Epitope Generation for Peptide-based Cancer Vaccine using Goal-directed Wasserstein Generative Adversarial Network with Gradient Penalty

Yen-Che Hsiao
Department of Electrical and Computer Engineering
University of Connecticut
Storrs CT 06269, USA
yen-che.hsiao@uconn.edu

Abhishek Dutta
Department of Electrical and Computer Engineering
University of Connecticut
Storrs CT 06269, USA

Abstract

We introduce a novel immunogenicity goal-directed peptide sequence generator in a Wasserstein generative adversarial network (GAN) with gradient penalty. The GAN is trained using the bladder cancer epitope sequences that are predicted to bind with the human leukocyte antigen, HLA-A*0201, to trigger cytotoxic T-cell immune responses. The convolutional neural network-based generator is guided by an immunogenicity predictor from DeepImmuno-CNN and a critic to generate immunogenic epitopes for bladder cancer vaccines. The convolutional neural network-based immunogenicity predictor is trained with class I peptide human leukocyte antigen sequences from the immune epitope database to produce a continuous immunogenic score. We incorporated the trained immunogenicity predictor by training the generator with the predicted immunogenicity score of the generated peptide sequences. We showed our generator can produce more immunogenic peptides after adding the predictor and can produce peptides that are similar to the epitopes shown in bladder cancerous cells.

1 Introduction

Tumor-specific antigens have been utilized in cancer vaccines, a form of cancer immunotherapy, to stimulate the production of tumor-specific T cells [1]. These antigens are encoded in the genome and are not present in normal cells, making them representative of aberrant proteins [2]. An epitope can be defined as a segment of an antigen that is generated through antigen processing [3]. The activation of the tumor-specific T cell response occurs when epitopes from class I or class II human leukocyte antigen (HLA) molecules are presented to T cell receptors (TCRs), which recognize antigens in the form of peptides [4]. The HLA class I molecule serves as the TCR ligand for CD8+ cytotoxic T lymphocytes (CTLs) and binds peptides consisting of 8-11 amino acids. On the other hand, the TCR ligand for CD4+ helper T cells is the HLA class II molecule, and the bound peptides typically consist of 15 amino acids [5].

Peptide-based cancer vaccines refer to cancer vaccines that employ polypeptides comprised of known or predicted tumor antigen epitopes [6]. Peptide-based vaccines are widely utilized in cancer vaccination practices and are designed to activate a tumor-specific cytotoxic T lymphocyte (CTL)

AI for New Drug Modalities at NeurIPS 2024.

response [7]. These vaccines consist of multiple epitopes, typically ranging from 8 to 11 amino acids in length [8]. Moreover, these peptides are commonly combined with a carrier protein to facilitate their recognition and processing by antigen-presenting cells (APCs), thereby triggering an immune response that involves CTLs [9].

To develop an effective peptide-based vaccine, it is crucial to ensure that the epitopes are recognizable by T cells, highly prevalent, and exclusive to tumor cells [10]. The process of identifying immunogenic epitopes begins by obtaining samples of both cancerous and normal cells through biopsy, followed by comparing their DNA sequences using techniques like Whole Exome Sequencing (WES) or Whole Genome Sequencing (WGS) to identify tumor-specific mutations [11]. Somatic variant callers can be employed to detect single nucleotide variants from the WES and WGS data, and subsequently, peptides containing the mutated regions are extracted using sliding windows from the varied protein sequence [12]. Epitope candidates can be identified using computational prediction tools, an approach that offers significant advantages in terms of time and cost when compared to traditional methods such as mass spectrometry techniques [13, 14, 15, 16].

Immunogenicity refers to the capacity of a substance to initiate an immune response, and the binding of a peptide to HLA molecules is essential for epitope immunogenicity [17]. [18] devised DeepHLApan, a recurrent neural network-based method that integrates binding affinity and immunogenicity details of peptide-HLA complexes to predict CD8+ T-cell epitopes. [19], on the other hand, utilized a convolutional neural network (CNN)-based method called DeepImmuno, where they utilized linear peptides of 9 to 10 amino acids and 4-digit class I HLA alleles as input to forecast the immunogenicity score of peptide-HLA pairs for T-cell immune responses. [20] introduced Seq2Neo, a CNN-based pipeline that leverages the binding affinity and transporters associated with antigen processing (TAP) transport efficiency of a given peptide-HLA pair to improve the prediction of immunogenicity scores for T-cell immune responses.

Generative Adversarial Networks (GANs) [21] are deep learning models that aim to understand and capture the distribution of training data, enabling the synthesis of samples from this learned distribution [22, 23]. GAN algorithms have found application in generating novel protein and peptide structures for use in drug screening and the discovery stage [24]. [25] developed a GAN equipped with a graph CNN that predicts solubility and toxicity from molecular descriptors, guiding the generator network to produce new small molecules with desired drug properties. [26] proposed a GAN architecture known as "RANC," which incorporates reinforcement learning to generate chemically diverse structures with desired features, with a focus on maintaining similar lengths of the SMILES string as their training data. [27] developed a semisupervised, guided, conditional, Wasserstein generative adversarial network capable of generating proteins with desired structure folds, while incorporating greater sequence diversity and novelty compared to conditional variational autoencoder designs. [19] employed a generator trained with a Wasserstein generative adversarial network with gradient penalty (WGAN-GP) to generate immunogenic epitopes binding to HLA-A*0201. They demonstrated that their generated peptides exhibited features and amino acid sequences similar to real epitopes within their training dataset.

As of March 2023, the Immune Epitope Database (IEDB) [28] reports only 24 experimentally tested linear bladder cancer epitopes that bind with HLA class I molecules, as identified by [29]. In contrast, the TSNAdb database offers 6234 bladder cancer neoepitopes, each featuring a single amino acid mutation, predicted to bind with HLA-A*0201 and having predicted binding affinity IC_{50} <500nM. We choose HLA-A*0201 as the binding target of the generated peptides since HLA-A*0201 binds with most of the bladder cancer neoepitopes in the TSNAdb database. We select bladder cancer neoepitopes as an illustrative example of our method. Since the neoepitopes from the TSNAdb database are not predicted to be immunogenic, we aim to design a goal-directed generator to provide potential immunogenic peptide sequences and increase the pool of peptides worthy of experimental testing.

