Mini-project Report

TengYao Tu, Wei Zeng, Kun Zhao, Zhenyu Zhang

***Abstract*—Researching the specificity of TCR contributes to the development of immunotherapy and provides new opportunities and strategies for personalized cancer immunotherapy. Therefore, we established a TCR generative specificity detection framework consisting of TCR classifier and specificity classifier based on random forest, aiming to efficiently screen out TCRs and corresponding antigens. We used the k-fold validation method to compare the performance of our model with ordinary deep learning methods. The result proves that adding a classifier to the model based on the random forest algorithm is very effective, and our model generally outperforms ordinary deep learning methods. Furthermore, we put forward feasible optimization suggestions for the shortcomings and challenges of our model found during model implementation.**

I. Introduction

Cancer is the leading cause of death worldwide. In 2022, there were an estimated 20 million new cancer cases and 9.7 million deaths (World Health Organization: WHO, 2024). Although traditional methods such as surgery and chemotherapy are effective in treating cancer to a certain extent, they are often not effective in preventing the metastatic spread of the disease through disseminated tumor cells (Schuster et al., 2006). With the deepening research on the immune system, immunotherapy that can harness the immune system to fight cancer has now firmly established itself as a novel pillar of cancer care (Esfahani et al., 2020).

The specificity of T cell receptor (TCR) plays a crucial role in immunotherapy. As an important class of lymphocytes in the immune system, T cells can recognize and bind to antigen peptides from the major histocompatibility complex (MHC) through the TCR on the surface to trigger an immune response to recognize and attack pathogens and abnormal cells. Researching the specificity of TCR contributes to the development of immunotherapy, but the various experimental methods used to identify the interactions between TCRs and peptides presented by MHC molecules (pMHCs) have limitations such as time-consuming, costly, or technically demanding (Zhao et al., 2023). Therefore, algorithms that can accurately identify and predict TCR specificity need to be developed.

We establish a TCR generative specificity detection framework, which enables efficient screening of TCRs and specific antigens using distance and machine learning algorithms. Furthermore, the performance of this model is compared with some baseline models, which demonstrates a further improvement in the accuracy and confidence of the model in predicting TCR specificity.

II. Literature Review

We researched various computational models developed for TCR specificity prediction, which can be broadly divided into three categories according to the working principle.

The first class of models predicts the specificity of TCR based on three-dimensional structural information of TCR and pMHC. These models perform well when there is a need for detailed knowledge and high-resolution prediction of binding patterns and specificity. However, the structural basis of TCR activation is poorly understood (Mariuzza et al., 2020), which reduces the accuracy and feasibility of the model.

The second class of models is based on the sequence information of the TCR and pMHC. The development of high-throughput sequencing methods and single-cell RNA sequencing technologies has enabled efficient and rapid derivation of large numbers of TCR sequences from donor samples, which are collated into rich datasets (Shugay et al., 2017). It promotes the application of deep learning methods in the modeling of TCR-pMHC interactions. However, these models only utilize the sequence information of the TCR and ignore structural information. Therefore, the accuracy of this class of model may not be ideal if structural features need to be considered in the prediction.

The third class of models combines the characteristics of the first two classes of models, which can make full use of structure and sequence information to improve accuracy and reliability. In other words, it also has higher requirements for the variety and quantity of data. Therefore, these models may cause high computational costs and training difficulties if computing resources and training data are limited.

III. Methodology

The framework of our model is depicted in Figure 1. The model includes two key components: (1) TCR classifier which is used to calculate the edit distance matrix of TRBs between the target TCR and other TCRs in the training database and select the antigen of the closest training data as the target antigen. (2) Specificity Classifier Based on Random Forest. First, the selected target antigen and CDR3 are encoded by N-Gram respectively. Then the encoded CDR3 and antigen sequences are used as feature input, and the labels are used as a model output to build a random forest model (RF) which finally produces prediction results. In the following section, we will provide detailed explanations of the model.

