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.

1.2 Relevance

Negative Selection algorithms have a number of applications in both biological algorithms and computer algorithms. Understanding Negative selection for biology can improve our understanding of, for example how the immune system handles negative cells, which can prevent certain diseases. For computer algorithms, 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, Negative Selection (NS) Algorithms are also part of these AIS, and are often used for anomaly detection or classification.

For both the biological algorithms and computer algorithms, the effectiveness of the NS algorithm heavily depends on the quality of the training data, especially the subset of self-peptides that is used to eliminate T-cells that could otherwise cause harm . Since using all the self-peptides is not really possible, as the number of self-peptides is too big it would be too expensive to process all of them.

Therefore being able to find the optimal subset to use for the NS algorithm is crucial to improve the accuracy of the NS algorithm.

1.3 Problem description

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. Additionally, we will be performing the Negative Selection on different language such as Xhosa and Tagalog and check if we get better results for using different sets of peptides, generated by the three algorithms.

1.4 Related work

This project is an extension to the paper by Wortel et al [2], They developed a greedy algorithm to approximate an optimal training set for NSA training, and found that the "artificial immune system" performed better in terms of generalization and discrimination using such optimized training sets. We want to verify those findings by comparing this greedy algorithm using a randomly selected training set and a training set created by an evolutionary algorithm.

In the paper by Wilde et al. [1] they propose a solution on how evolutionary algorithms can have a positive impact on optimizing training sets for classification problems. We hope to include and apply those findings when we are implementing our evolutionary algorithm to create an optimized training set.

2 Methodology

2.1 Datasets

This study first tried to deal with the easier problem of discriminating two different languages, and later on we consider the harder problem of distinguishing human peptides from foreign ones. Therefore, the datasets used in our experiments consist of two parts: 1) languages and 2) peptides. Each consists of a training set containing all the self peptides for negative selection, and test datasets containing foreign peptides and one with self peptides. In all our experiments, we set the length of short strings and peptides (from now on, for the sake of brevity, we will also refer to the strings in the language datasets as peptides, as they are functionally not different) to n to $n = 6$.

For the language datasets, we first defined a common alphabet of size of 27 which includes an additional '_' to signify spaces in the text and the 26 letters of the English alphabet. All texts were preprocessed such that every character not in the alphabet was replaced with a $'\$', and, as an extra pass, there could only be a single '\$'$ back-to-back. After that, we generated a training dataset ($n = 189718$) and a test set of strings of a length of 6 based on the novel 'Moby Dick' by Herman Melville. Then, test sets of foreign peptides were generated based on the first

few chapters of The Ghospel of Matthew from the Bible for the following languages: Hiligaynon, Latin, Middle-English, Plautdietsch, Tagalog and Xhosa. The test sets were all 2000 peptides long.

For the experiments with peptides, we used the various peptide datasets that were used by Wortel et al. [2] in their experiments found in their GitHub (`https://github.com/ingewortel/negative-selection-2020`). From this, we used the human peptides as the training ($n = 262000$) and test sets of self peptides ($n = 1216$), and peptides of the following diseases were defined as foreign peptides and used as test sets of various sizes: HIV, hepatitis B, and ebola. These peptide datasets used an alphabet of amino acids of length 20 (ACDEFGHIKLMNPQRSTVWY) as opposed to 28 from the languages part.

2.2 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.

2.2.1 Greedy Algorithm

The greedy algorithm iteratively builds a data set by greedily selecting a) the peptide that matches with the greatest umber of motifs and b) removing its matching motifs from the motifs set until there are no motifs left. A peptide matches with a motif when the number of consecutive AAs at their respective positions is equal to or greater than threshold t. For both parts, we used $t = 3$ and generated all possible motif combination with length n and the current alphabet, resulting in 27^6 motifs for the language part and 20^6 motifs for the peptide part.

Because the greedy algorithm has a high complexity (in every iteration, each peptide needs to be compared to all current motifs), computing the fully optimized dataset using all human peptides and all possible motifs was infeasible. Therefore, this study randomly sampled only 1% of all motifs to reduce the runtime of the experiments, and 4% of the self peptides. The greedily optimized dataset for each part also determined the size of the other generated datasets (3574 for languages, and 6125 for peptides). We did this because we wanted the comparisons between the other (random and evolved) resulting datasets to be consistent.