In this study, we have developed a novel training method for a goal-directed generator aimed at generating immunogenic epitope sequences specific to bladder cancer. Our choice of GAN architecture is the WGAN-GP, which has been proven to provide higher training stability [30]. The critic network in our architecture is trained using bladder cancer epitope sequences that are exclusive to cancerous cells and have the ability to bind with HLA-A*0201. In our newly proposed training scheme, the generator is trained using the critic's output, which assigns a high value if the peptide exists in cancerous cells and a low value if it does not. Additionally, the generator is trained using

the output of a scorer, which is a CNN trained with the immunogenicity predictor output from DeepImmuno [19]. Our results demonstrate that by incorporating the scorer, the generator can produce a greater number of immunogenic peptides. Furthermore, these peptides exhibit a high degree of similarity to the bladder cancer epitopes found within the training dataset. The python code are provided at: `https://github.com/YenCheHsiao/Goal-directed-WGAN-GP`.

2 Methods

2.1 Datasets

The bladder cancer epitopes used in our study were sourced from the tumor-specific neoantigen database (TSNAdb) [31]. These epitopes, along with their corresponding HLA molecules, were predicted using NetMHCpan v4.0 [32]. NetMHCpan v4.0 assesses the binding affinity of potential epitopes within a protein sequence based on mass spectrometry eluted ligands and half-maximal inhibition (IC50) scores, with a threshold of $\text{IC}50 < 500\text{nM}$. Only epitopes containing a single mutated amino acid were selected for extraction. It is important to note that these epitopes have not been experimentally verified in other literature. The bladder cancer dataset within TSNAdb was utilized as the training dataset for our goal-directed WGAN-GP.

During the data cleaning process, we specifically selected bladder cancer epitope sequences predicted to bind with HLA-A*0201 and having a length of either 9 or 10 amino acids. Our resulting training dataset comprised a total of 6234 epitopes. For standardization, we employed the same coding strategy as in [19]. This involved padding 9-mer peptides to become 10-mers by joining the first five amino acids and the last four amino acids with a placeholder "-". Thus, the peptide sequences in our study are represented as a sequence consisting of a placeholder "-" and the 20 amino acid types: Alanine (A), Arginine (R), Asparagine (N), Aspartate (D), Cysteine (C), Glutamine (Q), Glutamate (E), Glycine (G), Histidine (H), Isoleucine (I), Leucine (L), Lysine (K), Methionine (M), Phenylalanine (F), Proline (P), Serine (S), Threonine (T), Tryptophan (W), Tyrosine (Y), and Valine (V). The amino acids and placeholder were converted into one-hot encoded matrices [33].

2.2 Goal-directed WGAN-GP

We implemented our goal-directed WGAN-GP using the architecture illustrated in Figure 1. This architecture comprises a peptide sequence generator, a critic, an immunogenicity predictor, and a scorer. The generator and critic are CNN-based models adapted from [19]. The critic is trained using the predicted bladder cancer epitopes obtained from TSNAdb and the generated epitopes, thereby guiding the training of the generator. Before being input to the critic, the predicted bladder cancer epitopes are encoded as a one-hot matrix, as depicted in Figure 7 (**a**) in Appendix B. Conversely, the output of the generator is a matrix of probabilities, which can be decoded as a peptide sequence, as shown in Figure 7 (**b**) in Appendix B. The peptide sequence generator tries to generate sequences that are closely similar to the predicted bladder cancer epitopes in the training dataset.

The immunogenicity predictor utilized is the Deepimmuno-CNN from [19]. This predictor takes a 10-mer generated peptide sequence, along with a 46-mer HLA-A*0201 sequence, as input, predicting their respective immunogenicities. The scorer is a CNN-based model that outputs the immunogenicity score. In contrast to the immunogenicity predictor, the scorer is trained during the training of the goal-directed WGAN-GP. The scorer takes the generated sequence and the sequence in the training dataset as the input and the output of the immunogenicity predictor given the scorer's input as the target value. Throughout the training, the placeholder '-' remains in the peptide sequence and is only removed when utilizing the trained generator to produce peptide sequences, as depicted in the procedure outlined in Figure 7 (**b**), step **(3)**.

Consider a generative network $G_\theta : \mathbb{R}^m \to \mathbb{R}^{pq}$ with a set of parameters $\theta \in \mathbb{R}$ and a regression model $P : \mathbb{R}^{pq} \to [0, 1]$. We aim to update the set of parameters θ such that the expected value $E_{G_\theta(z)\sim\mathbb{P}_G}[\hat{p}]$ is maximized, where $\hat{p} = P(G_\theta(z))$, $z \in \mathbb{R}^m$ is a random noise vector, and $\mathbb{P}_G$ is the generated data distribution.

The loss function of the critic D_ω is the same as in [30], which is

$$\begin{aligned} L_\omega =& E_{\tilde{x}\sim\mathbb{P}_G}[D_\omega(\tilde{x})] - E_{x\sim\mathbb{P}_r}[D_\omega(x)] \\ &+ \lambda E_{\hat{x}\sim\mathbb{P}_{\hat{x}}}[(\|\nabla_{\hat{x}} D_\omega(\hat{x})\|_2 - 1)^2], \end{aligned} \tag{1}$$

