Evolving Training Sets for Peptide Discrimination via Evolutionary Algorithms

Loek Gerrits (s1032343), Evangelos Spithas (s1125593), Bart van Nimwegen

June 13, 2025

1 Introduction

1.1 Background

In human bodies, the immune system is responsible for detecting pathogens and exterminating them. One of the tools at its disposal, is the T cells. T cells grow in the thymus, and each one is responsible for identifying and attacking specific cells. However, that means that it is possible for healthy cells to be attacked as well. In order to mitigate this, human cells are presented to the T cells, and the T cells that attack them are eliminated. This process is called Negative Selection (NS). In practice, the amount of possible cells is very large, and as a result only a sub-set of self peptides are presented in the T cells.

Artificial Immune Systems (AIS) are systems that draw inspiration from the human immune system, similarly to how Neural Networks (NNs) are inspired by the human nervous system. A big challenge in using the NS algorithm is selecting an efficitive sub-set of peptides to train on. Selecting the optimal subset is important because it direcly impacts the algorithm's ability to differentiate between healthy and harmful peptides. Therefore being able to find the optimal subset to use for the NS algorithm is crucial to improve the accuracy of the NS algorithm.

In this project, we would like to explore how we can find this optimal subsets to train an NS algorithm for peptide selection. We will do this using 3 different methods, randomly sampling a subset, using a greedy algorithm and an Evolutionary Algorithm (EA) to produce them. Afterwards, we will train the negative selection algorithm with each one of them, and evaluate its performance against different sets of harmful peptides, such as HIV and ebola cells.

1.2 Relevance

1.3 Problem description

1.4 Related work

[1]

2 Methodology

2.1 Optimizers

In order to create more effective training dataset of self peptides for negative selection, we implemented two optimizers: 1) a greedy algorithm and 2) the evolutionary algorithm. These strategies try to select subsets that maximize the information of the training data.

2.1.1 Greedy Algorithm

The greedy algorithm optimizes ...

Because the greedy algorithm has a high complexity, computing the fully optimized dataset using all human peptides and all possible motifs was infeasible. Therefore, this study limited the number of peptides and motifs.

2.1.2 Evolutionary Algorithm

AA composition

AA frequency

Exchangeability

2.2 Negative Selection

In order to run the negative selection experiments we will use the jar that is provided by `https://johannes-textor.name/negativeselection.html`. The training of the algorithm will take place with the three training sets as described in section 2.1. All elements of each dataset have a fixed length of 6, and we will evaluate the effectiveness of the NS algorithm by experimenting with a variable length of contiguous selectors, ranging between 1 and 6.

For each peptide in the test datasets, the NS algorithm can return either the number of patterns in the repertoire that match it, or the normalised value $log_2(1 + x)$, where x is the number of matching patterns. We will use this normalised value here, as the number of selectors can be unwieldy to work with. Afterwards, we can classify each peptide as anomalous or self peptide, if its score exceeds the value r.

2.3 Experimental design

We set the following experimental setup to test our methods. We first generated three subsets of the english dataset from the text of Mobby Dick, with the methods described in section 2.1, which we will use as the self set of the negative selection algorithm. The datasets have a size 3574. This was the size of the dataset returned by the greedy algorithm and was then used as the target size of the datasets of the other methods.

As test sets, we used an english dataset with 2000 english tokens, which were obtained by random excerpts from The Bible (`https://www.wordproject.org/bibles/index.htm`). We also repeated the same method to obtain test datasets in the following languages: Hiligaynon, Latin, Middle English, Plautdietsch, Tagalog and Xhosa. The NS algorithm was run for each language, and after classifying the data, we estimated the receiver operating characteristic curve (AUC) to quantify how well our estimated datasets can optimise the NS algorithm.

Lastly, we repeated the same experiments but with actual peptide data. For that we used a subset of human peptides with size 126, as well peptides of the following diseases: HIV, hepatitis B, and ebola, as obtained by `https://github.com/ingewortel/negative-selection-2020`

3 Results

3.1 Language Discrimination

In figures 1, 2, 3, we see the ROCs and AUC scores of the NS algorithm, for different r values for discrimination between english and each other language. First of all, we can observe that for $r = 1$, the algorithm performs very poorly. That is to be expected, as for that value we look for contiguous patterns of length 1, which are very small to carry any important information. As the value of r increases, so does the AUC, until it reaches $r = 4$, as it then starts to decrease again, very steeply from 5 to 6, which potentially shows that we have a lot of false positives. The best results are achieved for either $r = 3$ or $r = 4$.