2.2.2 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.

2.2.3 Evolutionary Algorithm

AA composition

AA frequency

Exchangeability

2.3 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 2.2. 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.

2.4 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 2.2, 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 1216, 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 Receiver Operating Characteristic (ROC) curves 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. In figures 4, 5, 6, we see the ROC curves and AUC scores of the NS algorithm, for different r values for discrimination between self peptides and Ebola, Hepatitis-B and HIV 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 problem, 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. This stems from the fact that the foreign

peptides have patterns too similar to the human peptides. On the other hand, in the languages problem we saw that the NS algorithm performed relatively well, except Middle English, which are also very similar to modern English.

In the future we can explore improvements in the greedy and evolutionary algorithms we used to generate the optimised self peptide datasets. With regard to the evolutionary algorithm, we can experiment with different weights for the features we used to define the fitness function. Potential future work can entail exploring more languages to have a more diverse set of data, as well as pondering on more advanced problems, such as distinction between human and LLM generated text.

5 Conclusion

References

[1] Henry Wilde, Vincent Knight, and Jonathan Gillard. "Evolutionary dataset optimisation: learning algorithm quality through evolution". In: *Applied Intelligence* 50.4 (2020), pp. 1172–1191.

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

A Figures

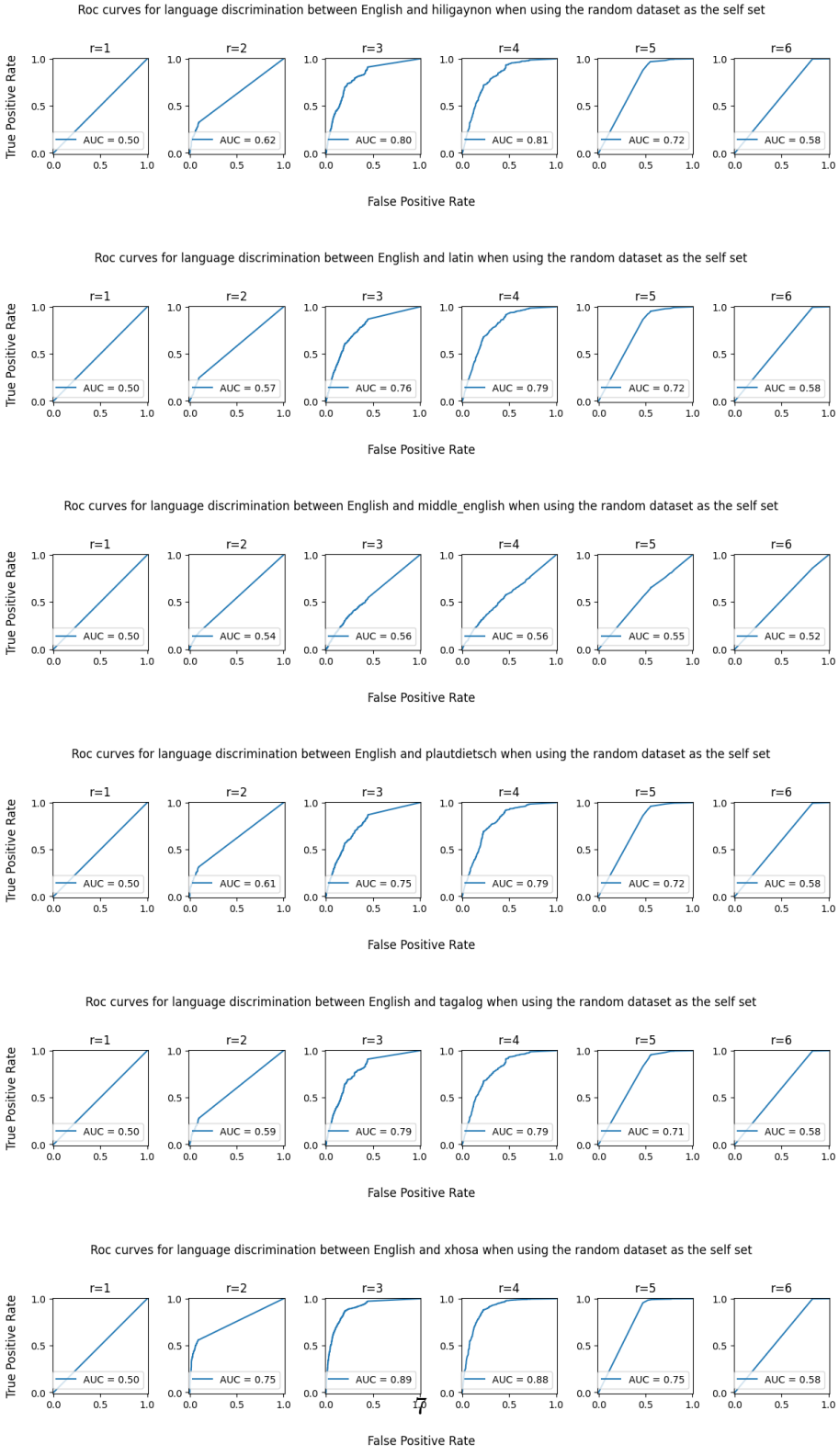

Figure 1: ROC curves for language Discrimination between English and each other language, when using the random dataset for the self set