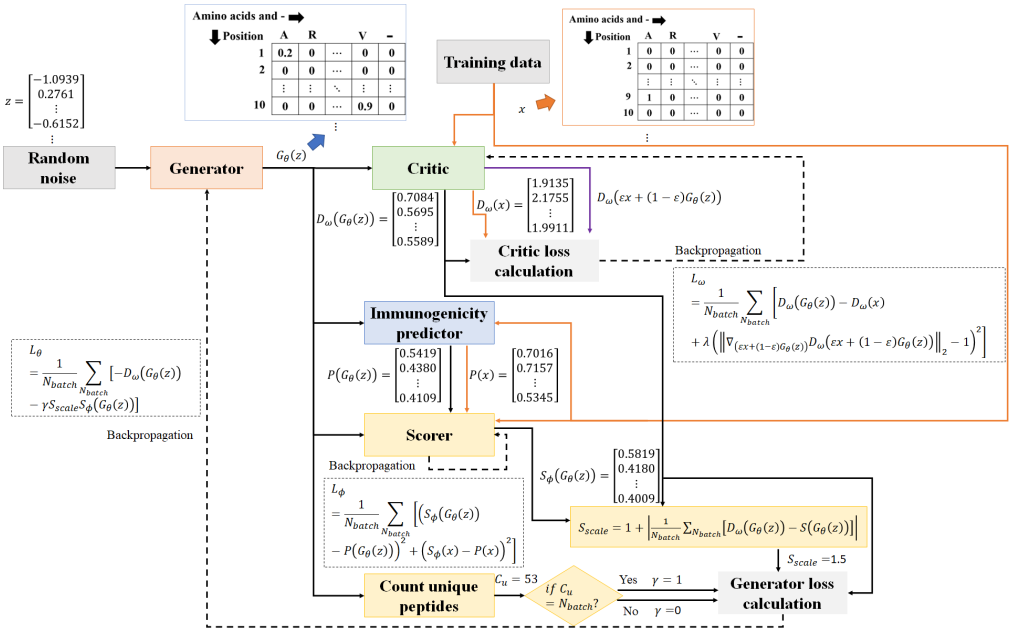

Figure 1: Training scheme of our goal-directed WGAN-GP. The training of our goal-directed Wasserstein Generative Adversarial Network with Gradient Penalty (WGAN-GP) involves incorporating an immunogenicity predictor from [19] to train a generator and a scorer. The objective of the scorer is to assist the generator in producing highly immunogenic peptides, while the critic aims to guide the generator in generating peptides with high similarity (though not identical) to those in the training dataset. The generator takes a set of N_{batch} random noise vectors z, each composed of 128 elements sampled from a zero-mean, unit-variance Gaussian distribution, as input. Implemented as a deep convolutional neural network, the generator produces N_{batch} one-probability matrices $G_\theta(z)$, which serve as input for the critic, immunogenicity predictor, and scorer. The critic's output is a vector of real values $D_\omega(G_\theta(z))$, the output of the immunogenicity predictor $P(G_\theta(z))$ is a vector of real values between 0 and 1, and the scorer's output is a vector with real values $S_\phi(G_\theta(z))$. The scorer is trained using the mean squared error between the output of the immunogenicity predictor and its own output, considering the generated one-probability matrices $G_\theta(z)$ and the one-hot encoded matrix of peptides in the training data x as input. The outputs of the scorer $S_\phi(G_\theta(z))$ and the output of the critic $D_\omega(G_\theta(z))$ are used to compute a scaling variable S_{scale}. The variable γ indicates whether the generated peptides are repeated in a batch ($\gamma = 0$ for repetitions, $\gamma = 1$ otherwise). The critic is trained using the standard critic loss in a WGAN-GP, while the generator is trained with a loss aiming to minimize the critic's output given the generated one-probability matrices $D_\omega(G_\theta(z))$and to minimize the scorer's output given the generated one-probability matrices $S_\phi(G_\theta(z))$ multiplied by the scaling variable S_{scale} and the variable γ.

where

$$\tilde{x} = G_\theta(z), \tag{2}$$

$$\hat{x} = \epsilon x + (1 - \epsilon)\tilde{x}, \tag{3}$$

$E_{x\sim\mathbb{P}}[f(x)]$ is the expected value of a function $f(x)$ with x sampled from the distribution $\mathbb{P}$, $\mathbb{P}_r$ and $\mathbb{P}_G$ are the training data distribution and the generated data distribution, respectively, $\mathbb{P}_{\hat{x}}$ is the distribution of the data sampled from the training and generated distribution as defined in (3), $G_\theta(\cdot)$ and $D_\omega(\cdot)$ is the function of the generator and the critic in WGAN-GP, λ is the penalty coefficient, $\|\cdot\|_2$ is the L-2 norm, $\epsilon \in [0, 1]$ is a random real number between zero and one, z is a random noise vector and each element is sampled from a normal distribution with zero mean and unit variance, θ is the weights in the generative network G_θ, and ω is the weights in the critic network D_ω.

The designed loss function of the generator G_θ is

$$L_\theta = -E_{\tilde{x}\sim\mathbb{P}_G}[D_\omega(\tilde{x})] - \gamma S_{scale} E_{\tilde{x}\sim\mathbb{P}_G}[S_\phi(\tilde{x})], \tag{4}$$

where

$$\gamma = \begin{cases} 1 & \text{if } n(A) = N_{batch} \\ 0 & \text{otherwise} \end{cases}, \tag{5}$$

A is a set composed of the generated one-probability matrix $G_\theta(z)$ in a batch, $n(A)$ is the cardinality of A (the number of elements in the set A), N_{batch} is the number of generated peptides in a batch,

$$S_{scale} = 1 + |E_{\tilde{x}\sim\mathbb{P}_G}[D_\omega(\tilde{x})] - E_{\tilde{x}\sim\mathbb{P}_G}[S_\phi(\tilde{x})]|, \tag{6}$$

$|\cdot|$ denotes as modulus, S_ϕ is the scorer, and ϕ is the weights in the scorer network S_ϕ.

The loss function of the scorer S_ϕ is defined as

$$L_\phi = E_{\tilde{x}\sim\mathbb{P}_G}[(S_\phi(\tilde{x}) - P(\tilde{x}))^2] + E_{x\sim\mathbb{P}_r}[(S_\phi(x) - P(x))^2], \tag{7}$$

where $P(\cdot) \in [0, 1]$ is the output of the immunogenicity predictor.

The variable S_{scale} in the loss function of the generator (4) is the addition of one and the modulus of the difference between the expected value of the critic output and the expected value of the scorer output given the generated one-probability matrices $G_\theta(z)$ as input. This term ensures that the expected value of the critic output will not dominate the generator's loss function by multiplying it with the expected value of the scorer's output.