If we now compare the performance for each language, we can observe that for Middle English, for all the different self datasets, the results are poor. This happens because Middle English is very similar to modern English, without enough distinctions for the NSA to pick up. The more different the language gets from English, the better results we have, with Xhosa, having the best AUC values.

Lastly, we shall evaluate the performance between each different self dataset we have created. Overall, all the datasets achieve similar results, without big differences. This means that they all carry meaningful information to discriminate the languages. If we look closely, we will see that when using the dataset generated from the evolutionary algorithm, we achieve the best performance in Hiligaynon, Latin, Tagalog and Xhosa. The random self dataset achieves the best results for Plautdietsch, while the greedy self dataset performs the best with Middle English. As we can see, we do not achieve significant gains from the different datasets, however the evolutionary datasets seems promising as it is always the best or second best.

3.2 Foreign Peptide Detection

Next we will have a look at the results for foreign peptide detection. If figures 4, 5, 6, we see the ROCs and AUC scores of the NS algorithm, for different r values for discrimination between self peptides and HIV, Ebola and Hepatitis-B peptides. As we can observe, the results are rather poor for all the differently created self datasets. This indicates that the discrimination of foreign and self peptides is a rather difficult one, most probably because of the similarities between peptides, unlike the more complex structure of languages. It is worth noting however, that we still observe a small bump in the AUC scores for r values of 3 and 4, like before.

4 Discussion

In the Results section we observed that the performance for the peptide detection was unfortunately very poor.

5 Conclusion

The end.

References

[1] Inge MN Wortel et al. "Is T cell negative selection a learning algorithm?" In: *Cells* 9.3 (2020), p. 690.

