GTE: A Graph Learning Framework for Prediction of T-cell Receptors and Epitopes Binding Specificity

Appendix

A Experiments set up

A.1 Baseline Models

In this paper, we compare three baselines: TEIM [6], TEINet [3], and pMTnet [5]. The prediction of TCR-epitope interactions by TEIM involves sequence contact prediction and residue contact prediction. The training process includes two phases: pre-training TEIM-Seq on sequence-level binding data and then fine-tuning TEIM-Res on residue-level binding data. During this process, an epitope-based pre-training model is added to speed up the model convergence. This step-by-step approach helps in learning localized interaction information between TCR and epitope pairs.

TEINet is a deep learning framework that utilizes transfer learning to solve this predictive problem. It converts TCR and epitope sequences into numerical vectors using two separate pre-trained encoders. The vectors are then input into a fully connected neural network to predict their binding specificity.

pMTnet differs from the previous two baselines in that it includes additional MHC information. Specifically, pMTnet employs Long Short-Term Memory (LSTM) networks to train numerical embeddings for pMHC (Class I only), allowing the representation of peptides and MHC protein sequences in a numerical format. Second, pMTnet employs stacked autoencoders to train embeddings for TCR sequences, which once again encode text strings of TCR sequences into numerical representations. In the final stage, pMTnet builds a deep neural network on top of these two embeddings, combining knowledge about TCR, peptide sequences, and MHC allele genes for binding prediction. It is worth mentioning that pMTnet introduces a differential loss function, which is a novel loss function designed to ensure that the prediction scores for positive samples are greater than those for negative samples. When reproducing pMTnet, we replaced MHC information with padding symbols to disregard MHC information to ensure a fair comparison.

A.2 Input features

The input features of TCR and epitopes are obtained by a SOTA pre-trained model TCRpeg [4]. TCRpeg is an advanced autoencoder that employs a recurrent neural network with GRU layers to characterize TCRs, enabling the capture of critical features from sequence inputs through unsupervised learning. We utilize two separate pre-trained TCRpeg models to encode TCRs and epitopes, with both resulting in vector embeddings of 768 dimensions for consistency.

A.3 Model architecture

As shown in Fig. S1, the GNN model incorporates two SAGEConv [2] layers for node feature propagation and information aggregation within the heterogeneous graph of TCR and epitopes. A ReLU activation function connects these layers to introduce non-linear relationships. A dropout rate of 0.1 is used to prevent overfitting. The output of the GNN is then fed into a two-layer MLP to process the node features of the graph further. The two layers' output dimensions are 256 and 128, respectively. ReLU is used as the activation function, and the dimension of the final output layer is set as 1 for binary classification.

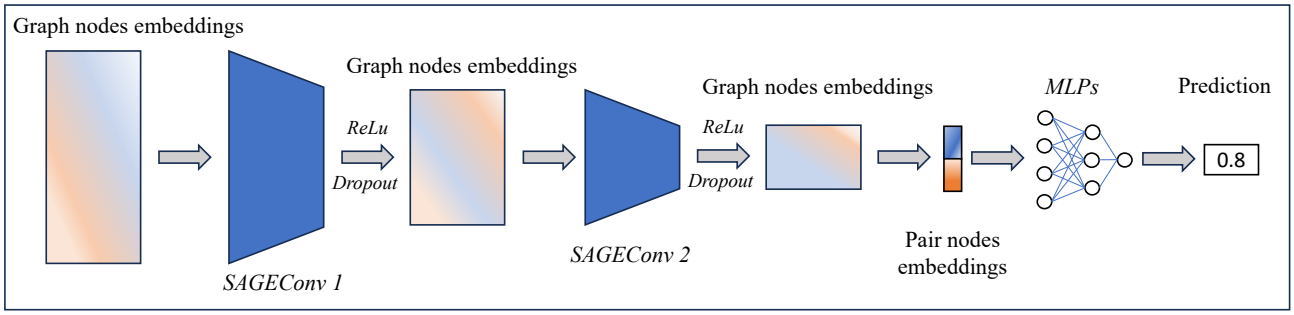

Fig. S1. GTE model architecture.

A.4 Evaluation metric

The effectiveness of our approach is evaluated by two primary metrics: Area under the Curve (AUC) and Area under the Precision-Recall Curve (AUPR) [1]. These metrics are widely used as the evaluation criteria in machine learning and bioinformatics to assess binary classification models.

Area under the Curve: AUC quantifies the ability of the model to distinguish between positive and negative samples. It provides a single, intuitive value that reflects the overall classification performance, calculated as:

$$AUC = \int_0^1 \mathrm{ROC}(x)\, dx,$$

where $\mathrm{ROC}(x)$ is the Receiver Operating Characteristic curve.