The variable γ in the loss function of the generator (4) is used to force the generator to produce diverse peptide sequences in the training stage. If $\gamma = 1$, (4) becomes a weighted sum of the expected value of the critic output and the expected value of the scorer output given the generated one-probability matrices $G_\theta(z)$ as input. If $\gamma = 0$, the training of the goal-directed WGAN-GP is the vanilla WGAN-GP in [30].

The architecture of the generator, the critic, the scorer, and the residual block in these three networks can be found in Table 1, Table 2, Table 3, and Table 4 in Appendix C, respectively.

2.3 Training of the goal-directed WGAN-GP

In this paper, the batch size is 64 ($N_{batch} = 64$). $z \in \mathbb{R}^{128}$ is defined to be a random noise vector with 128 elements, and each element is sampled from a normal distribution with zero mean and unit variance. The generative network, the critic network, and the scorer network are defined as $G_\theta : \mathbb{R}^{128} \to \mathbb{R}^{10\times21}$, $D_\omega : \mathbb{R}^{10\times21} \to \mathbb{R}$, and $S_\phi : \mathbb{R}^{10\times21} \to \mathbb{R}$, respectively. For each epoch, the critic is trained 97 times, the generator is trained 10 times, and the scorer is trained 97 times. The algorithm of our goal-directed WGAN-GP is shown in Algorithm 1 in Appendix A.

The model training is conducted on Windows 10, version 22H2, utilizing an Intel(R) Core(TM) i7-10875H CPU @ 2.30GHz with 8 logical CPUs. The training process is executed on the CPU, employing Python 3.8.0 as the coding language, and the model is constructed using PyTorch 2.0.0.

3 Experiments

To validate the efficacy of our proposed generator training scheme, we adopted the same architecture for both the generative and critic networks as outlined in [19]. We then compared the peptides generated by the generator trained with and without our devised training scheme. For assessing immunogenicity, we utilized the CNN-based immunogenicity predictor, DeepImmunoCNN, from [19]. The CNN was retrained using the "remove0123_sample100.csv" data file provided in their source code. The immunogenicity predictor takes a 10-mer peptide sequence and a 4-digit encoded HLA sequence as inputs, predicting their real-value immunogenicity score within the range of $[0, 1]$. In this analysis, we specifically considered HLA-A*0201 as the HLA sequence input, defining an immunogenic peptide as one with an immunogenicity score exceeding 0.5.

3.1 Immunogenicity

We conducted a comparative analysis between the generator trained with our devised training scheme and the generator trained using the architecture from [19]. Our goal-directed WGAN-GP was trained on bladder cancer epitope data sourced from TSNAdb [31]. We compared the number of immunogenic epitopes generated by our approach with the results presented in [19]. The WGAN-GP was separately

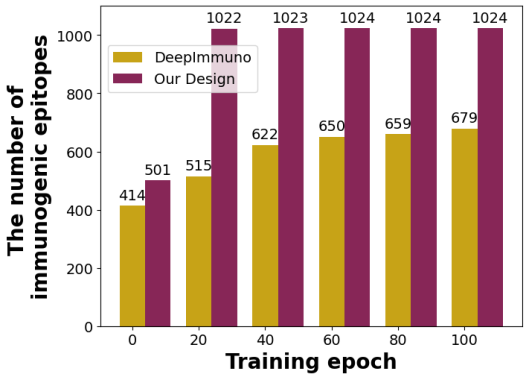

Figure 2: The comparison of the generator trained by our goal-directed WGAN-GP to the Deepimmuno generator [19].

trained for 0, 20, 40, 60, 80, and 100 epochs, producing 1024 peptides from each generator trained during these epochs. To ensure consistent comparison, we removed peptide sequences containing more than 2 placeholders ('-'), resulting in 957, 1024, 1024, 1024 , 1024, and 1024 peptides from the generators trained with 0, 20, 40, 60, 80, and 100 epochs, respectively. After inputting these peptides into the immunogenicity predictor, we counted the number of immunogenic peptides with a score higher than 0.5. The results are depicted in Figure 2, indicating that our designed training scheme with TSNAdb [31] yields more immunogenic epitopes compared to the generator from [19].

We conducted two training sessions: the first involved training the WGAN-GP without incorporating the immunogenicity predictor, and the second involved our goal-directed WGAN-GP trained, both training for 1000 epochs and using the TSNAdb dataset [31]. The averaged immunogenicity scores of the generated peptide sequences used to train the generator are presented in Figure 3, where the orange and blue solid lines represent the averaged scores calculated using

$$P_{epoch}^{(s)} = \frac{1}{10}\sum_{k=1}^{10} P_{mean}^{(k)}, \tag{8}$$

where $k = 1, 2, \cdots, 10$ is the number of training of the generator in an epoch, $s = 1, 2, \cdots, N$ is the number of epoch, $N = 1000$ is the maximum iteration number, and

$$P_{mean}^{(k)} = \frac{1}{N_{batch}} \sum_{j=1}^{N_{batch}} P(G_\theta(z^{(j)})), \tag{9}$$

Additionally, the cumulative averaged immunogenicity score is defined as

$$P_{cumulate}^{(s)} = \frac{1}{s}\sum_{t=1}^{s} P_{epoch}^{(t)}. \tag{10}$$

The cumulative averaged immunogenicity score in (10) is depicted as a brown dashed line and a purple dotted line in Fig.3. Notably, the immunogenicity scores of sequences generated from our goal-directed WGAN-GP continue to increase, reaching approximately 0.93. Conversely, sequences generated without the immunogenicity predictor in the GAN only reach around 0.6 and show no further increase. The new Fig.3 (**b**) demonstrates our method's superior maximum immunogenicity compared to the vanilla WGAN-GP. To further validate that our goal-directed WGAN-GP generates peptides with higher immunogenicity scores compared to the WGAN-GP without the immunogenicity predictor, we employed the generator trained for 1000 epochs from both schemes to produce 10000 peptide sequences. Considering only 9 or 10-mers, the resulting number of peptide sequences is 9998 for our goal-directed WGAN-GP and 9830 for the WGAN-GP. In Fig.4, the immunogenicity scores of the peptide sequences from both models are presented in dots and box plots. Each orange or blue dot represents a peptide sequence from our goal-directed WGAN-GP or WGAN-GP, respectively. The band in the box indicates the median, while the lower and upper bounds of the box represent the first (25th percentile) and the third (75th percentile) quartiles, respectively. The whiskers indicate $\pm 1.5\times$