IV. Descriptive Statistics and Data Preprocessing

Based on the detailed explanation of VDJDB[1], we have summarized the meanings of each attribute. Due to the detailed explanations in the VDJDB documentation, the article will not repeat the meanings of each attribute. For the task of TCR specificity testing, the following 12 attributes are considered valuable: complex.id, cdr3, gene, v.segm, j.segm, antigen.epitope, antigen.gene,species, vdjdb.score, mhc.a, mhc.b, mhc.class.

We first need to detect missing values in the database, as shown in Figure 2:

Index	Name	Missing values
0	complex.id	0
1	cdr3	0
2	gene	0
3	v.segm	0
4	j.segm	0
5	antigen.epitope	0
6	antigen.gene	0
7	species	0
8	vdjdb.score	0
9	mhc.a	0
10	mhc.b	0
11	mhc.class	0

Fig. 1. Missing value detection

From Figure 2, it can be seen that there are no missing values in the database for the attributes we need. Next, we conduct Spearman correlation analysis on these 12 attributes, and the results of the correlation analysis can help us screen for attributes with collinearity, thereby reducing the number of data attributes that the model needs to use. Our Spearman correlation analysis results are shown in Figure 3:

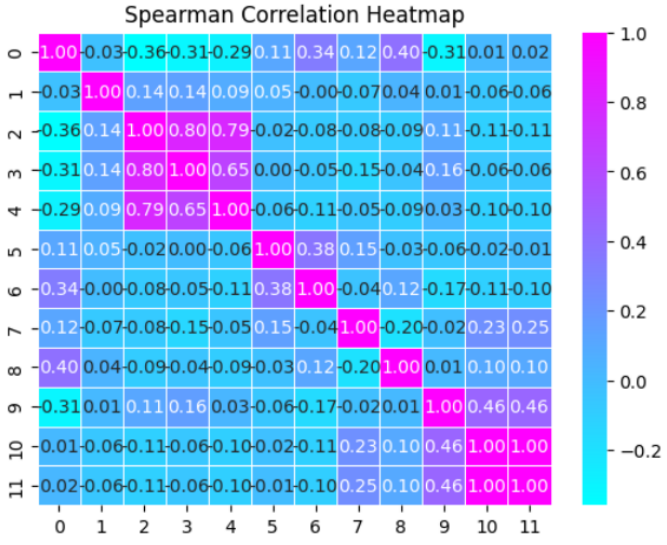


Fig. 2. Correlation Analysis

From the graph, it can be seen that mhc.class and mhc.b are completely correlated, and j.segm, v.segm, and gene are highly correlated. So in order to avoid the problem of high collinearity in the model, we need to avoid using highly correlated data simultaneously.

Next, we counted the data categories of some attributes as shown in the Figure 4:

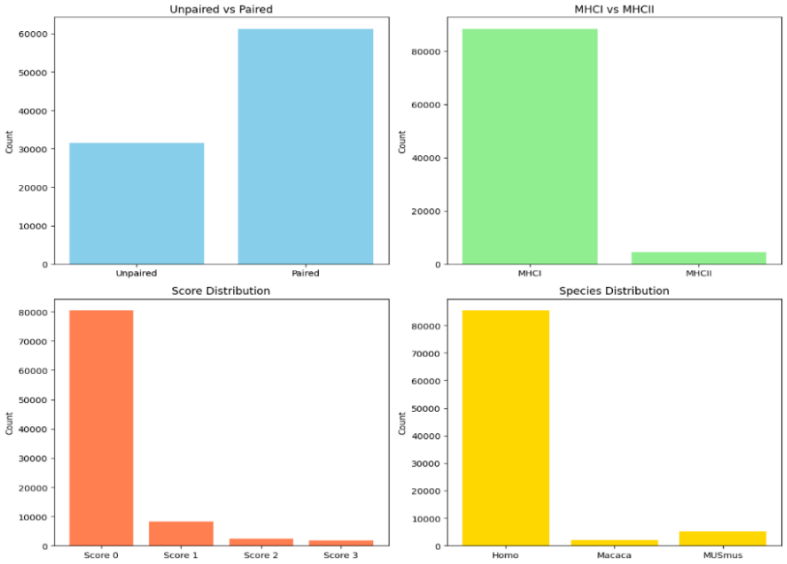


Fig. 3. Data categories for certain attributes

Lu Tianshi's team[2] also focused on the data in the above figure when using deep learning to predict TCR specificity. When processing the data, they filtered out paired data that only selected MHC category as MHCI and belonged to human species. Rudolph, M. G [3] also pointed out in their article that the interaction process between TCR and MHC is different between MHCI and MHCII. Kim, S. M. [4] emphasized the importance of paired data in the article, pointing out that -and -chains play a significant role in the function of T cells.

Based on the above analysis, our preprocessing steps are as follows:

Step 1, retain only the complex.id, cdr3, gene, antigen.epitope, species, vdjdb.score, and mhc.class attributes.

Step 2, filter out data with mhc.class as MHCI.

Step 3, filter out paired data with complex.id not 0.

Step 4, filter out the data with species as HomoSapien.

Step 5, filter out data with vdjdb.score not being 0.

Step 6, rearrange the paired datapoint and merge them into the same datapoint(cdr3 with TRA and cdr3 with TRB).

V. Results and Discussions

A. Different encoding methods for TCR

Using only the cdr3 sequence for specificity prediction is the best approach, so we first discuss the encoding method of the cdr3 sequence. A common encoding method is one-hot representation.

1) Encoding sequences using One-hot representation: We first counted all types of amino acids in CDR3, and the one-hot representations are as follows:

Amino acids	One-hot representation
A	[1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
C	[0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
D	[0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
E	[0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
F	[0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
G	[0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
H	[0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
I	[0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
K	[0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
L	[0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
M	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0]
N	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0]
P	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0]
Q	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0]
R	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0]
S	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0]
T	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0]
V	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0]
W	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0]
Y	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1]

Fig. 4. Enter Caption

After using mapping code encoding, each cdr3 sequence is encoded as a matrix with a size of length * 20. Due to the non-uniform length, we use 0 padding backward, and after padding, the cdr3 sequence is encoded as a matrix with a maximum length * 20. The above is the entire process of cdr3 one-hot representation. The encoded cdr3 is shown in the following formula:

$$Encode(CDR3) = padding(onehotmapping(CDR3)) (1.1)$$

We can choose to use one-hot representation for subsequent calculations, but there are many problems with one-hot representation. Firstly, it ignores the length distribution of cdr3. Secondly, it ignores the relationships between different amino acids. Finally, it ignores the local structural information of cdr3. We have the following improvement methods.

2) Using additional features to improve One-hot representation: Given the idea that spam classification can improve classification accuracy by adding additional email length information, we can also incorporate the length information of cdr3 as additional input into the encoding process. The encoded cdr3 is shown in the following formula:

$$Encode(CDR3) = padding\big(onehotmapping(CDR3)\big) + length(CDR3) (1.2)$$

3) Encoding sequences using word bag model: We can also use the bag of words model to reduce the sparsity of dimensions in single hot encoding. For a CDR3 sequence, we count the number of each amino acid in the sequence and combine it into a 1 * 20 vector, with each dimension representing the number of that amino acid. Compared to single hot encoding, it significantly reduces the dimensionality, making it easier for some machine learning models to converge. The formula is as follows:

$$Encode(CDR3) = Wordbag(CDR3) (1.3)$$

4) Encoding sequences using N-gram model: Although the word bag model significantly reduces dimensions, it only considers length information and loses a lot of information about structure and order. We can encode the cdr3 sequence using N-gram, which preserves local sequence information, length information, and flexibly adjusts the dimension of the feature space. The formula is as follows: Given the excellent

$$Encode(CDR3) = Ngram(CDR3, N, feature) (1.4)$$

performance of N-gram for TCR specificity, we ultimately chose the N-gram model to encode the cdr3 sequence. However, we can provide a comprehensive summary of the effects of different encoding methods on different encoding methods by discussing (1.1), (1.2), (1.3), and (1.4), and provide experimental basis for subsequent researchers.

