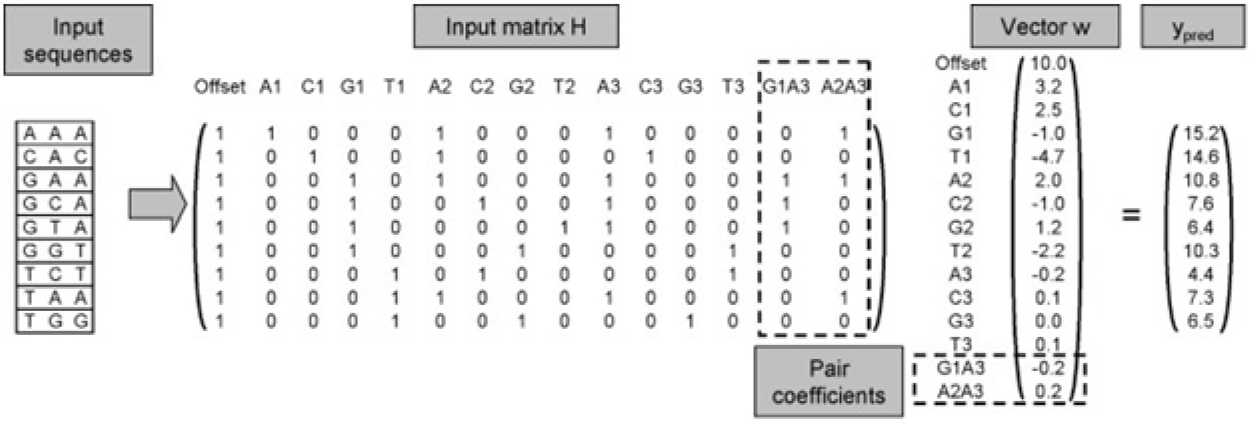
Epitope Predictions

EpitopePredictions is one of the programs used to predict the IC50 value of a randomly generated epitope on a chosen human haplotype, but it can also do this for alleles from mice, chimpanzees and rhesus macaques [1]. To do this it uses the so called stabilized matrix method (SMM), developed by Bjoern Peters and colleagues and uses Peptide:MHC binding energy covariance (PMBEC) as a Bayesian prior, which, in turn was developed by Johannes Textor and colleagues to improve current peptide:MHC binding prediction methods.

Stabilized matrix method (SMM)

The stabilized matrix method is one of the methods used for the prediction of MHC binding. Other methods include neural networks, like the one we use: MHCnuggets. Another example of a neural network is NetMHC, which was shown to be the current best prediction program, according to a large-scale benchmark [2]. The program uses XML files for input and output, where each amino acid in an epitope is encoded as a binary vector with a length of 20, because there are 20 amino acids. Every position is set to 0 except for the one coding for the specific amino acid. An example of this could be 00001000000000000000, which, depending on the order in which the amino acids are ranked (in this case alphabetically by their one letter code [18]), would indicate that phenylalanine is present here. These vectors are then stacked up and form a so-called input matrix which is multiplied by a weight matrix, w, resulting in a new matrix, the prediction matrix.

An example for this can be found in the SMM article [1]. Here, instead of amino acid sequences, nucleic acid sequences were used. These can be seen under Input sequences. The letters correspond to each of the four nucleic acids found in DNA, T for Thymine, A for Adenine, C for Cytosine and G for Guanine. The input matrix turns these sequences into a binary vector. For each position, all nucleic acids are set to 0, except for the one present at that position. This nucleic acid is represented by a 1. When we look at the first row of Input matrix H, we see A1 is set to 1, indicating an A, Adenine molecule is present at position one. C1, G1 and T1 are set to 0 which indicates they are not present at position one in the Input sequence. This is done for each position resulting in Input matrix H. This matrix is then multiplied by weight vector w, outputting the ypred, or predictions, matrix.

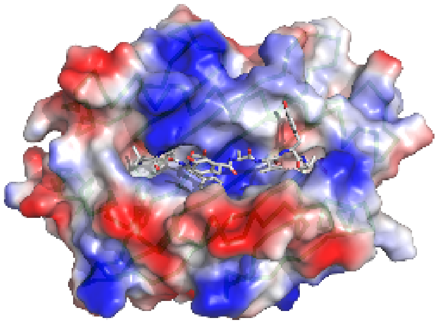


A	A	A
C	A	C
G	A	A
G	C	A
G	T	A
G	G	T
T	C	T
T	A	A
T	G	G

$$\begin{array}{ccccccccccccc|cc}
\text{Offset} & A1 & C1 & G1 & T1 & A2 & C2 & G2 & T2 & A3 & C3 & G3 & T3 & G1A3 & A2A3 \\
1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\
1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\
1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0
\end{array}
\quad
\begin{array}{rr}
\text{Offset} & 10.0 \\
A1 & 3.2 \\
C1 & 2.5 \\
G1 & -1.0 \\
T1 & -4.7 \\
A2 & 2.0 \\
C2 & -1.0 \\
G2 & 1.2 \\
T2 & -2.2 \\
A3 & -0.2 \\
C3 & 0.1 \\
G3 & 0.0 \\
T3 & 0.1 \\
G1A3 & -0.2 \\
A2A3 & 0.2
\end{array}
=
\begin{pmatrix} 15.2 \\ 14.6 \\ 10.8 \\ 7.6 \\ 6.4 \\ 10.3 \\ 4.4 \\ 7.3 \\ 6.5 \end{pmatrix}$$

The weight vector w is derived from the hydrophobicity of each of the amino acids. Hydrophobicity is the physical property of a molecule that is seemingly repelled from a mass of water [10]. An every-day example for this is oil. Oils are hydrophobic, meaning they do not mix with water and are repelled by it. This is also the case for some amino acids. This is where MHC class I is important. The MHC I molecule is built up of α- and β-chains [9], which differ between haplotypes. These chains form a groove, in which a peptide can bind. The MHC I groove is closed and because of this, mostly short epitopes can properly bind to it, though research has shown longer epitopes do sometimes bind to MHC I [3]. These epitopes are mostly between 9 and 11 amino acids long. However, epitopes with different lengths often use alternative binding grooves, which complicates predictions [3]. Because of this, Epitope Predictions uses epitopes that are 9 amino acids long, because they are most common [11], and to simplify predictions.