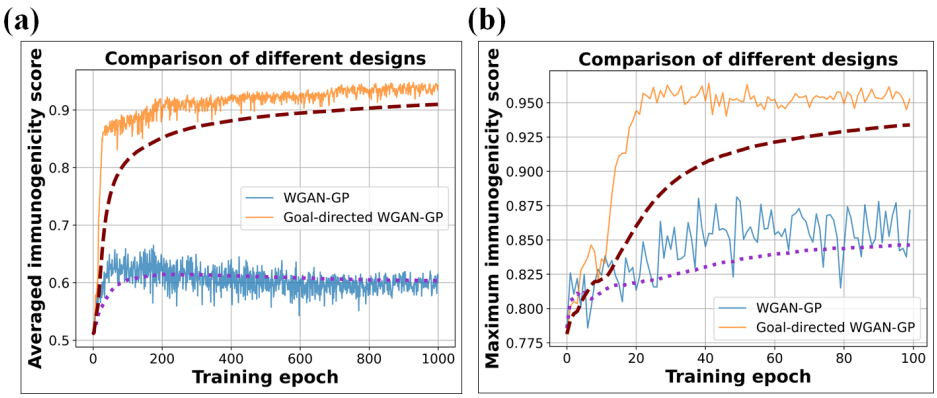

Figure 3: (**a**) The averaged immunogenicity score of the peptide sequences generated from our designed goal-directed WGAN-GP (orange line) compared to the WGAN-GP without the immunogenicity predictor (blue line) through training and their cumulative average score in the brown dashed line and the purple dotted line, respectively. (**b**) The maximum immunogenicity score of the peptide sequences generated from our designed goal-directed WGAN-GP (orange line) compared to the WGAN-GP without the immunogenicity predictor (blue line) through training and their cumulative average score in the brown dashed line and the purple dotted line, respectively.

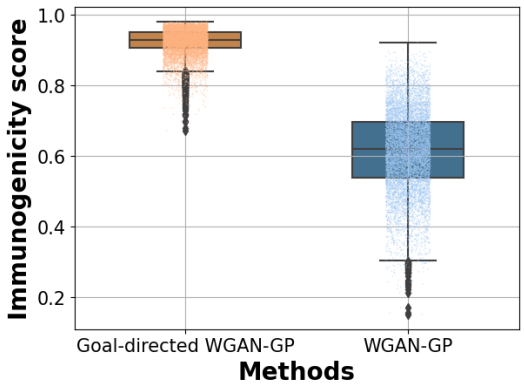

Figure 4: The dots and box plot showing the median immunogenicity score of the 9998 peptides produced from the generator trained with our goal-directed WGAN-GP is about 0.95, and it is higher than the score of the 9830 peptides generated from the vanilla WGAN-GP with about 0.6.

the interquartile range [34]. The median immunogenicity score in our goal-directed WGAN-GP is approximately 0.95, significantly higher than the median in the WGAN-GP, which is around 0.62. Our method exhibits a narrower interquartile range, focusing more peptides at the median. The outlier minimum closely matches the vanilla WGAN-GP's median, while the maximum aligns with our method's minimum, demonstrating our generator's effectiveness in producing immunogenic peptides. These results underscore the effectiveness of our designed training method in enhancing the generator's capability to produce peptide sequences with higher immunogenicity compared to the vanilla training scheme for the WGAN-GP.

The additional experiment results of the prediction of immunogenicity score using DeepHLApan [35] and IEDB [28] are in Appendix D. In addition, the evaluation of our method using brain cancer epitopes are in Appendix E.

3.2 Binding affinity

The binding affinity between the generated peptides and HLA-A0201 is quantified by the half-maximal inhibition (IC_{50}) values, representing the concentration of the test peptide resulting in 50%

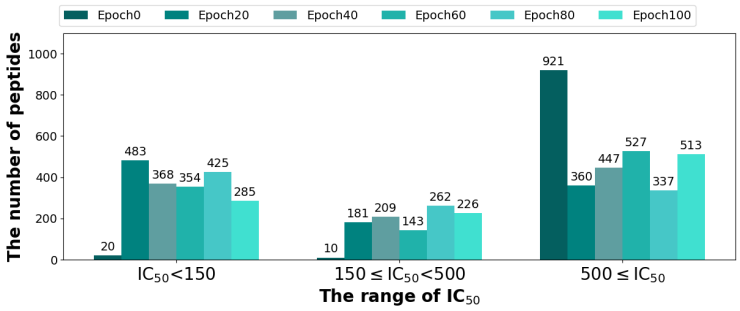

Figure 5: The number of the immunogenic peptides compared to the number of peptides that bond to HLA-A*0201 with different ranges of the binding affinity IC_{50}.

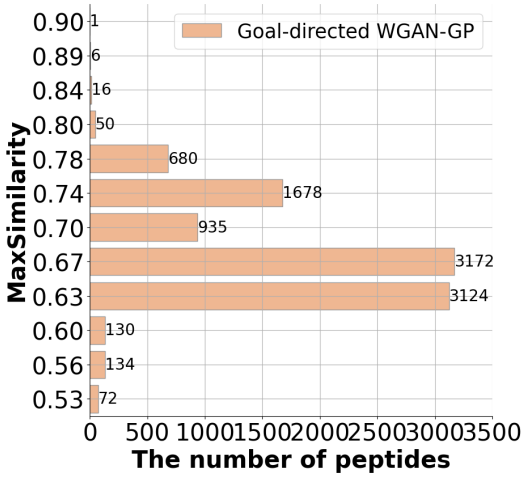

Figure 6: The histogram of the maximum similarity with the corresponding number of the peptides generated from our goal-directed WGAN-GP and the WGAN-GP.

inhibition of the binding of a probe peptide [36]. In the training dataset, the bladder cancer epitopes are predicted to have a binding affinity smaller than 500nM using NetMHCpan v4.0 [32] by [31]. To investigate whether the generated peptides also exhibit a strong binding affinity ($<$ 500nM), we employed the binding affinity prediction tool NetMHCpan 4.1 [35] to predict the IC_{50} values. Peptides with a predicted binding affinity IC_{50} below 500nM are considered binders to HLA-A*0201 [37].

