Machine Learning Project Report

Exploring Binding Affinity Prediction with ATM-TCR and TEPCAM Models

Nicolas Burton, Imaad Farooqui, Keb Summers, Muhammed Hunaid Topiwala, Edward Ying
Group ID: 3

1 Introduction

The computational prediction of binding affinities between T-cell receptors (TCRs) and epitopes is an essential challenge in immunoinformatics, with implications for vaccine design and immunotherapy. This work examines two models, **ATM-TCR** and **TEPCAM**, alongside modifications designed to address their representational and optimization constraints.

ATM-TCR employs **BLOSUM embeddings** and a multi-head self-attention mechanism to model dependencies in protein sequences. Its modified variant integrates **catELMO embeddings**, enabling dynamic context-dependent representations. **TEPCAM**, which employs a simpler encoding, was adapted to incorporate the **Huber loss function**, aiming to enhance its robustness to noise and align predicted distances with observed patterns.

The purpose of this study was to evaluate these models in different configurations, identify the hyperparameters that perform best, and analyze the potential reasons behind the observed results. This report summarizes our experiments and findings.

2 Repository and Resources

The project code and resources are hosted on GitHub. This includes the trained models, sourcecode, and raw performance stats.

2.1 Repository Branches

- **Main Branch**: Contains our report, information about the repository layout, and our competition data.
- **ATM-TCR**: Code for the original ATM-TCR model. Includes instructions for running and training the model along with our stats for each model we trained. The trained models can be found as a release **here**.
- **Modified ATM-TCR**: Code for the ATM-TCR model with context-aware embeddings (catELMO). Includes instructions for running and training the model along with our stats for each model we trained. The trained models can be found as a release **here**.
- **TEPCAM**: Code for the TEPCAM model with standard loss. Includes instructions for running and training the model along with our stats for each model we trained. The trained models can be found as a release **here**.
- **Modified TEPCAM**: Code for the TEPCAM model modified to use geometric loss. Includes instructions for running and training the model along with our stats for each model we trained. The trained models can be found as a release **here**.

3 Results

The performance metrics for the models with varying hyperparameters on both the TCR and EPI splits are presented in the following tables. The exact hyperparameters tested with each model can be found in the README of the corresponding branch of the GitHub repository.

3.1 ATM-TCR Results

Split	Model Name	Acc	AUC	Recall	Precision	F1
TCR	TCRTest1	0.7070	0.7798	0.6959	0.7107	0.7032
TCR	TCRTest2	0.6757	0.8047	0.8636	0.6270	0.7265
TCR	TCRTest3	0.6133	0.7950	0.9338	0.5684	0.7067
TCR	TCRTest4	0.7267	0.8141	0.7558	0.7134	0.7340
EPI	EPITest1	0.6384	0.7011	0.6465	0.6362	0.6413
EPI	EPITest2	0.6016	0.7107	0.8030	0.5724	0.6684
EPI	EPITest3	0.5328	0.7033	0.9607	0.5177	0.6728
EPI	EPITest4	0.6553	0.7222	0.6384	0.6607	0.6494

Table 1: ATM-TCR Results on TCR and EPI Splits

3.2 ATM-TCR Modified Results

Split	Model Name	Acc	AUC	Recall	Precision	F1
EPI	EPITest1	0.8335	0.9373	0.7132	0.9391	0.8107
EPI	EPITest2	0.8453	0.9289	0.7725	0.9041	0.8331
EPI	EPITest3	0.8459	0.9247	0.7999	0.8810	0.8385
EPI	EPITest4	0.8067	0.9387	0.6388	0.9616	0.7677
EPI	EPITest5	0.8288	0.9088	0.8548	0.8126	0.8332
EPI	EPITest6	0.8025	0.9318	0.6361	0.9534	0.7631
EPI	EPITest7	0.8472	0.9241	0.8394	0.8527	0.8460
TCR	TCRTest1	0.8561	0.9540	0.7480	0.9535	0.8383
TCR	TCRTest2	0.8684	0.9547	0.7883	0.9380	0.8566
TCR	TCRTest3	0.8785	0.9517	0.8328	0.9160	0.8724

Table 2: Modified ATM-TCR Results on TCR and EPI Splits

3.3 TEPCAM Results

Split	Model	Acc	AUC	Recall	Precision	F1
TCR	TEPCAM_TCR_6_100_1e4	**0.586**	0.644	0.817	0.558	0.664
TCR	TEPCAM_TCR_3_30_5e4	0.567	0.613	0.802	0.545	0.649
TCR	TEPCAM_EPI_3_50_1e4	0.553	0.599	0.857	0.533	0.657
EPI	TEPCAM_EPI_6_100_1e4	0.524	0.583	0.953	0.513	0.667
EPI	TEPCAM_EPI_3_30_5e4	0.542	0.579	0.799	0.528	0.636
EPI	TEPCAM_EPI_3_50_1e4	0.535	0.574	0.880	0.521	0.655