The MHC-I molecule is partly hydrophilic [23], meaning it mixes well with, and is attracted to, water, and partly hydrophobic. This means that a peptide consisting of hydrophobic amino acids is repelled by the hydrophobic cleft in the MHC-I molecule. The distribution of hydrophobic and hydrophilic amino acids and their positions determines how well a certain epitope can bind to MHC-I. The weight vector is chosen in such a way that the predicted values correspond to measured values. This is done with multiple sets of data and then the average weight value of all these tests is used in the final matrix.



The stabilized matrix method also has a built-in method to suppress the effects of noise in input data. A positive scalar is added, which is determined by taking the average of multiple training sets. A scalar can be defined as something which only has a magnitude but no direction. Examples of scalars are temperature and pressure. A certain point in space has a temperature value (magnitude) but no direction, since temperature does not work in a direction. This scalar shifts all optimal entries in w closer to 0.

We use the following example. Say Ypred, the predicted value is equal to 4 and the measured value, ymeas is equal to 5 and the scalar is set to 0. In this case w needs to be adjusted to minimize the difference between these values since we want the predicted results to be as close to the measured results as possible. If we change the value of our scalar to a positive value that is not equal to 0, this shifts all optimal entries for w (the entries that minimize difference between ypred and ymeas) closer to 0. When values of w are lower, this means noise is multiplied by a smaller number and is thus reduced. This process is called regularization [2].

Peptide:MHC binding energy covariance (PMBEC)

PMBEC is a new amino acid similarity matrix and was created using experimental data. Like many other matrices used for the same or similar purposes, PMBEC takes into account well-known physiochemical (chemistry of organs and tissues of the body) properties of amino acid residues [2]. Amino acid residues are two or more amino acids linked together, a peptide, where the elements of water have been removed [25]. PMBEC differs in cases where a charged amino acid is changed for an oppositely charged amino acid. An example for this is substitution of Glutamic Acid with Arginine. Glutamic Acid has a negatively charged chain in its molecular structure, while Arginine has a positively charged chain [18].

EpitopePrediction uses PMBEC as a Bayesian prior for SMM. A Bayesian prior, short for prior probability distribution, refers to the Bayesian inference, which is a method of statistical inference in which Bayes' theorem is used [24]. Bayes' theorem "describes the probability of an event, based on prior knowledge of conditions that might be related to the event. [24]" This is used to update the probability for a hypothesis as more information is available. Meaning the prediction method can be refined by benchmarking it against measured data.

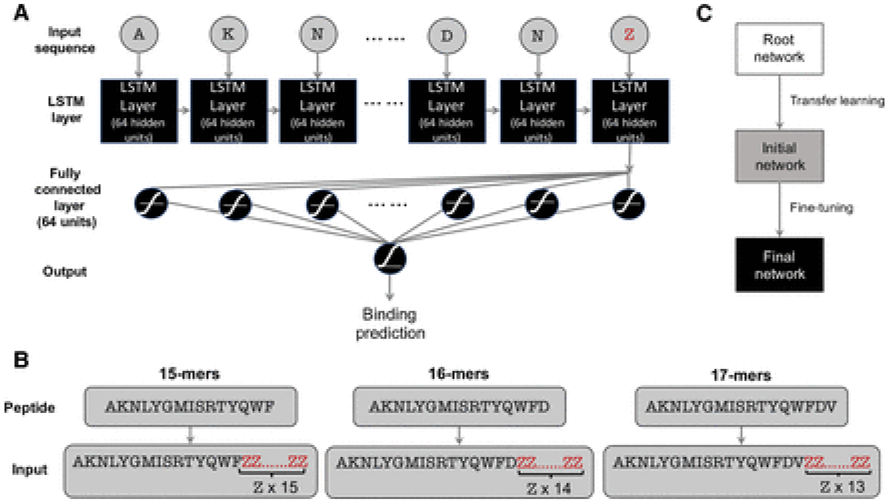
The researchers compared results from PMBEC to 10 other prediction matrices, one of which was BLOSUM50. PMBEC and BLOSUM50 had a Pearson Correlation of 0.64 between each other, while BLOSUM50 had correlation of more than 0.93 with seven of the other matrices, indicating PMBEC is different from most algorithms.

The article about PMBEC describes how $\text{SMM}^{\text{PMBEC}}$ has significantly better prediction accuracy than SMM, especially when the amount of data was smaller. They also found the average difference in performance between $\text{SMM}^{\text{PMBEC}}$ and NetMHC, which is currently the best method for prediction peptide binding to MHC-I is not statistically significant. Meaning they are almost identical in results.

The main benefit of $\text{SMM}^{\text{PMBEC}}$ over NetMHC is that $\text{SMM}^{\text{PMBEC}}$ is easier to understand and way simpler, while performance is about equal.