The predicted binding affinity of all generated peptides across different epochs is illustrated in Figure 5. The count of peptides with binding affinity $IC_{50} < 500$ nM rises from 30 to 664 as the training epoch progresses from 0 to 20. Subsequently, as the epoch advances from 20 to 40, the count decreases from 664 to 577. However, from epoch 40 to 100, the count of peptides with binding affinity $IC_{50} < 500$ nM increases again, reaching 798. These observations suggest that our training scheme may contribute to the reduction of IC_{50} values over the course of training.

3.3 Similarity

We investigate whether increasing the training epochs can enhance the number of peptides similar to the neoepitopes in TSNAdb [31]. Similarity, as computed by [19], is determined using [19]

$$Similarity = \frac{2n}{l_{p_g} + l_{p_r}}, \tag{11}$$

where n represents the number of matches between the generated peptide sequence p_g and the peptide sequence p_r in the training dataset, l_{p_g} denotes the length of the generated peptide sequence, l_{p_r}

stands for the length of the peptide sequence in the training dataset, and $Similarity$ is the computed similarity.

We calculate the similarity between each generated peptide and all peptides in the TSNAdb bladder cancer dataset. The number of each maximum similarity value is counted, aiming to assess whether increasing the training epoch improves the similarity between the generated peptide and bladder cancer epitopes in the TSNAdb dataset.

The maximum similarity of peptides generated from the generator trained by our goal-directed WGAN-GP is compared with the generator trained by the vanilla WGAN-GP, both trained for 1000 epochs using the TSNAdb dataset [31]. 10000 peptides are generated from the two generators, and sequences with more than 2 placeholders '-' are removed. The resulting number of peptide sequences is 10000 for our goal-directed WGAN-GP and 9837 for the WGAN-GP. In Figure 6, we present the number of peptides with respect to each calculated maximum similarity, rounded off to the second decimal place. The WGAN-GP exhibits a higher number of peptides with maximum similarity exceeding 0.78 compared to our goal-directed WGAN-GP. However, it also demonstrates that peptides produced by our goal-directed WGAN-GP can exhibit similar sequences to those in the TSNAdb bladder cancer dataset [31].

The generated set of 10,000 peptides from our goal-directed generator contains some repeated sequences. Upon removing the duplicates, we obtained 8,156 unique peptides, in contrast to the generator trained by the vanilla WGAN-GP, which yields 9,827 unique peptides. The most frequent sequence, "FLEPGIPRL," occurs 18 times in the 10,000 generated peptides from our goal-directed generator. We think that the occurrence of repeated peptides from our generator is likely due to the scarcity of highly predicted immunogenic peptides in the training dataset. This increases the likelihood of repetition compared to vanilla WGAN-GP, which directly trains using the entire dataset.

4 Conclusion

The prediction algorithms for immunogenic neoantigens face challenges due to the limited number of proven immunogenic neoantigens [38]. Experimental validation of immunogenicity is a crucial step in enhancing the chances of successful immunotherapy [39]. However, the preparation time for a personalized peptide-based cancer vaccine is lengthy, taking at least 3 months from sample collection to vaccine production [40]. Our method aims to increase the pool of predicted immunogenic neoepitope candidates, enabling the generation of more peptides worthy of testing. These peptides can undergo experimental evaluation for their binding affinity to HLA-A*0201 or their ability to elicit cytotoxic T-cell response.

The HLA class I–presented peptides are subject to HLA allotype restriction, potentially excluding patients with incompatible HLA class I allotypes [10]. In cases where the peptide-HLA complex presented in the patient's tumor cannot be predicted to have high immunogenicity, our goal-directed WGAN-GP can be employed to generate hypothesized immunogenic peptides. This can be achieved by training our goal-directed WGAN-GP with a dataset containing all predicted neoepitope sequences that are predicted to exhibit high binding affinity with the specific four-digit HLA, as retrieved from the TSNAdb database [31].

Our findings demonstrate that the generator, trained using our designed training scheme, is capable of generating 9-mer and 10-mer peptide sequences with high immunogenicity. Additionally, our analysis reveals that the generator can generate peptide sequences with a strong binding affinity (IC_{50} $<$500nM). We anticipate that our designed training scheme for the WGAN-GP can serve as a valuable tool for facilitating the design of peptide-based vaccines and can find applicability in diverse scenarios requiring a goal-directed WGAN-GP.

References

[1] Stefan Stevanovic. Identification of tumour-associated t-cell epitopes for vaccine development. *Nature Reviews Cancer*, 2(7):514–514, 2002.

[2] Matthew M Gubin, Maxim N Artyomov, Elaine R Mardis, Robert D Schreiber, et al. Tumor neoantigens: building a framework for personalized cancer immunotherapy. *The Journal of clinical investigation*, 125(9):3413–3421, 2015.

[3] John Sidney, Bjoern Peters, and Alessandro Sette. Epitope prediction and identification-adaptive t cell responses in humans. In *Seminars in immunology*, volume 50, page 101418. Elsevier, 2020.

[4] Malte Mohme and Marian Christoph Neidert. Tumor-specific t cell activation in malignant brain tumors. *Frontiers in immunology*, 11:205, 2020.

[5] Malcolm JW Sim and Peter D Sun. T cell recognition of tumor neoantigens and insights into t cell immunotherapy. *Frontiers in Immunology*, 13, 2022.

[6] Jian Liu, Minyang Fu, Manni Wang, Dandan Wan, Yuquan Wei, and Xiawei Wei. Cancer vaccines as promising immuno-therapeutics: Platforms and current progress. *Journal of Hematology & Oncology*, 15(1):28, 2022.

[7] Lisa H Butterfield. Cancer vaccines. *Bmj*, 350, 2015.