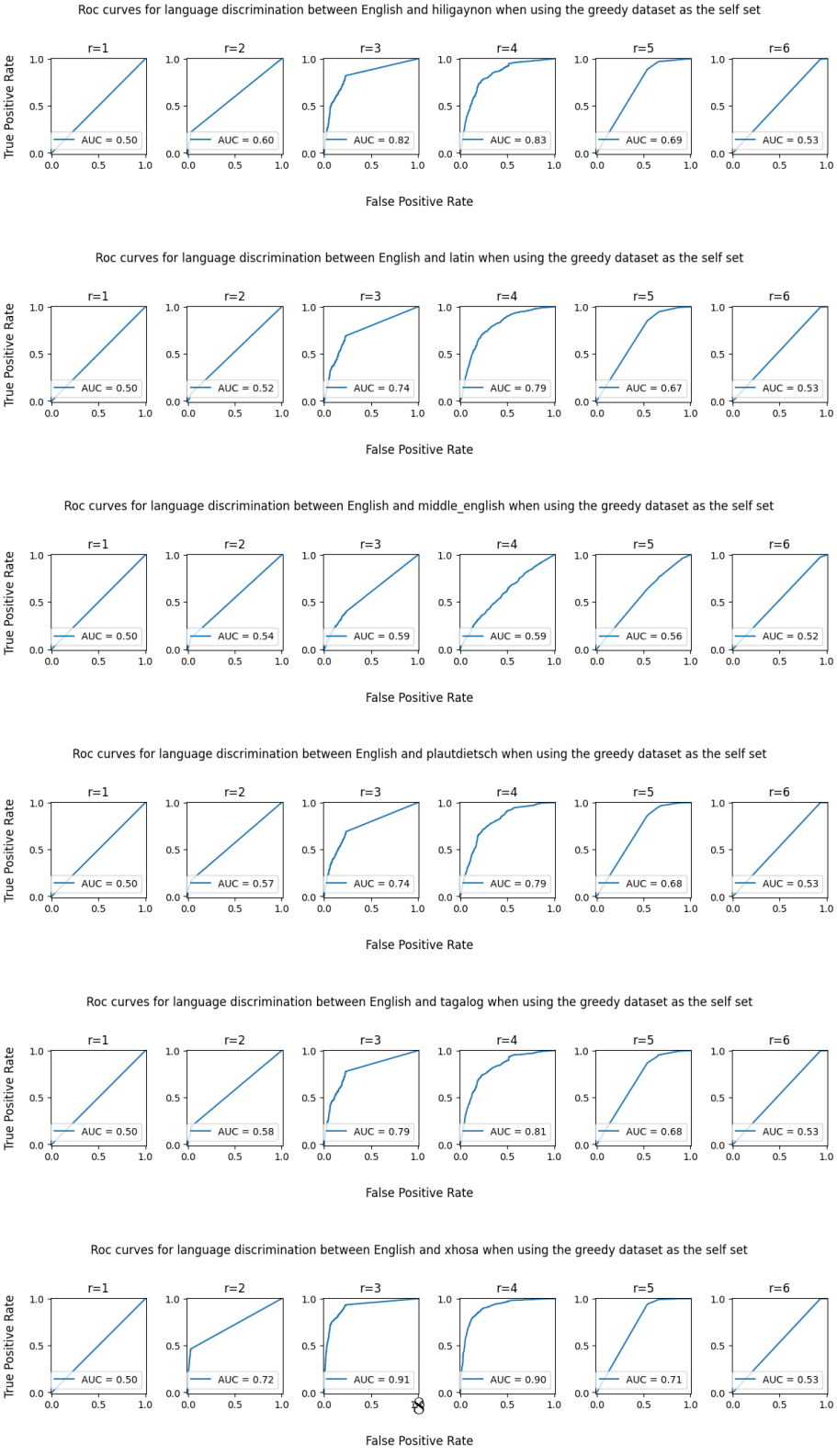

Figure 2: ROC curves for language Discrimination between English and each other language, when using the greedy dataset for the self set