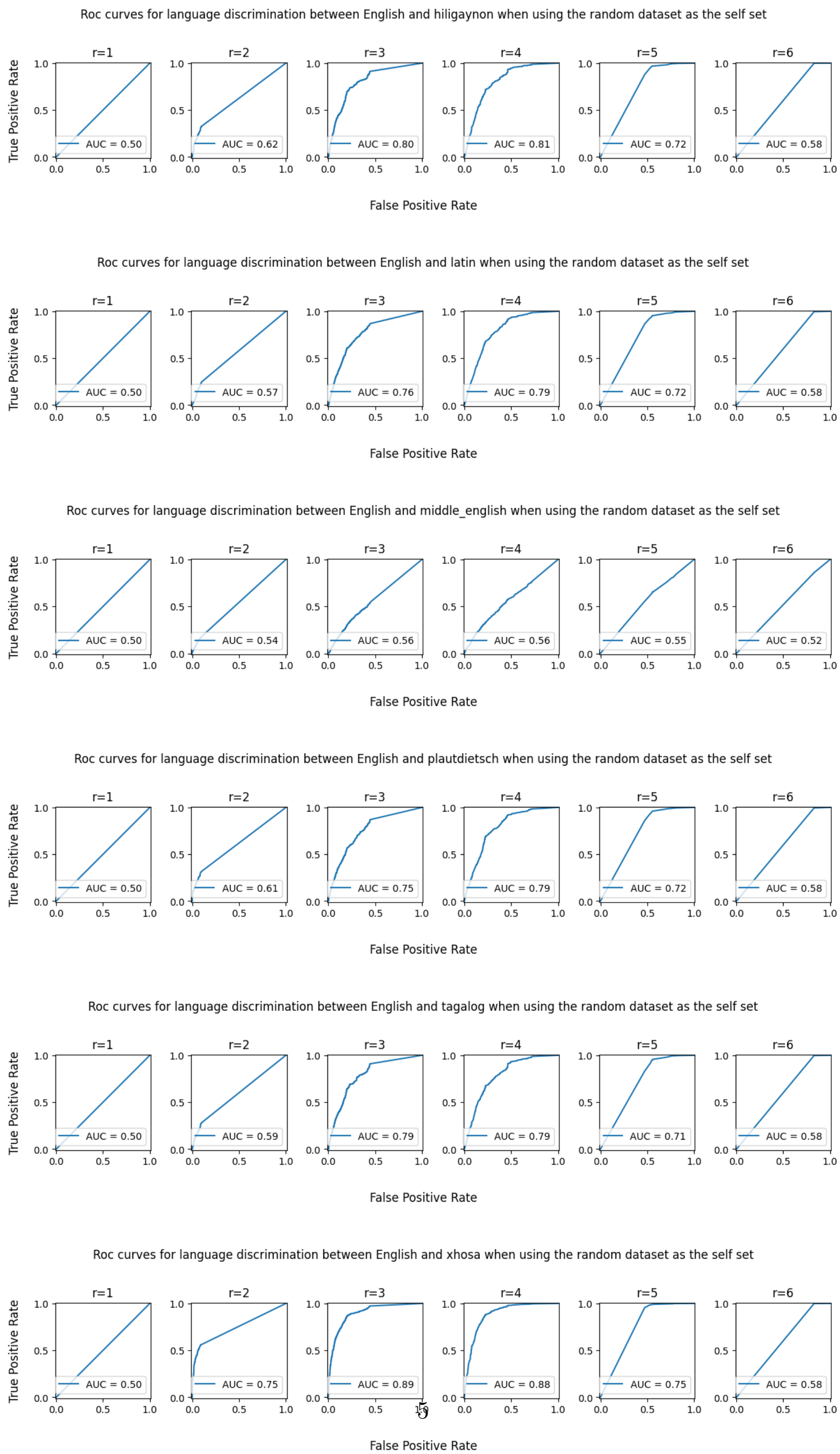

Figure 1: ROC curves for language Discrimination between English and each other language, when using the random dataset for the self set