[8] Wensi Liu, Haichao Tang, Luanfeng Li, Xiangyi Wang, Zhaojin Yu, and Jianping Li. Peptide-based therapeutic cancer vaccine: current trends in clinical application. *Cell Proliferation*, 54(5):e13025, 2021.

[9] Noraini Abd-Aziz and Chit Laa Poh. Development of peptide-based vaccines for cancer. *Journal of Oncology*, 2022, 2022.

[10] Annika Nelde, Hans-Georg Rammensee, and Juliane S Walz. The peptide vaccine of the future. *Molecular & Cellular Proteomics*, 20, 2021.

[11] Guilhem Richard, Michael F Princiotta, Dominique Bridon, William D Martin, Gary D Steinberg, and Anne S De Groot. Neoantigen-based personalized cancer vaccines: the emergence of precision cancer immunotherapy. *Expert Review of Vaccines*, 21(2):173–184, 2022.

[12] Megan M Richters, Huiming Xia, Katie M Campbell, William E Gillanders, Obi L Griffith, and Malachi Griffith. Best practices for bioinformatic characterization of neoantigens for clinical utility. *Genome medicine*, 11(1):1–21, 2019.

[13] Claudia Lemmel and Stefan Stevanović. The use of hplc-ms in t-cell epitope identification. *Methods*, 29(3):248–259, 2003.

[14] Xiaotong Chen, Ju Yang, Lifeng Wang, and Baorui Liu. Personalized neoantigen vaccination with synthetic long peptides: recent advances and future perspectives. *Theranostics*, 10(13):6011, 2020.

[15] Luca Hensen, Patricia T Illing, Louise C Rowntree, Jane Davies, Adrian Miller, Steven YC Tong, Jennifer R Habel, Carolien E van de Sandt, Katie L Flanagan, Anthony W Purcell, et al. T cell epitope discovery in the context of distinct and unique indigenous hla profiles. *Frontiers in Immunology*, 13, 2022.

[16] Sepideh Parvizpour, Mohammad M Pourseif, Jafar Razmara, Mohammad A Rafi, and Yadollah Omidi. Epitope-based vaccine design: a comprehensive overview of bioinformatics approaches. *Drug Discovery Today*, 25(6):1034–1042, 2020.

[17] Dinler A Antunes, Jayvee R Abella, Didier Devaurs, Maurício M Rigo, and Lydia E Kavraki. Structure-based methods for binding mode and binding affinity prediction for peptide-mhc complexes. *Current topics in medicinal chemistry*, 18(26):2239–2255, 2018.

[18] Jingcheng Wu, Wenzhe Wang, Jiucheng Zhang, Binbin Zhou, Wenyi Zhao, Zhixi Su, Xun Gu, Jian Wu, Zhan Zhou, and Shuqing Chen. Deephlapan: a deep learning approach for neoantigen prediction considering both hla-peptide binding and immunogenicity. *Frontiers in immunology*, 10:2559, 2019.

[19] Guangyuan Li, Balaji Iyer, VB Surya Prasath, Yizhao Ni, and Nathan Salomonis. Deepimmuno: deep learning-empowered prediction and generation of immunogenic peptides for t-cell immunity. *Briefings in bioinformatics*, 22(6):bbab160, 2021.

[20] Kaixuan Diao, Jing Chen, Tao Wu, Xuan Wang, Guangshuai Wang, Xiaoqin Sun, Xiangyu Zhao, Chenxu Wu, Jinyu Wang, Huizi Yao, et al. Seq2neo: A comprehensive pipeline for cancer neoantigen immunogenicity prediction. *International Journal of Molecular Sciences*, 23(19):11624, 2022.

[21] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[22] Liang Gonog and Yimin Zhou. A review: generative adversarial networks. In *2019 14th IEEE conference on industrial electronics and applications (ICIEA)*, pages 505–510. IEEE, 2019.

[23] Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A Bharath. Generative adversarial networks: An overview. *IEEE signal processing magazine*, 35(1):53–65, 2018.

[24] Eugene Lin, Chieh-Hsin Lin, and Hsien-Yuan Lane. De novo peptide and protein design using generative adversarial networks: an update. *Journal of Chemical Information and Modeling*, 62(4):761–774, 2022.

[25] Abhishek Dutta. Deep graph generation of small molecules guided by drug-likeness. In *IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology*, pages 1–2. IEEE, 2022.

[26] Evgeny Putin, Arip Asadulaev, Yan Ivanenkov, Vladimir Aladinskiy, Benjamin Sanchez-Lengeling, Alán Aspuru-Guzik, and Alex Zhavoronkov. Reinforced adversarial neural computer for de novo molecular design. *Journal of chemical information and modeling*, 58(6):1194–1204, 2018.

[27] Mostafa Karimi, Shaowen Zhu, Yue Cao, and Yang Shen. De novo protein design for novel folds using guided conditional wasserstein generative adversarial networks. *Journal of chemical information and modeling*, 60(12):5667–5681, 2020.

[28] Randi Vita, Swapnil Mahajan, James A Overton, Sandeep Kumar Dhanda, Sheridan Martini, Jason R Cantrell, Daniel K Wheeler, Alessandro Sette, and Bjoern Peters. The immune epitope database (iedb): 2018 update. *Nucleic acids research*, 47(D1):D339–D343, 2019.

[29] Chen Wang, Yu Ding, Yuanyong Liu, Qingchen Zhang, Shiqiang Xu, Liliang Xia, Huangqi Duan, Shujun Wang, Ping Ji, Weiren Huang, et al. Identification of mutated peptides in bladder cancer from exomic sequencing data reveals negative correlation between mutation-specific immunoreactivity and inflammation. *Frontiers in Immunology*, 11:576603, 2020.

[30] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. *Advances in neural information processing systems*, 30, 2017.

[31] Jingcheng Wu, Wenyi Zhao, Binbin Zhou, Zhixi Su, Xun Gu, Zhan Zhou, and Shuqing Chen. Tsnadb: a database for tumor-specific neoantigens from immunogenomics data analysis. *Genomics, proteomics & bioinformatics*, 16(4):276–282, 2018.