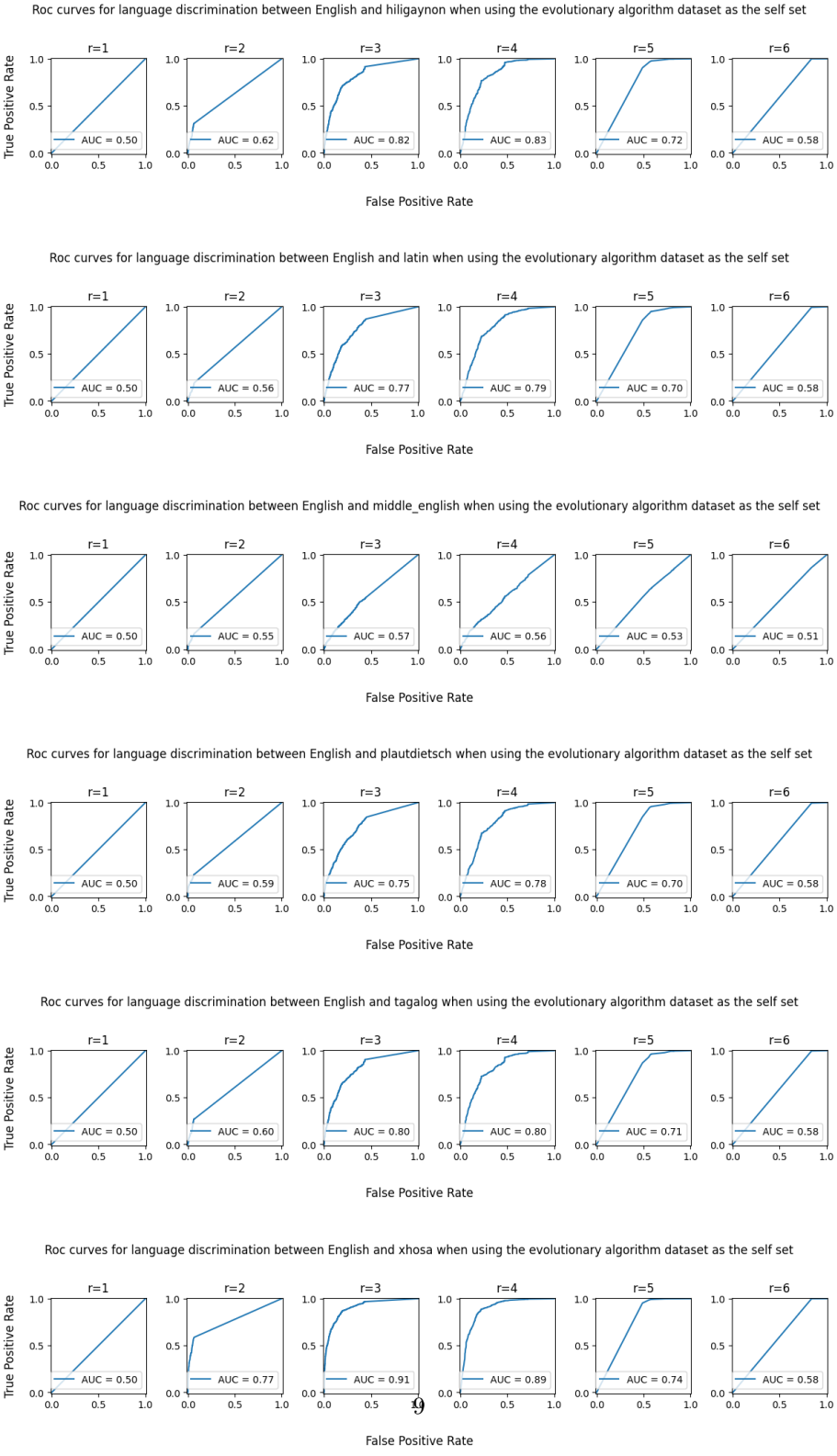

Figure 3: ROC curves for language Discrimination between English and each other language, when using the evolutionary algorithm dataset for the self set