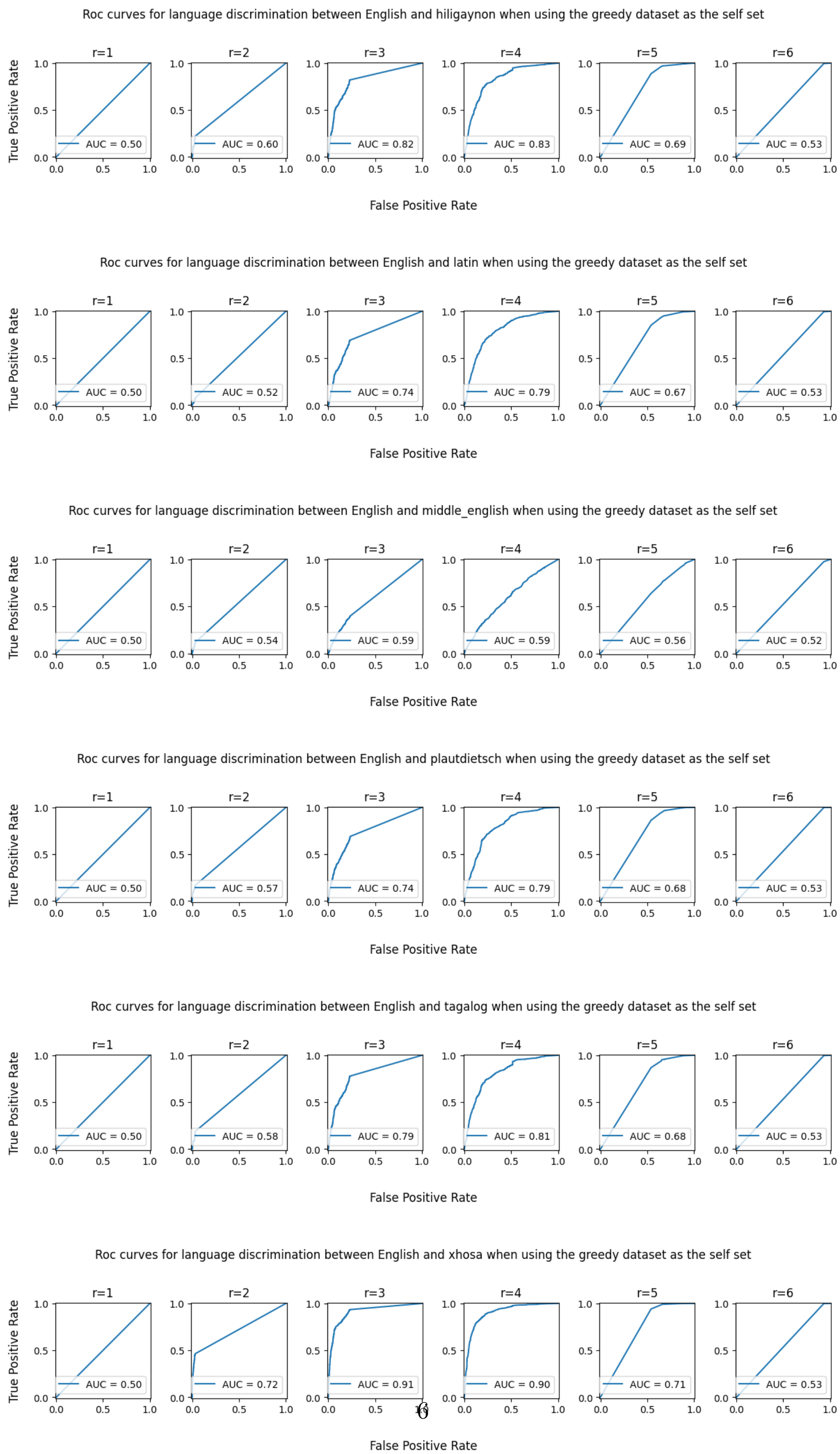

Figure 2: ROC curves for language Discrimination between English and each other language, when using the greedy dataset for the self set