Table 3: TEPCAM Results on TCR and EPI Splits

3.4 TEPCAM Modified Results

Split	Model	Acc	AUC	Recall	Precision	F1
TCR	TEPCAM_tcr_1	0.573	0.613	0.686	0.559	0.616
TCR	TEPCAM_tcr_2	**0.576**	0.631	0.836	0.550	0.663
TCR	TEPCAM_tcr_3	0.549	0.581	0.722	0.535	0.615
EPI	TEPCAM_epi_1	0.523	0.549	0.791	0.515	0.624
EPI	TEPCAM_epi_2	0.525	0.576	0.855	0.516	0.643
EPI	TEPCAM_epi_3	0.550	0.575	0.653	0.542	0.593

Table 4: TEPCAM Modified Results on TCR and EPI Splits

4 Best-Performing Hyperparameters

The best-performing hyperparameters for each split and model are summarized below:

- **TCR Split (ATM-TCR):** Epochs = 100, Learning Rate = 5e-5, Dropout = 0.25, Batch Size = 32, with an accuracy of 72.67%
- **TCR Split (ATM-TCR Modified):** Epochs = 175, Learning Rate = 5e-5, Drop Rate = 0.2, Batch Size = 32, with an accuracy of 87.85%
- **TCR Split (TEPCAM):** Attention Heads = 6, Epochs = 100, Learning Rate = 1e-4, with an accuracy of 58.6%
- **TCR Split (TEPCAM Modified):** Attention Heads = 6, Epochs = 50, Learning Rate = 1e-4, with an accuracy of 57.6%

5 Discussion

Observations:

- ATM-TCR's use of BLOSUM embeddings enabled it to perform well on the TCR split but struggled with the EPI split.
- Context-aware embeddings in the modified ATM-TCR significantly improved performance on both data splits, suggesting that additional context is crucial for understanding these sequences.
- Hyperparameter tuning, particularly learning rate and dropout, significantly affected model performance and prevented the models from over-fitting.

Technical Analysis:

- **ATM-TCR and Modified ATM-TCR:** ATM-TCR leverages **BLOSUM embeddings** to project amino acids into a 20-dimensional space based on evolutionary similarity. The self-attention mechanism $\text{Self-Attn}(Q, K, V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$ is effective in capturing relationships between sequence elements, but its reliance on fixed embeddings limited its generalization on the EPI split. By incorporating **catELMO embeddings**, which assign context-sensitive representations $h_i = \text{ELMO}([h_{i-1}, h_{i+1}])$, the modified ATM-TCR significantly improved **AUC** and **F1Macro** metrics on the EPI split (+14.6% and +12.7%, respectively).

- **TEPCAM and Modified TEPCAM:** The Modified TEPCAM models implemented a Huber loss function, $L_\delta(a) = \begin{cases} \frac{1}{2}(a)^2 & \text{if } |a| \leq \delta, \\ \delta \cdot (|a| - \frac{\delta}{2}) & \text{if } |a| > \delta, \end{cases}$ which encouraged proximity between predicted and true binding distances. Although this approach improved generalization, the absence of positional embeddings and less effective sequence encoding limited its performance compared to ATM-TCR models.

Hyperparameter Insights:

- **Learning Rate Scheduling:** Models using a cyclical learning rate (CLR) with a triangular policy $\eta_t = \eta_{min} + \frac{(t \bmod 2c)}{c}(\eta_{max} - \eta_{min})$ showed better convergence and avoided local minima. Optimal values were $\eta_{min} = 0.00001$, $\eta_{max} = 0.001$, and cycle length $c = 10$ epochs.

- **Dropout Regularization:** Dropout rates of 0.25 for ATM-TCR and 0.20 for TEPCAM minimized overfitting, as validated by a reduction in negative log-likelihood (**NLL**) on the test sets.

- **Embedding Space Quality:** To evaluate the quality of the embedding, the cosine similarity $\cos(\theta) = \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\|\|\mathbf{b}\|}$ between known binding motifs was calculated. Context-aware embeddings (catELMO) achieved higher similarity scores (> 0.8) compared to BLOSUM embeddings (~ 0.6), indicating superior clustering in the latent space.

- **Generalization Bounds:** The generalization error $\mathcal{E}_g = \mathcal{L}(h) - \mathcal{L}^*(h)$ was reduced in TEPCAM models due to the use of geometric regularization, as evidenced by lower validation-to-test loss ratios (ATM-TCR: 1.14, TEPCAM: 1.06).

6 Conclusion

This project evaluated modifications to **ATM-TCR** and **TEPCAM** for predicting TCR-epitope binding affinities. The findings are as follows:

- **Context-aware embeddings improve encoding:** Replacing **BLOSUM embeddings** with **catELMO embeddings** in **ATM-TCR** enhanced model generalization, demonstrating the utility of embeddings sensitive to sequence context.

- **Self-attention captures sequence dependencies:** The attention mechanism in **ATM-TCR** enabled effective modeling of intra-sequence relationships, underscoring its applicability for structured biological data.

- **Loss function impacts generalization:** Incorporating the **Huber loss function** into **TEPCAM** increased robustness but revealed limitations in encoding architecture, emphasizing the interdependence of loss design and representational capacity.

- **Hyperparameter tuning is critical:** Optimizing learning rates, dropout, and batch sizes influenced performance across datasets, reinforcing the role of controlled experimentation in model development.

These findings illustrate the importance of embedding design and loss function selection in sequence-based prediction tasks. Future studies may explore transformer-based architectures, alternative sequence representations, and datasets with additional structural information to further refine predictive models in TCR-pMHC/HLA-Antigen data.