MHCnuggets

MHCnuggets[27] is another program which predicts an IC_{50}-value. MHCnuggets is developed to to work around the limitations other predictors had, by using a neural network that predicts the peptide-MHC bindings. Some limitations MHCnuggets solves are limited support for rare MHC alleles, the slowness of real experiments, and no support for peptides of longer length. MHCnuggets uses a method called LSTM(long short-term memory)[26] which is good at processing peptides which can be of any length. The neural network got trained with a method called transfer learning, which in this case means the network uses the data(binding affinity, IC50-value) from other alleles to predict the IC_{50}-value of an unknown (to the program) allele. The neural network gets this data from real tests, or from previously-predicted alleles. For the best results, it uses data from the most similar alleles.

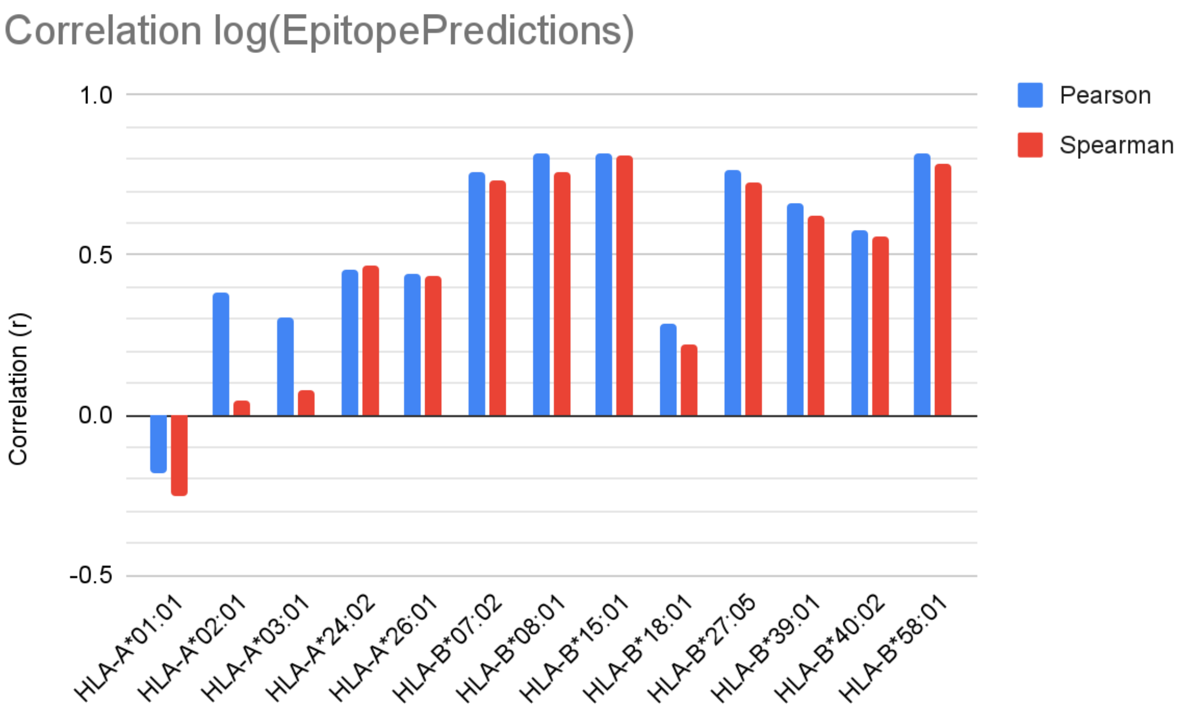


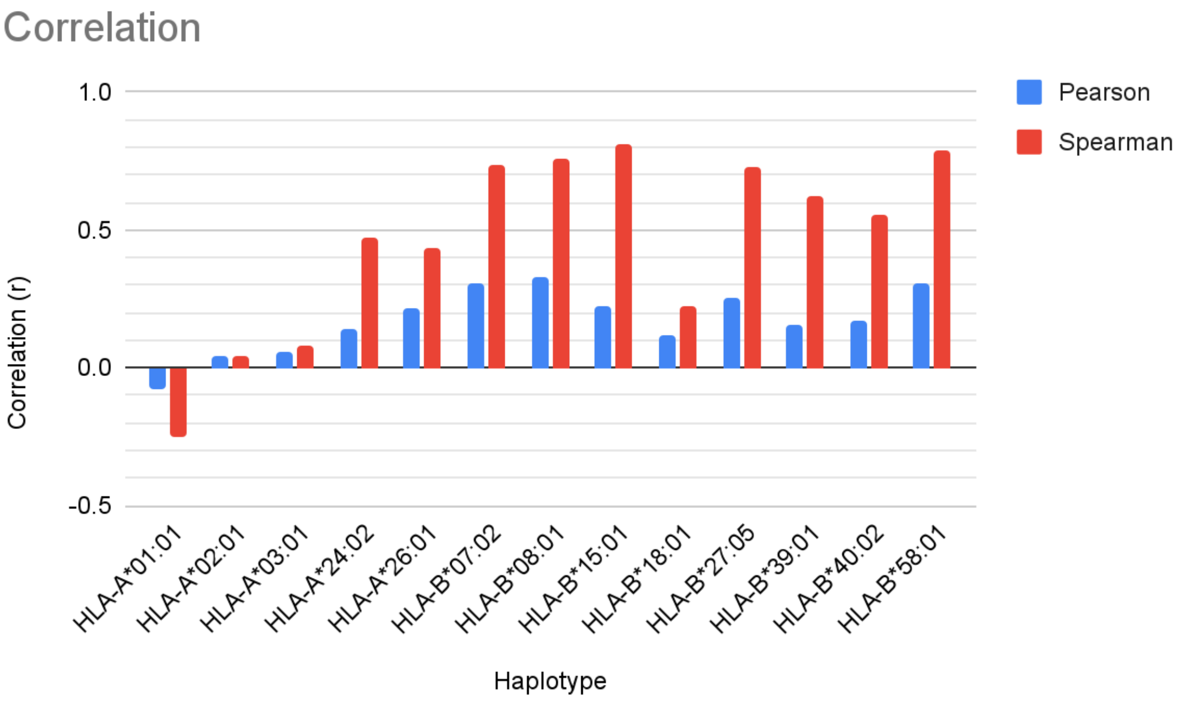
Visualisation of the neural network

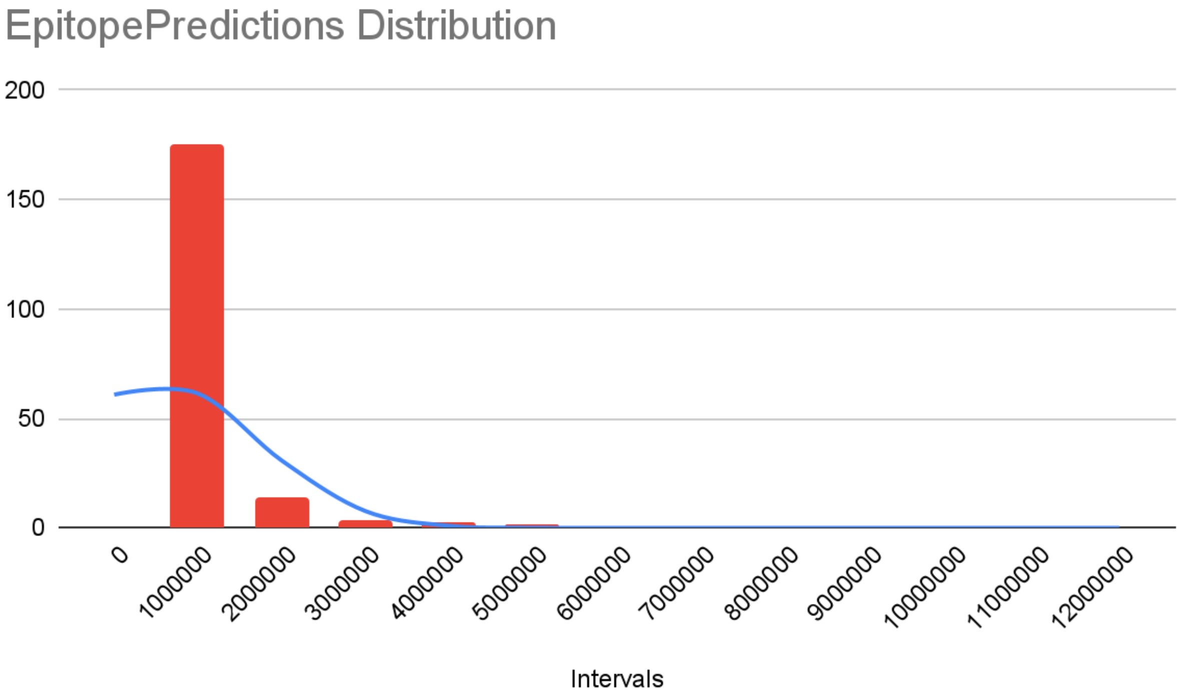
R-code

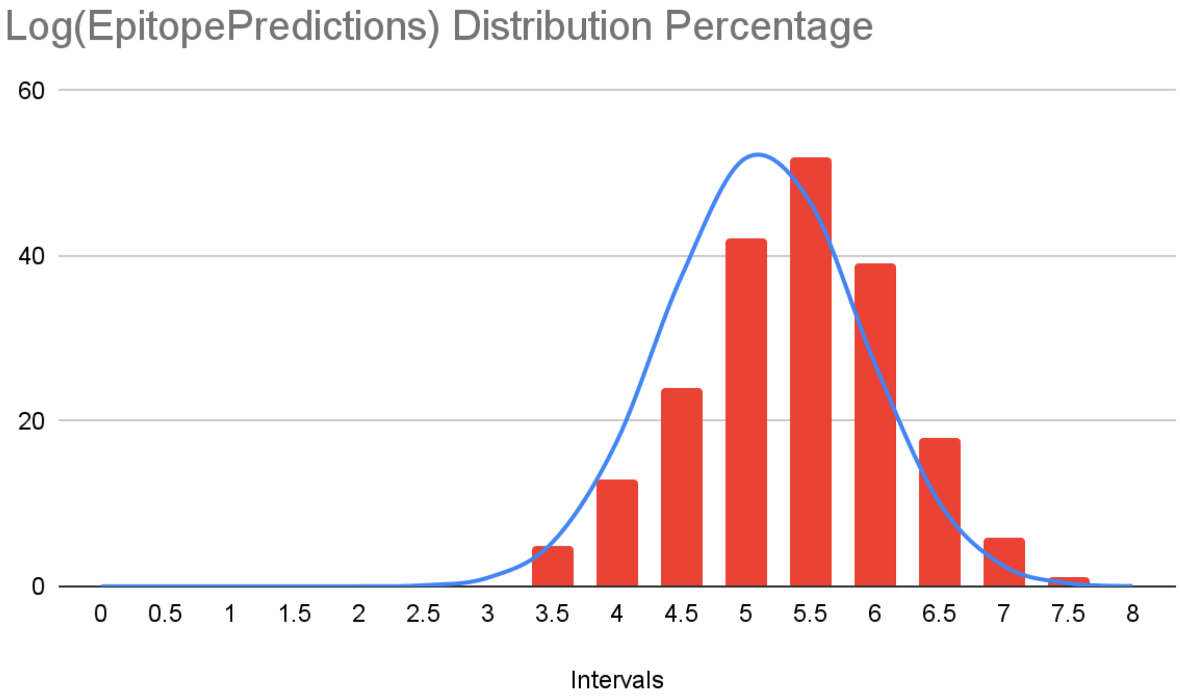
For the generation of the misbehaving IC_{50} values that were the motivation for this research 2 R scripts were used.
Those scripts can be found here: https://github.com/richelbilderbeek/ep_vs_mhcn.
They roughly work as follows:
the first script, https://github.com/richelbilderbeek/ep_vs_mhcn/blob/master/create_dataset.R, is used to generate random peptides (line 20 to 31) and after that the two programs MHCnuggets and EpitopePrediction are used to predict the ic50 values of the randomly generated peptides (line 33 to 41), these are stored in the file “ep_vs_mhcn.csv” (line 43).