Area under the Precision-Recall Curve: AUPR focuses on the precision-recall trade-off, which is particularly important in imbalanced datasets. It measures the precision of positive predictions against the recall (sensitivity) rate, computed as:

$$AUPR = \int_0^1 \text{P-R}(x)\, dx,$$

where $\text{P-R}(x)$ represents the Precision-Recall curve.

These two metrics collectively offer a robust evaluation framework for assessing the predictive capabilities of our method, ensuring a comprehensive understanding of its strengths and limitations.

A.5 Infrastructure and software

Our model is implemented through the Pytorch package. Our model is trained in a self-hosted 4-GPU server with Intel i7 6700K @ 4.00 GHz CPU, 64 Gigabytes RAM, and four Nvidia GTX 1080Ti GPUs.

B Performance of GTE

Figure S2 shows the results for four datasets using the RandomTCR split method. Our results are significantly better than the baseline.

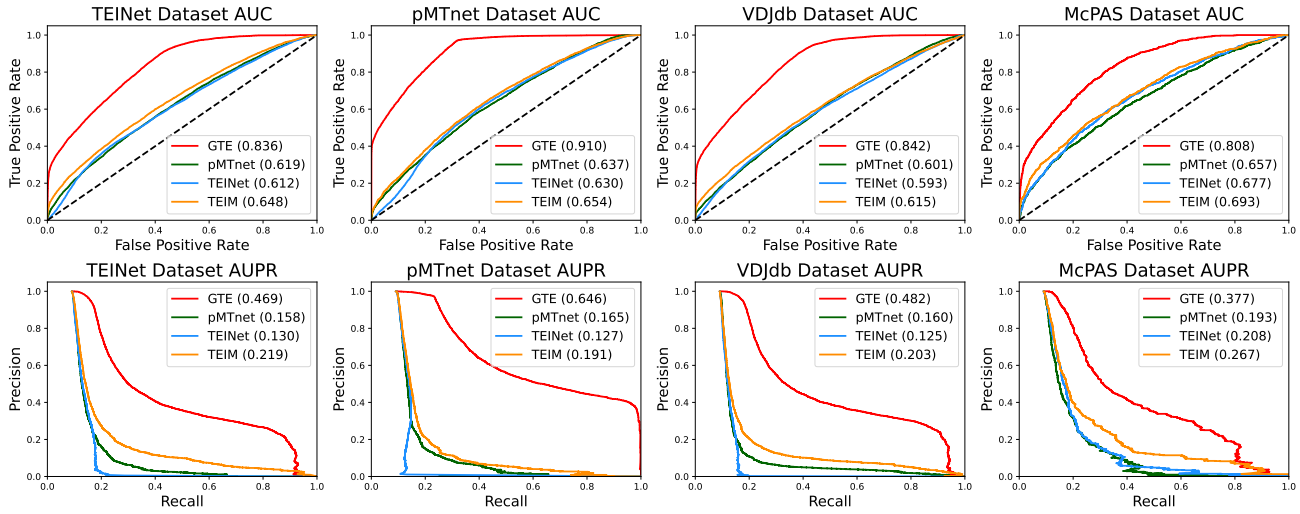

Fig. S2. Comparison results under **RandomTCR** setting. Experiments are conducted on four datasets using three baseline models. The first row demonstrates the visualizations of AUC scores and the second row illustrates the AUPR scores.

Table S1 and Table S2 illustrate the performance of GTE on four datasets. Specifically, Table S1 presents the results for GTE and three baseline methods using the RandomTCR splitting, and Table S2 shows the results with the StrictTCR splitting. Each result represents the average of 5-fold cross-validation with the standard deviation. It is evident that, regardless of the splitting method, our results consistently outperform the baselines. This underscores the efficacy of graph-based topological structure methods in enhancing TCR-epitope prediction outcomes.

Table S1. Performance Comparison in the **RandomTCR** Setting: Mean AUC and AUPR scores with standard deviations (std) across four datasets with 5-fold cross-validation are reported.