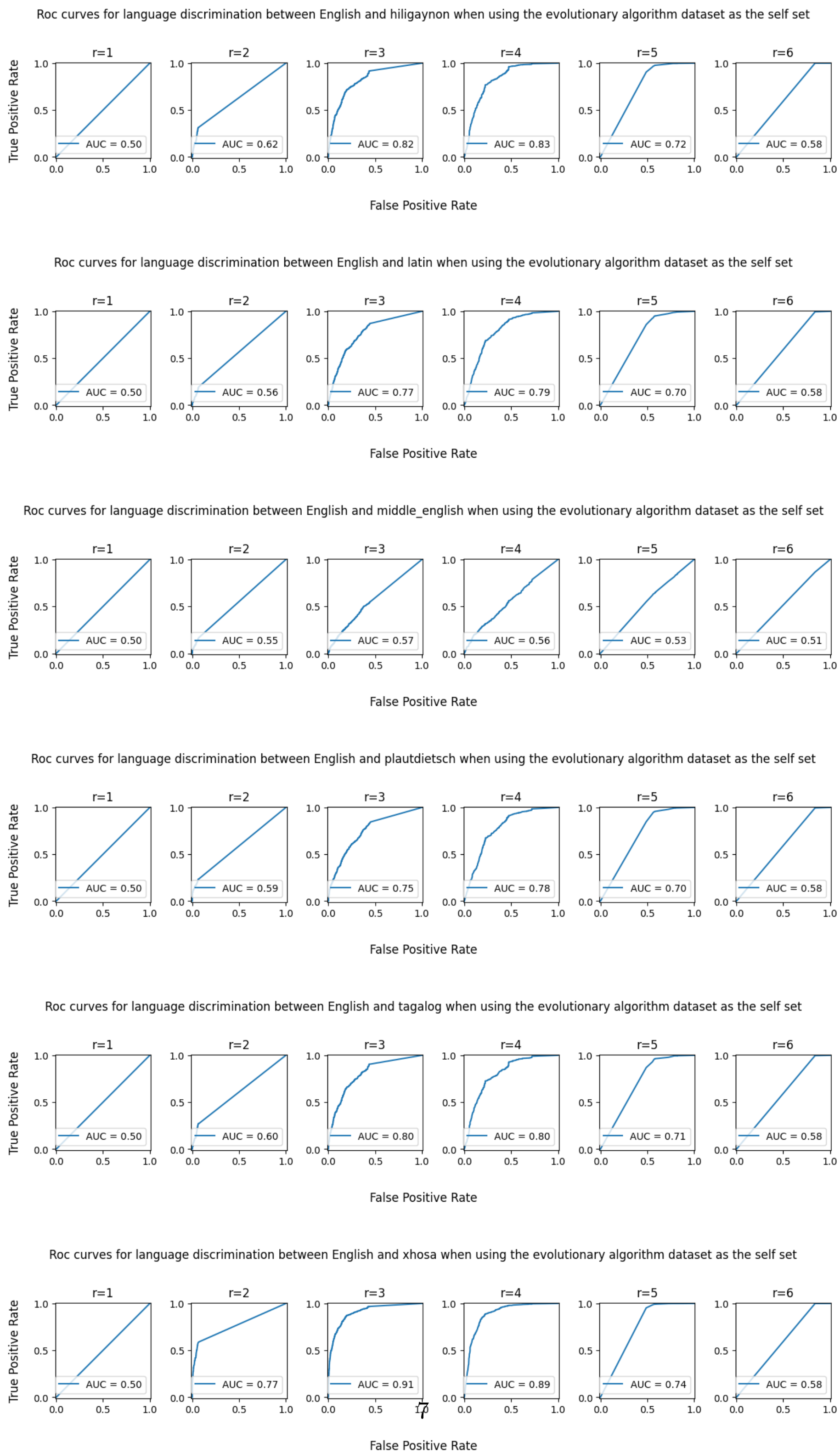

Figure 3: ROC curves for language Discrimination between English and each other language, when using the evolutionary algorithm dataset for the self set

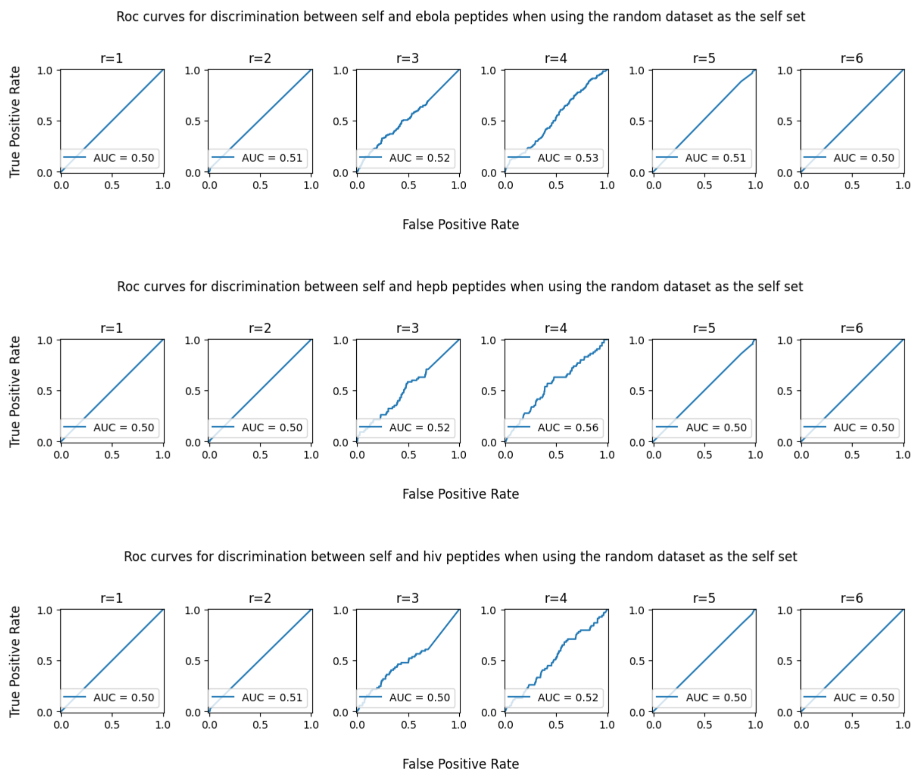

Figure 4: ROC curves for peptide detection between self and foreign, when using the random dataset for the self set