Results

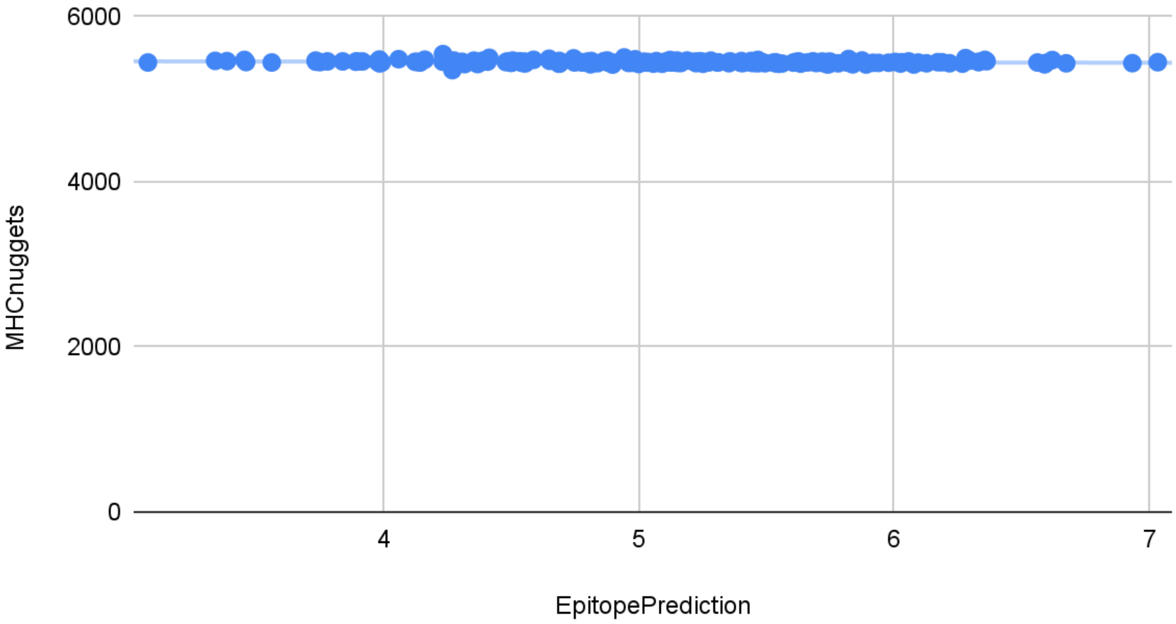

Correlation log(EpitopePredictions)
Pearson
Spearman
1.0
0.5
0.0
-0.5
Correlation (r)
HLA-A*01:01
HLA-A*02:01
HLA-A*03:01
HLA-A*24:02
HLA-A*26:01
HLA-B*07:02
HLA-B*08:01
HLA-B*15:01
HLA-B*18:01
HLA-B*27:05
HLA-B*39:01
HLA-B*40:02
HLA-B*58:01


Correlation
Pearson
Spearman
1.0
0.5
0.0
-0.5
Correlation (r)
HLA-A*01:01
HLA-A*02:01
HLA-A*03:01
HLA-A*24:02
HLA-A*26:01
HLA-B*07:02
HLA-B*08:01
HLA-B*15:01
HLA-B*18:01
HLA-B*27:05
HLA-B*39:01
HLA-B*40:02
HLA-B*58:01
Haplotype