		TEINet	TEIM	pMTnet	GTE (ours)
TEINet	AUC	0.6118±0.0065	0.6483±0.0063	0.6192±0.0036	**0.8358**±0.0059
dataset	AUPR	0.1302±0.0031	0.2191±0.0050	0.1583±0.0033	**0.4689**±0.0156
pMTnet	AUC	0.6301±0.0078	0.6544±0.0028	0.6371±0.0063	**0.9095**±0.0017
dataset	AUPR	0.1274±0.0022	0.1911±0.0096	0.1652±0.0055	**0.6458**±0.0046
VDJdb	AUC	0.5934±0.0165	0.6149±0.0024	0.6014±0.0034	**0.8419**±0.0070
dataset	AUPR	0.1253±0.0173	0.2033±0.0066	0.1603±0.0061	**0.4816**±0.1259
McPAS	AUC	0.6773±0.0134	0.6931±0.0100	0.6569±0.0089	**0.8077**±0.0145
dataset	AUPR	0.2081±0.0212	0.2672±0.0245	0.1934±0.0092	**0.3766**±0.0400

Table S2. Performance Comparison in the **StrictTCR** Setting: Mean AUC and AUPR scores with standard deviations (std) across four datasets with 5-fold cross-validation are reported.

		TEINet	TEIM	pMTnet	GTE (ours)
TEINet	AUC	0.6072±0.0059	0.6354±0.0030	0.6162±0.0012	**0.8389**±0.0054
dataset	AUPR	0.1294±0.0023	0.2114±0.0015	0.1533±0.0070	**0.4863**±0.0119
pMTnet	AUC	0.6084±0.0062	0.6372±0.0020	0.6344±0.0034	**0.9114**±0.0031
dataset	AUPR	0.1311±0.0074	0.1753±0.0050	0.1652±0.0078	**0.6549**±0.0068
VDJdb	AUC	0.5814±0.0121	0.6168±0.0020	0.6022±0.0034	**0.8471**±0.0052
dataset	AUPR	0.1184±0.0153	0.2024±0.0037	0.1603±0.0089	**0.4999**±0.0101
McPAS	AUC	0.6622±0.0042	0.6904±0.0035	0.6453±0.0094	**0.8125**±0.0143
dataset	AUPR	0.2369±0.0055	0.3159±0.0129	0.1934±0.0089	**0.3904**±0.0313

C Single Sequence Protein Model Evaluation

Table S3 displays the results of using different embeddings on four datasets with the RandomTCR splitting method. The findings indicate that the outcomes of embeddings based on ESM2 are slightly inferior to those obtained by employing TCRpeg as the embedding. It is observed that the results produced by pre-trained models based on TCR and epitope sequences are marginally superior to those generated by large-scale protein language models. One possible reason is that TCR and epitope sequences are relatively short, which might lead to a performance decline when using embeddings generated from large protein language models.

Table S3. Comparison experiments are conducted to evaluate different input embeddings using the proposed GTE model on four datasets with the **RandomTCR** setting. Both AUC and AUPR scores are reported. Higher scores are better, and the best scores are marked in **bold**.

Embeddings	TEINet Dataset		pMTnet Dataset		VDJdb Dataset		McPAS Dataset	
	AUC	AUPR	AUC	AUPR	AUC	AUPR	AUC	AUPR
ESM2(650M)	0.826±0.007	0.444±0.016	0.900±0.001	0.613±0.005	0.828±0.006	0.448±0.013	0.739±0.025	0.299±0.041
ESM2(3B)	0.817±0.09	0.431±0.017	0.900±0.003	0.612±0.008	0.826±0.005	0.445±0.009	0.743±0.025	0.305±0.042
TCRpeg	**0.836**±0.006	**0.469**±0.016	**0.910**±0.002	**0.646**±0.005	**0.842**±0.007	**0.482**±0.013	**0.808**±0.014	**0.377**±0.040

D MSA-based Protein Model Evaluation

Experimental results show that the MSA-based (AlphaFold2-based) protein model achieved an accuracy of 0.4292 on all positive samples in McPAS. As shown in Table S4, the MSA count and prediction results for all positive samples are reported.

E Model Complexity

To illustrate the model complexity, we execute a meticulous comparison of the parameter counts between our GTE model and three baseline models. Specifically, pMTnet [5] has 128,223 parameters, TEIM [6] has 1,214,723 parameters, TEINet [3] boasts a staggering 19,420,973 parameters, while our GTE model has only 295,681 parameters. Our model outperforms all baseline methods while keeping a low complexity. In practical applications, a smaller number of parameters translates to reduced memory usage and faster inference speeds, which are essential in resource-constrained environments. This demonstrates the efficiency of graph-based topological methods in maintaining high prediction accuracy while conserving computational resources.