B. Selecting candidate antigens based on editing distance

Vujovic M[1] used the distance matrix of TCR for clustering in their paper. Inspired by this, the first part of our model framework also selects candidate antigens through the distance matrix of TCR.

When we calculate the distance between sequences, we take into account the edit distance, which is the minimum number of times a sequence can be changed into another sequence by adding characters, deleting characters, and modifying characters. For each change in the sequence, three cases are considered. Through calculation, we can get the formula for editing distance as follows:

$$D[i,j] = D[i-1,j] + 1 \ if \ next \ step \ is \ deleting (1.1)$$
$$D[i,j] = D[i,j-1] + 1 \ if \ next \ step \ is \ adding (1.2)$$
$$D[i,j] = D[i-1,j-1] \ if \ next \ step \ is \ modifying \ and \ S_1[i] == S_2[j] (1.3)$$
$$D[i,j] = D[i-1.j-1] + 1 \ if \ next \ step \ is \ modifying \ and \ S_1[i] != S_2[j] (1.4)$$

D[i,j] represents the distance from the first character to the i character in and the distance from the first character to the j character in .

And, since the distance from any string to the empty string is the length of the string itself. We have the following formula for initializing D:

$$D[i,0] = i \ and \ D[0,j] = j (1.5)$$

From (1.1) (1.2)(1.3)(1.4)(1.5) we can get the edit distance of the two sequences.

Matrix	TRA	TRB	TRA-TRB
variance	0. 96041	0. 965254	0. 93969

Fig. 5. PCA explained variance

1) TCR visualization based on PCA and t-SNE: After calculating TRA, TRB, and TRA-TRB separately, we obtained three different distance matrices based on edit distance. Then we first use PCA to reduce the distance matrix to 50 dimensions and explain the variance as shown in the table:

It can be seen that after using PCA to reduce dimensionality to 50 dimensions, most of the information was still retained. Then, we used t-SNE for dimensionality reduction, and the results of dimensionality reduction are shown in the following figure:

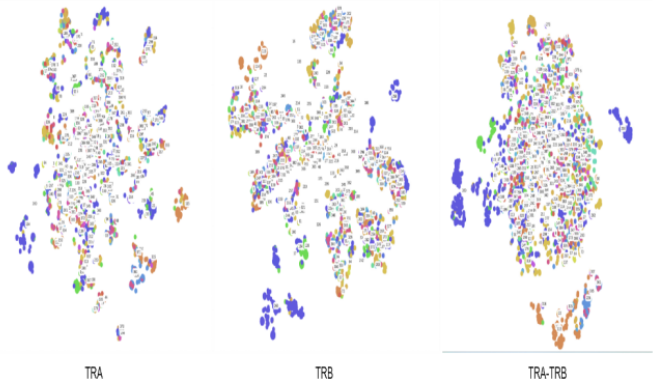


Fig. 6. Visualization after dimensionality reduction

It can be intuitively seen that the clusters of TRB after dimensionality reduction are more prominent, and more TCRs of the same antigen category are assigned to the same cluster.

2) Clustering TCR based on editing distance: We can directly use clustering algorithms to cluster the distance matrix and obtain clustering results, as shown in the Figure 7,Figure 8:

	TRA	TRB	TRA-TRB
NMI	0. 01507	0. 02621	0. 10711
ARI	0. 112103	0. 113624	0. 022367

Fig. 7. Cluster performance based on k-means

	TRA	TRB	TRA-TRB
NMI	0. 41187	0. 405358	0. 352910
ARI	0. 01912	0. 014908	0. 00949

Fig. 8. Cluster performance based on DBSCAN

From the table, it can be seen that directly using clustering algorithms has poor clustering performance on the entire distance matrix. This is because TCRs with specificity for similar antigens will be grouped into the same cluster, and there are many such TCRs in the database. If clustering is only performed on TCRs with dissimilar antigens, the effect will be relatively ideal, as shown in the following figure:

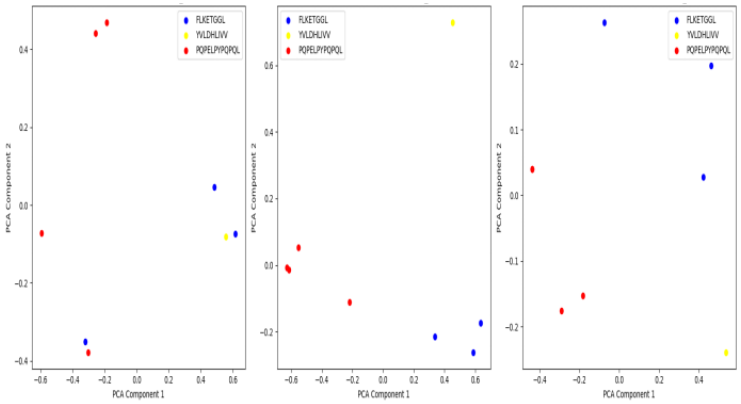


Fig. 9. Dimensionality reduction display of minority points with dissimilar antigens

For a small number of TCRs with dissimilar antigens, the clustering effect will be very good, but for databases with a large number of TCRs, it is not appropriate to directly use clustering to predict specificity. However, we can still conclude that TCRs with close editing distances have similar antigens.

So the idea of the first part of our model is to generate the target antigen through a distance matrix: calculate the edit distance matrix of the TRB between the target TCR and other TCRs in the training database, calculate the top five closest training data, and use our subsequent classifier for further more accurate discrimination.

C. Specificity Classifier Based on Random Forest

1) Data preparation: We use data with vdjdb.score 0 as negative class data (0 labels) and data with vdjdb.score 1, 2, and 3 as positive class data (1 label). Due to the large number of negative class data compared to positive class data, in order to reduce model bias caused by class imbalance, we adopt a random sampling method to sample negative class data, randomly selecting the same amount of negative class data as positive class data as training data.

Next, we used N-gram models to encode the CDR3 sequence and antigen receptor sequence, respectively. The hyperparameters of the model are as follows:

parameter	analyzer	max features	ngram_range	lowercase
value	char	2000	(5,5)	False

Fig. 10. Hyperparameter settings

2) Construction of Random Forest: We will use the encoded CDR3 and antigen sequence as feature inputs and labels as model outputs to construct a random forest model(RF).

The model construction is shown in the following figure:

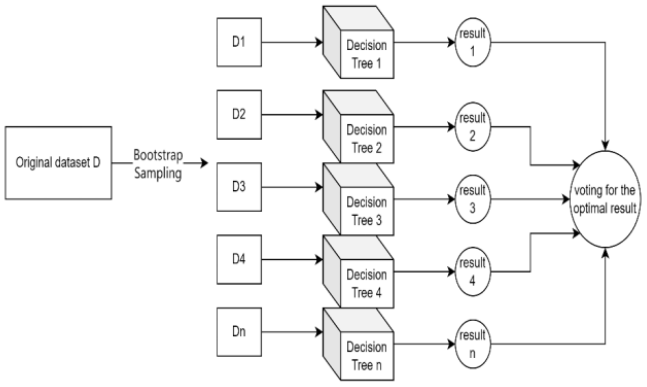


Fig. 11. RF classification principle

For the construction of decision trees, we use the GINI coefficient for evaluation, as shown by the following formula:

$$GINI(D) = 1 - \sum_{i=1}^{k} (P_i)^2 (1.1)$$

In formula (1.1), D represents the dataset, k represents the number of categories, and represents the proportion of samples belonging to category i in the dataset.