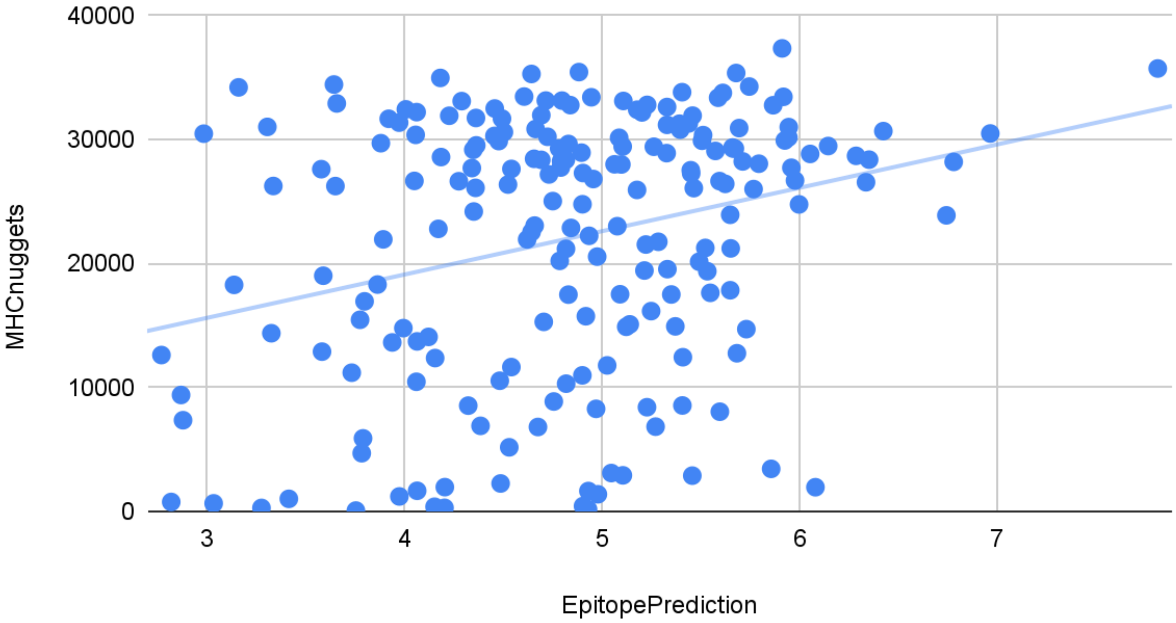

EpitopePredictions Distribution
200
150
100
50
0
0
1000000
2000000
3000000
4000000
5000000
6000000
7000000
8000000
9000000
10000000
11000000
12000000
Intervals


Log(EpitopePredictions) Distribution Percentage
60
40
20
0
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
5.5
6
6.5
7
7.5
8
Intervals

HLA-A*01:01



HLA-B*18:01



Haplotype	Pearson Correlation	Spearman Correlation	Pearson Log	Spearman Log
HLA-A*01:01	-0.074	-0.251	-0.182	-0.251
HLA-A*02:01	0.042	0.043	0.381	0.043
HLA-A*03:01	0.055	0.08	0.303	0.08
HLA-A*24:02	0.138	0.469	0.457	0.469
HLA-A*26:01	0.218	0.434	0.439	0.434
HLA-B*07:02	0.309	0.734	0.758	0.734
HLA-B08:01	0.329	0.762	0.815	0.762
HLA-B*15:01	0.227	0.812	0.821	0.812
HLA-B*18:01	0.119	0.221	0.287	0.221
HLA-B*27:05	0.252	0.73	0.769	0.73
HLA-B*39:01	0.156	0.623	0.662	0.623
HLA-B*40:02	0.171	0.556	0.578	0.556
HLA-B*58:01	0.31	0.785	0.816	0.785

Here we have the correlation-values as calculated using both Pearson's and Spearman's formulas. The data is split into two parts, one where we did not use EpitopePredictions' Log value and one where we did. The data with Log-values is labeled Log. Do note that Pearson Log and Spearman Log only indicate that the predicted IC50-values of EpitopePredictions are Log transformed, and not that the calculated correlations are Log transformed. The Pearson Correlation gave a value between -0.074 and 0.309 and the Spearman Correlation ranged from -0.251 to 0.812. For the Log-values, Pearson ranged from -0.182 to 0.821 and the Spearman correlation was between -0.251 and 0.812.

A notable exception is that the first haplotype has a negative correlation, contrary to all the others which have a positive correlation. This means that on average the graph has a slightly negative slope.

Another interesting observation is that the values calculated by Pearson's coefficient are consequently lower than those calculated by Spearman's coefficient, except for the logarithmic values, where the exact opposite is the case for some values.

Underneath the results we have all predicted values by EpitopePredictions for HLA-A*01:01 in a histogram. Additionally the normal distribution curve is added in blue. Additionally, the first 4 haplotypes show an unexpected graph. HLA-A*01:01 is used as an example. The graph has a horizontal line, indicating all MHCnuggets' predictions are very similar.

MHC	Average	Max Deviation
HLA-A*01:01	5449.2375	97.8475

In this table the average predicted IC50 by MHCnuggets for haplotype HLA-A*01:01 is shown. All predictions average at 5449.23375 and the prediction that deviated from this the most, only differed by 97.8475, which is within 98% of 5449.2375.