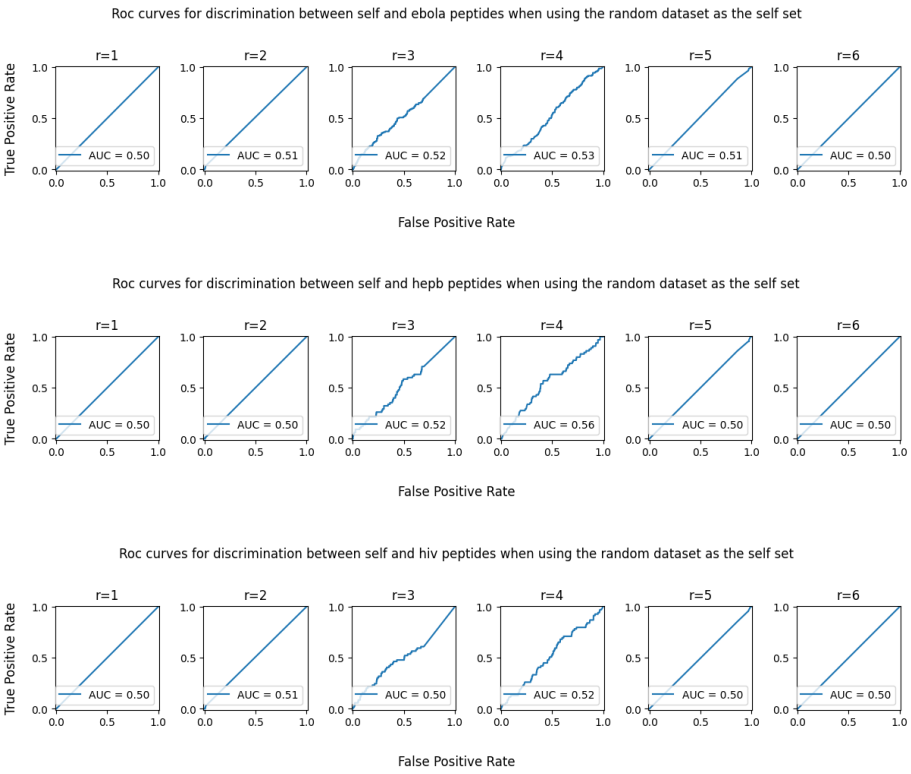

Figure 4: ROC curves for peptide detection between self and foreign, when using the random dataset for the self set