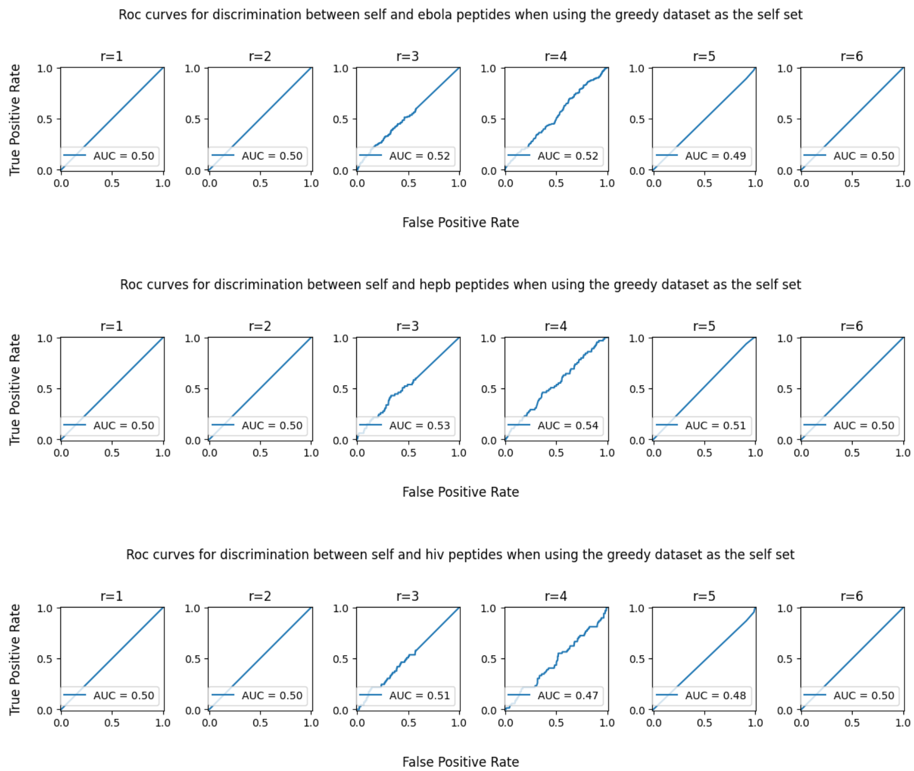

Figure 5: ROC curves for peptide detection between self and foreign, when using the greedy dataset for the self set

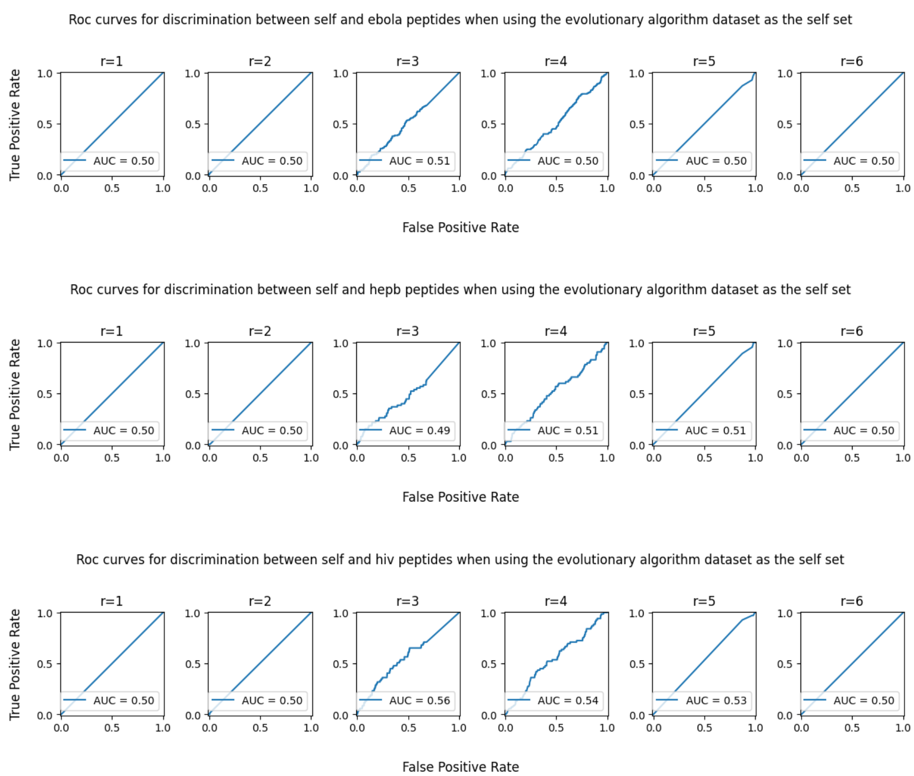

Figure 6: ROC curves for peptide detection between self and foreign, when using the evolutionary algorithm dataset for the self set