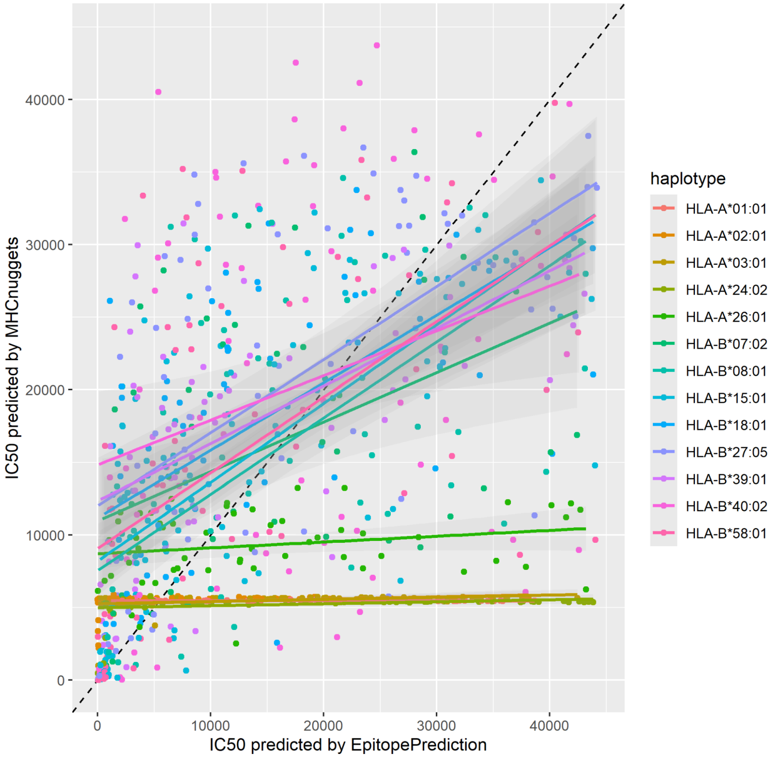
Conclusions

In this paper we discussed the workings of both MHC-I binding prediction programs EpitopePredictions and MHCnuggets. We used the Pearson- and Spearman correlation coefficients to determine whether or not there was a statistically significant linear or monotonic relation between the predicted IC50 values of both programs. When looking at the normal distribution curves, it is clear the EpitopePrediction needs to be log transformed. The log transformed values almost match the normal distribution, while the 'normal' values do not match a normal distribution. The first 5 haplotypes and HLA-B*18:01 have a Pearson correlation of less than 0.5, which is considered a moderate relation[13]. A correlation above 0.5 is considered strong.

As shown in the results section, the first 4 haplotypes have an unexpected graph where MHCnuggets' predictions are all very similar. We suspect this is where there is a mistake in MHCnuggets' prediction method, since it is unlikely that all 200 epitopes would have the same binding affinity. For HLA-A*01:01 the maximum deviation was only 2%. This anomaly is only found in the first 4 haplotypes. The other two (HLA-A*26:01, HLA-B*18:01) haplotypes do not show this specific aberration, but those values are a bit disordered. There is not a clear trend visible in the data points and they have a low correlation (0,218 and 0,469 for HLA-A*26:01, 0,119 and 0,221 for HLA-B*18:01)

All in all we think all other haplotypes, which have a Pearson correlation greater than 0.5, have a strong enough correlation for us to conclude that both programs predicted the IC50-value correctly. We conclude this, because it is highly unlikely that both programs would predict the same, wrong results. While both the Pearson- and Spearman correlation are very similar for all these haplotypes, we still use the Peason correlation coefficient as a determinant for whether or not the predicted values are trustworthy, because a linear correlation is to be expected when both programs should output the same results. And since we Log-transformed EpitopePredictions' graph we expect the correlation to be linear.

Figure 1



Dashed line: x = y. Blue line = fit to linear model, should ideally match the dashed line

It is also interesting to see that the correlations of different haplotypes appear to correlate with how close a trendline's slope in the right graph is to y=x (the dashed line in figure 1), though this does not apply to HLA-B*18:01, the calculated r values are very low, but it's trendline is fairly close to x=y. Which at first sight is pretty strange, because both the coefficients and the trendline describe some form of correlation This was to be expected as

that describes some form of correlation too. This same effect is also visible in other calculations done on this data, for example the relative values[I1] and the sorted values[I2], we do not know where this effect originates from.
Another interesting observation is that by eyeballing higher numbered haplotypes appear to have trendlines closer to x=y. We do not know where this effect originates from either.

As shown by the results, our hypothesis was partially incorrect. We suspected the highest prediction for EpitopePredictions should also be the highest prediction for MHCnuggets, for all haplotypes, but it turns out this is only the case for some.

Discussion

There are many points in this research that could be further investigated. For example a comparison with other similar papers, there are quite a few papers that did similar research on these programs that have found varying results. Because it could be true that our research is incorrect and the IC_{50} values are actually fairly similar in other cases. This could especially gain insight into MHCnuggets' sometimes unpredictable results. [5,6]
Research could also be done to find out why MHCnuggets' predictions for HLA-A*01:01, HLA-A*02:01, HLA-A*03:01 and HLA-A*24:02 are all very similar to each other, since this is only the case for these haplotypes, lack of training data, or a bug in the prediction method, could be the cause for this, but further research is required. Furthermore, research can be done to find out why both programs differ so much on HLA-A*26:01 and HLA-B*18:01, where MHCnuggets' results do differ. This would probably require more insight into the working of MHCnuggets.

As to the improvement of our research, a larger sample size of epitopes could benefit results. More haplotypes could also be tested. It would also be beneficial for our research to know more about the inner workings of MHCnuggets, but this is fairly hard as MHCnuggets is a neural network that taught itself how to predict the values.

Epilogue

Our thoughts

When Richel Bilderbeek offered us two projects to choose one from, we all agreed that we should choose the project that we did. After finishing the project we still think that this was the right choice. We found it very interesting to research the immune system and how it works, which is a complex but relevant topic. Research like this and further research into this topic can help speed up the development of vaccines, which is very important during a pandemic.

What was also fun and interesting, was the statistical research we did, mainly in google sheets(excel). Comparing data-tables, values and graphs to see if there were correlations between them, and trying to find out what methods worked best. There is also something that we didn’t like about the project, which is that in the end we had to rush it a bit. Due to circumstances like the test week being moved, and ofcourse the pandemic, we had less time to work on the project than normally.

Acknowledgements

For this project we would like to thank Richel Bilderbeek, for providing us with the project, valuable resources and data, and guiding us through it. We would also like to thank our teachers Michel Romeijn and Willy Reinalda for helping us when it was needed.