3) Selection of hyperparameters: For a random forest, selecting how many evaluators (number of decision trees) is an important hyperparameter. We use k-fold validation to calculate the average accuracy of samples for different numbers of random forest models, as shown in the following figure:

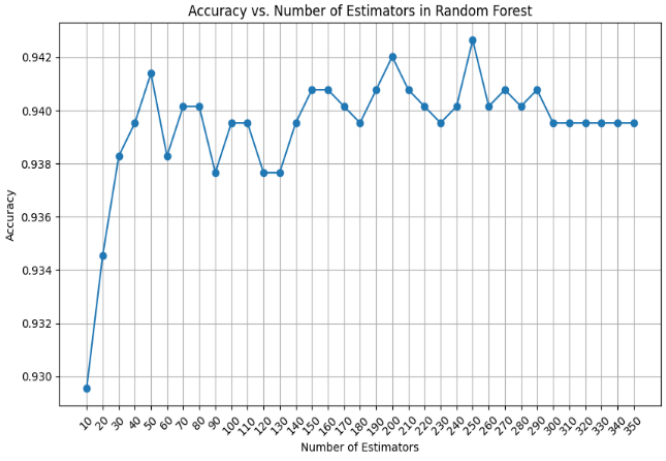


Fig. 12. Accuracy vs. Number of Estimators in Random Forest

From the graph, we can intuitively see that when the number of decision trees is 250, the model achieves the best accuracy. So we have chosen 250 evaluators, and our choices for other hyperparameters and k-fold validation settings are shown in the table below:

parameter	setting
max_depth	[None, 5, 10, 15]
max_features	['auto', 'sqrt', 'log2']
min_samples_split	[2, 5, 10]
cv	5
scoring	accuracy

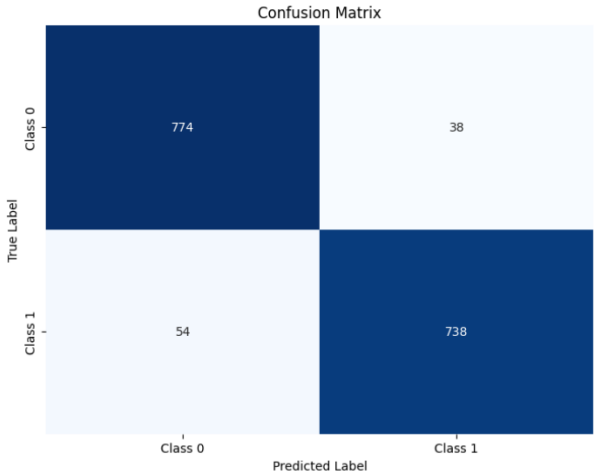
Fig. 13. hyperparameters settings

The final hyperparameter design determined by grid search is shown in the table below:

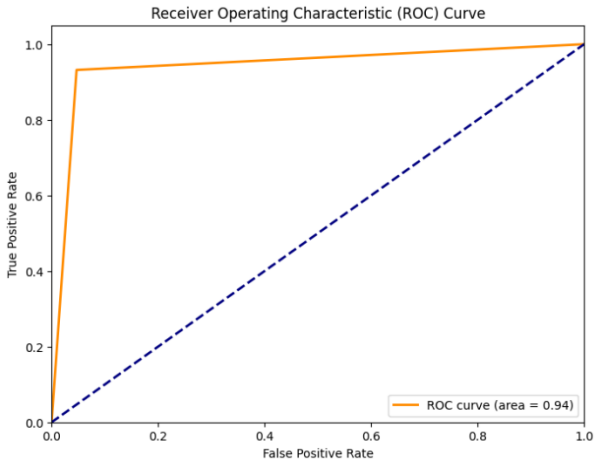
parameter	setting
max_depth	None
max_features	log2
min_samples_split	2
n_estimators	250

Fig. 14. final hyperparameter design

4) Prediction of Random Forest:



Recall: 0.9318181818181818
Precision: 0.9510309278350515
F1 Score: 0.9413265306122448
Accuracy: 0.942643391521197



VI. Further Work and lmprovement

In this section, we will focus on better data augmentation methods, such as using GANs to enhance data.

VII. Conclusion

TCR specificity prediction technology has broad prospects in many fields such as cancer immunotherapy and autoimmune disease treatment. Compared with experimental methods, the computational methods that uses algorithms to achieve TCR-specific prediction has various advantages like high efficiency, economy, and repeatability. Inspired by previous work, we established a TCR generative specificity detection framework to efficiently screen out TCRs and corresponding antigens through TCR classifier and specificity classifier based on random forest. A series of comparisons and analyzes indicate that our model generally outperforms ordinary algorithms.