F Inference Phase

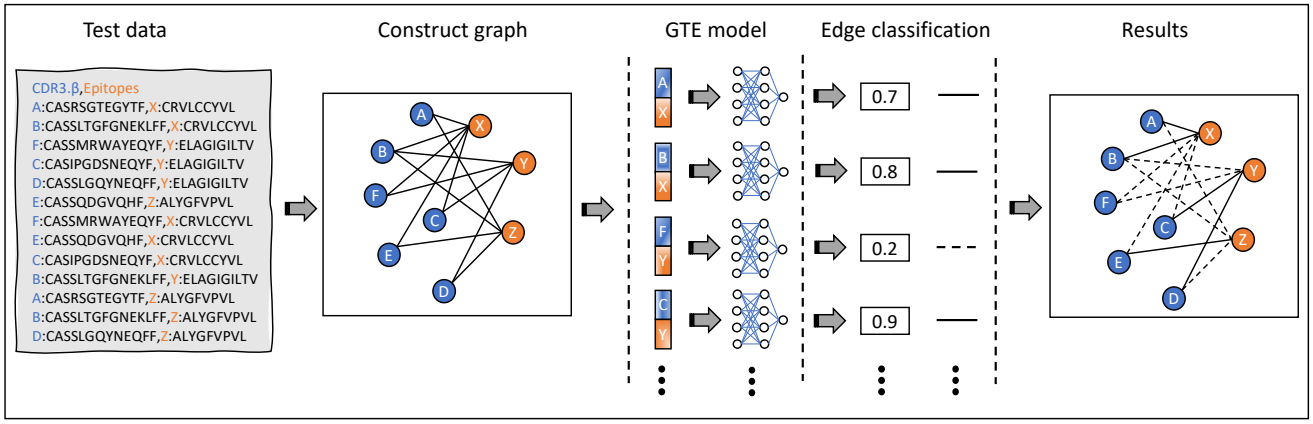

Fig. S3. GTE inference framework.

As part of our discourse, we offer an elucidation on the inference phase of GTE. As illustrated in Figure S3, the test data is structured as a graph, with CDR3.β nodes denoted in blue and epitope nodes in red. Each edge signifies a sample pair, and thus, the number of samples corresponds to the number of edges in the graph. Subsequently, this graph is input into the GTE model, where GTE concatenates the embeddings of the two connected nodes linked by each edge. It then passes through an MLP layer to compute a binding score. Based on this score, the edges are classified as solid or dashed. Solid edges represent binding between the sample pairs, while dashed edges indicate negative samples.

Table S4. The quantity of MSA and the prediction results on the FoldDock.

msa_num	sample_num	prediction 0	prediction 1	accuray
2	3069	1671	1398	0.4555
3	258	144	114	0.4419
4	106	72	34	0.3208
5	19	12	7	0.3684
6	7	4	3	0.4286
7	3	3	0	0.0000
8	16	14	2	0.1250
14	1039	709	330	0.3176
15	92	72	20	0.2174
16	35	28	7	0.2000
17	7	5	2	0.2857
18	8	7	1	0.1250
19	1	1	0	0.0000
20	4	4	0	0.0000
21	1	0	1	1.0000
22	28	18	10	0.3571
23	9	7	2	0.2222
25	4	2	2	0.5000
32	257	86	171	0.6654
33	7	1	6	0.8571
34	19	16	3	0.1579
35	5	5	0	0.0000
37	3	0	3	1.0000
38	1	0	1	1.0000
47	1	1	0	0.0000
434	49	14	35	0.7143
435	1	0	1	1.0000
440	2	2	0	0.0000
948	1	0	1	1.0000
3117	3	1	2	0.6667
4467	1	0	1	1.0000
8888	4	2	2	0.5000
10347	1	1	0	0.0000
20794	8	3	5	0.6250
21283	32	7	25	0.7813
21284	1	0	1	1.0000

References

1. Tom Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874, 2006.
2. Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.
3. Yuepeng Jiang, Miaozhe Huo, and Shuai Cheng Li. Teinet: a deep learning framework for prediction of tcr–epitope binding specificity. *Briefings in Bioinformatics*, 24(2):bbad086, 2023.
4. Yuepeng Jiang and Shuai Cheng Li. Deep autoregressive generative models capture the intrinsics embedded in t-cell receptor repertoires. *Briefings in Bioinformatics*, 24(2):bbad038, 2023.
5. Tianshi Lu, Ze Zhang, James Zhu, Yunguan Wang, Peixin Jiang, Xue Xiao, Chantale Bernatchez, John V Heymach, Don L Gibbons, Jun Wang, et al. Deep learning-based prediction of the t cell receptor–antigen binding specificity. *Nature machine intelligence*, 3(10):864–875, 2021.
6. Xingang Peng, Yipin Lei, Peiyuan Feng, Lemei Jia, Jianzhu Ma, Dan Zhao, and Jianyang Zeng. Characterizing the interaction conformation between t-cell receptors and epitopes with deep learning. *Nature Machine Intelligence*, 5(4):395–407, 2023.