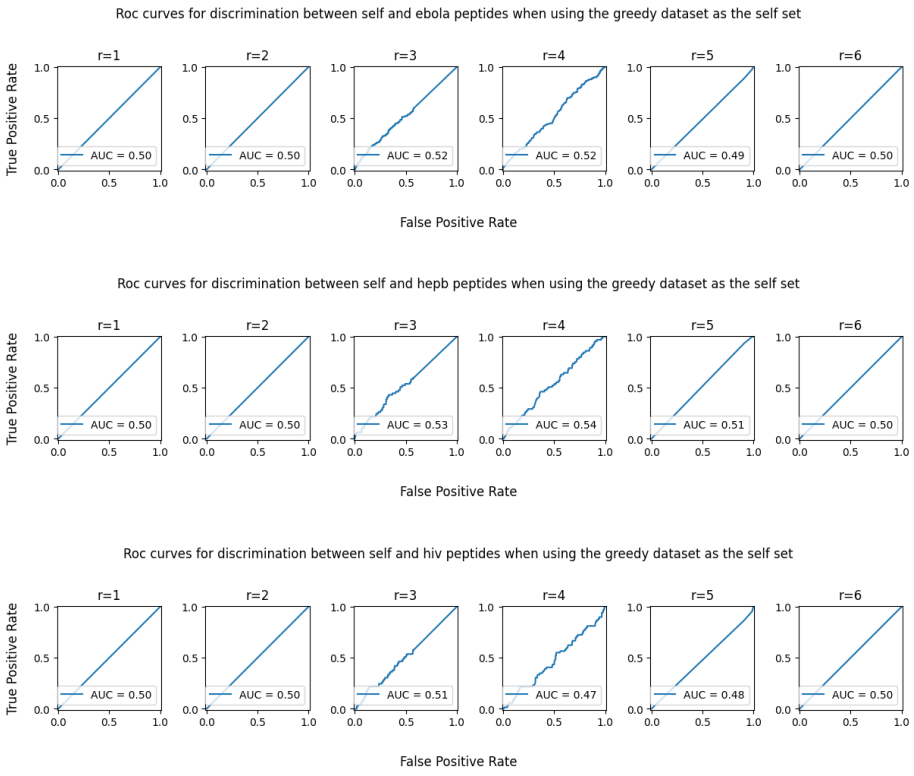

Figure 5: ROC curves for peptide detection between self and foreign, when using the greedy dataset for the self set

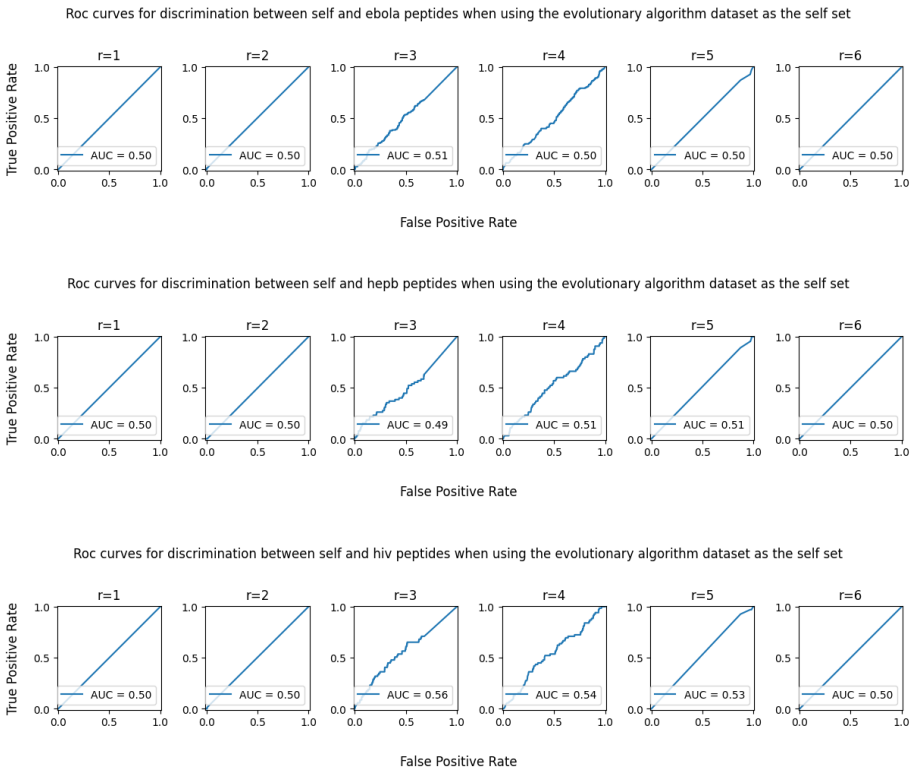

Figure 6: ROC curves for peptide detection between self and foreign, when using the evolutionary algorithm dataset for the self set