Finally we would like to thank everyone who contributed to the articles we used in our references.

Definitions

Epitope: the part of an antigen molecule to which an antibody attaches itself. An epitope is a specific protein on the surface of an antigen and is used for recognition of cells. [28]

IC_{50}: a quantitative measure that indicates how much of a particular inhibitory substance (e.g. drug) is needed to inhibit, in vitro, a given biological process or biological component by 50%. The biological component could be an enzyme, cell, cell receptor or microorganism. For our research this means a lower IC_{50} corresponds to an epitope being more likely to be presented. [15]

Haplotype: a group of genes within an organism that was inherited together from a single parent. A different haplotype can correspond to a different MHC I-complex, which in itself means that different epitopes are more or less likely to be presented to the immune system. [27]

References

[1] Peters, Bjoern, and Alessandro Sette. "Generating quantitative models describing the sequence specificity of biological processes with the stabilized matrix method." *BMC bioinformatics* 6.1 (2005): 1-9.

[2] Kim, Yohan, et al. "Derivation of an amino acid similarity matrix for peptide: MHC binding and its application as a Bayesian prior." *BMC bioinformatics* 10.1 (2009): 1-11.

[3] Sanchez-Trincado, Jose L., Marta Gomez-Perosanz, and Pedro A. Reche. "Fundamentals and methods for T-and B-cell epitope prediction." *Journal of immunology research* 2017 (2017).

[4] Trolle, Thomas, et al. "The length distribution of class I–restricted T cell epitopes is determined by both peptide supply and MHC allele–specific binding preference." *The Journal of Immunology* 196.4 (2016): 1480-1487.

[5] Janeway, Charles A, and Jr. “The major histocompatibility complex and its functions.” *Immunobiology: The Immune System in Health and Disease. 5th edition.*, U.S. National Library of Medicine (1970).

[6] Bhattacharya, Rohit, et al. "Evaluation of machine learning methods to predict peptide binding to MHC Class I proteins." *BioRxiv* (2017): 154757.

[7] Kim, Y., Sidney, J., Pinilla, C. *et al.* Derivation of an amino acid similarity matrix for peptide:MHC binding and its application as a Bayesian prior. *BMC Bioinformatics* 10, 394 (2009). https://doi.org/10.1186/1471-2105-10-394

[8] Various calculators for correlation coefficients
https://www.socscistatistics.com/

[9] MHC class I
https://en.wikipedia.org/wiki/MHC_class_I

[10] hydrophobicity
https://en.wikipedia.org/wiki/Hydrophobe

[11] Epitope Predictions
https://github.com/jtextor/epitope-prediction

[12] Initial research done by Richel Bilderbeek
richelbilderbeek/ep_vs_mhcn

[13] Interpretation of Pearson’s coefficient
https://www.questionpro.com/blog/pearson-correlation-coefficient/

[14] Viral Envelope
https://en.wikipedia.org/wiki/Viral_envelope

[15] IC_{50}
https://en.wikipedia.org/wiki/IC50

[16] Our biology textbook
https://www.noordhoff.nl/voortgezet-onderwijs/biologie/nectar

[17] Spearman’s correlation coefficient
https://www.statstutor.ac.uk/resources/uploaded/spearmans.pdf

[18] Amino acids
https://www.technologynetworks.com/applied-sciences/articles/essential-amino-acids-chart-abbreviations-and-structure-324357

[19] Pearson tabel
https://libguides.library.kent.edu/spss/pearsoncorr

[20] Research done by Joshua van Waardenberg
richelbilderbeek/meesterproef_joshua

[22] Spearman’s vs Pearson’s correlation coefficient
https://statisticsbyjim.com/basics/spearmans-correlation/

[23] MHC I binding cleft
https://www.sciencedirect.com/topics/immunology-and-microbiology/peptide-binding-cleft

[24] Prior Probability
https://en.wikipedia.org/wiki/Prior_probability

[25] amino-acid residue
https://www.ebi.ac.uk/chebi/searchId.do?chebiId=33708

[26] LSTM(long short term memory)
https://www.researchgate.net/publication/13853244_Long_Short-term_Memory

[27] Shao, Xiaoshan M., et al. "High-throughput prediction of MHC Class I and Class II neoantigens with MHCnuggets." CANCER IMMUNOLOGY RESEARCH. Vol. 8. No. 3. 615 CHESTNUT ST, 17TH FLOOR, PHILADELPHIA, PA 19106-4404 USA: AMER ASSOC CANCER RESEARCH, 2020.

[27] Haplotype
https://en.wikipedia.org/wiki/Haplotype

[28] Epitope
https://en.wikipedia.org/wiki/Epitope

Github repository:
https://github.com/GitOwenM/Technasium2021
Google sheets:
https://docs.google.com/spreadsheets/d/1FXp76K3rlhghKClpzPvp1BTIyHo1Klz_ZUuFSbvLDa8/edit?usp=sharing

Images:

[I1] Relative values
https://raw.githubusercontent.com/richelbilderbeek/ep_vs_mhcn/master/ep_vs_mhcn_perc.png
[I2] Sorted values
https://raw.githubusercontent.com/richelbilderbeek/meesterproef_joshua/master/ep_vs_mhcn_sort.png
[I3] Visualisation of the neural network
https://cancerimmunolres.aacrjournals.org/content/8/3/396.long
[I4] MHCnuggets
https://cancerimmunolres.aacrjournals.org/content/8/3/396.long
[I5] LSTM
https://www.researchgate.net/publication/13853244_Long_Short-term_Memory