However, our model still has shortcomings and faces some challenges. First, scarcity and quality of data and high redundancy of the available data still exist (Montemurro et al., 2022b). On the one hand, relevant data collection work is required to obtain richer and high-quality available data. On the other hand, it is feasible to enhance the dataset using machine learning methods (GAN) or using data from multiple databases. Furthermore, the search time may be too long for some TCRs. If we have more time, we will optimize the algorithm for selecting target antigens in our model.

References

[1] Esfahani, K., Roudaia, L., Buhlaiga, N., Del Rincón, S. V., Papneja, N., Miller, W. H. (2020). A review of cancer immunotherapy: From the past, to the present, to the future. Current Oncology, 27(12), 87–97. https://doi.org/10.3747/co.27.5223

[2] Mariuzza, R. A., Agnihotri, P., Orban, J. (2020). The structural basis of T-cell receptor (TCR) activation: An enduring enigma. Journal of Biological Chemistry/the Journal of Biological Chemistry, 295(4), 914–925. https://doi.org/10.1016/s0021-9258(17)49904-2

[3] Schuster, M., Nechansky, A., Kircheis, R. (2006). Cancer immunotherapy. Biotechnology Journal, 1(2), 138–147. https://doi.org/10.1002/biot.200500044

[4] Shugay, M., Bagaev, D. V., Zvyagin, I. V., Vroomans, R. M. A., Crawford, J. C., Dolton, G., Komech, E. A., Sycheva, A. L., Koneva, A. E., Egorov, E. S., , . ., Van Dyk, E., Dash, P., Attaf, M., Rius, C., Ladell, K., McLaren, J. E., Matthews, K., Clemens, E. B., . . . Chudakov, D. M. (2017). VDJdb: a curated database of T-cell receptor sequences with known antigen specificity. Nucleic Acids Research, 46(D1), D419–D427. https://doi.org/10.1093/nar/gkx760

[5] World Health Organization: WHO. (2024, February 1). Global cancer burden growing, amidst mounting need for services. WHO. https://www.who.int/news/item/01-02-2024-global-cancer-burden-growing–amidst-mounting-need-for- services

[6] Zhao, M., Xu, X. S., Yang, Y., Yuan, M. (2023). GGNPTCR: a generative graph structure neural network for predicting immunogenic peptides for T-cell immune response. Journal of Chemical Information and Modeling, 63(23), 7557–7567. https://doi.org/10.1021/acs.jcim.3c01293

[7] Antigenomics. (2023, June). vdjdb-db. GitHub. https://github.com/antigenomics/vdjdb-db/blob/2023-06-01/README.md.

[8] 2.Lu, T., Zhang, Z., Zhu, J., Wang, Y., Jiang, P., Xiao, X., ... Wang, T. (2021). Deep learning-based prediction of the T cell receptor–antigen binding specificity. Nature machine intelligence, 3(10), 864-875.

[9] 3.Rudolph, M. G., Wilson, I. A. (2002). The specificity of TCR/pMHC interaction. Current opinion in immunology, 14(1), 52-65.

[10] 4.Kim, S. M., Bhonsle, L., Besgen, P., Nickel, J., Backes, A., Held, K., ... Prinz, J. C. (2012). Analysis of the paired TCR -and -chains of single human T cells. PloS one, 7(5), e37338.

[11] Vujovic, M., Degn, K. F., Marin, F. I., Schaap-Johansen, A. L., Chain, B., Andresen, T. L., ... Marcatili, P. (2020). T cell receptor sequence clustering and antigen specificity. Computational and Structural Biotechnology Journal, 18, 2166-2173.

[12] Montemurro, A., Jessen, L. E., Nielsen, M. (2022b). NetTCR-2.1: Lessons and guidance on how to develop models for TCR specificity predictions. Frontiers in Immunology, 13. https://doi.org/10.3389/fimmu.2022.1055151