[32] Vanessa Jurtz, Sinu Paul, Massimo Andreatta, Paolo Marcatili, Bjoern Peters, and Morten Nielsen. Netmhcpan-4.0: improved peptide–mhc class i interaction predictions integrating eluted ligand and peptide binding affinity data. *The Journal of Immunology*, 199(9):3360–3368, 2017.

[33] Limin Jiang, Jijun Tang, Fei Guo, and Yan Guo. Prediction of major histocompatibility complex binding with bilateral and variable long short term memory networks. *Biology*, 11(6):848, 2022.

[34] Tim Rädsch, Annika Reinke, Vivienn Weru, Minu D Tizabi, Nicholas Schreck, A Emre Kavur, Bünyamin Pekdemir, Tobias Roß, Annette Kopp-Schneider, and Lena Maier-Hein. Labelling instructions matter in biomedical image analysis. *Nature Machine Intelligence*, pages 1–11, 2023.

[35] Birkir Reynisson, Bruno Alvarez, Sinu Paul, Bjoern Peters, and Morten Nielsen. Netmhcpan-4.1 and netmhciipan-4.0: improved predictions of mhc antigen presentation by concurrent motif deconvolution and integration of ms mhc eluted ligand data. *Nucleic acids research*, 48(W1):W449–W454, 2020.

[36] Mollie M Jurewicz, Richard A Willis, Vasanthi Ramachandiran, John D Altman, and Lawrence J Stern. Mhc-i peptide binding activity assessed by exchange after cleavage of peptide covalently linked to β2-microglobulin. *Analytical biochemistry*, 584:113328, 2019.

[37] Claus Lundegaard, Kasper Lamberth, Mikkel Harndahl, Søren Buus, Ole Lund, and Morten Nielsen. Netmhc-3.0: accurate web accessible predictions of human, mouse and monkey mhc class i affinities for peptides of length 8–11. *Nucleic acids research*, 36(suppl_2):W509–W512, 2008.

[38] Ravi Chand Bollineni, Trung T Tran, Fridtjof Lund-Johansen, and Johanna Olweus. Chasing neoantigens; invite naïve t cells to the party. *Current Opinion in Immunology*, 75:102172, 2022.

[39] Antonella Vitiello and Maurizio Zanetti. Neoantigen prediction and the need for validation. *Nature biotechnology*, 35(9):815–817, 2017.

[40] Minjun Ma, Jingwen Liu, Shenghang Jin, and Lan Wang. Development of tumour peptide vaccines: From universalization to personalization. *Scandinavian journal of immunology*, 91(6):e12875, 2020.

A Proposed algorithm for Goal-directed WGAN-GP

Algorithm 1 The training procedure of the Goal-directed WGAN-GP. The value for the parameters are $k = 1$, $\lambda = 10$, $n_{critic} = 10$, $N_{data} = 6232$, $N_{batch} = 64$, $\alpha = 10^{-4}$, $\beta_1 = 0.5$, and $\beta_1 = 0.9$.

Require: A maximum iteration number N, the initial weights ω_0, θ_0, and ϕ_0 for the critic D_ω, the generator G_θ, and the scorer S_ϕ, respectively.
$n_{split} \leftarrow \lfloor \frac{N_{data}}{N_{batch}} \rfloor$
while $k \leq N$ **do**
  Randomly split the training dataset with data distribution $\mathbb{P}_r$ to n_{split} data $\{\mathbb{P}_r^{(1)}, \mathbb{P}_r^{(2)}, \cdots, \mathbb{P}_r^{(n_{split})}\}$
  for $i = 1, \cdots, n_{split}$ **do**
    Sample a batch of noise vectors $\{z^{(j)}\}_{j=1}^{N_{batch}} \sim N(0, 1)$.
    for $j = 1, \cdots, N_{batch}$ **do**
      Sample real data $x \sim \mathbb{P}_r^{(i)}$ without replacement, and a random number $\epsilon \in U[0, 1]$.
      $\tilde{x} \leftarrow G_\theta(z^{(j)})$
      $L_\phi^{(j)} \leftarrow \frac{1}{N_{batch}}((S_\phi(\tilde{x}) - P(\tilde{x}))^2 + (S_\phi(x) - P(x))^2)$
      $\hat{x} \leftarrow \epsilon x + (1 - \epsilon)\tilde{x}$
      $L_\omega^{(j)} \leftarrow \frac{1}{N_{batch}}(-D_\omega(x) + D_\omega(\tilde{x}) + \lambda(\|\nabla_{\hat{x}} D_\omega(\hat{x})\|_2 - 1)^2)$
    end for
    $\phi \leftarrow \text{Adam}(\nabla_\phi \sum_{j=1}^{N_{batch}} L_\phi^{(j)}, \phi, \alpha, \beta_1, \beta_2)$
    $\omega \leftarrow \text{Adam}(\nabla_\omega \sum_{j=1}^{N_{batch}} L_\omega^{(j)}, \omega, \alpha, \beta_1, \beta_2)$
    if $(i \bmod n_{critic}) = 0$ **then**
      Sample a batch of noise vectors $\{z^{(j)}\}_{j=1}^{N_{batch}} \sim N(0, 1)$.
      $A \leftarrow \emptyset$
      for $j = 1, \cdots, N_{batch}$ **do**
        Add $G_\theta(z^{(j)})$ to a set A
        $S_{scale}^{(j)} \leftarrow \frac{1}{N_{batch}}(1 + |D_\omega(G_\theta(z^{(j)})) - S_\phi(G_\theta(z^{(j)}))|)$
      end for
      if $n(A) = N_{batch}$ **then**
        $\gamma \leftarrow 1$
      else
        $\gamma \leftarrow 0$
      end if
      $L_\theta \leftarrow \frac{1}{N_{batch}} \sum_{j=1}^{N_{batch}} (-D_\omega(G_\theta(z^{(j)}) - \gamma S_{scale} S_\phi(G_\theta(z^{(j)}))$
      $\theta \leftarrow \text{Adam}(\nabla_\theta L_\theta, \theta, \alpha, \beta_1, \beta_2)$
    end if
  end for
  $k := k + 1$
end while

B Encoding and decoding of peptide sequences

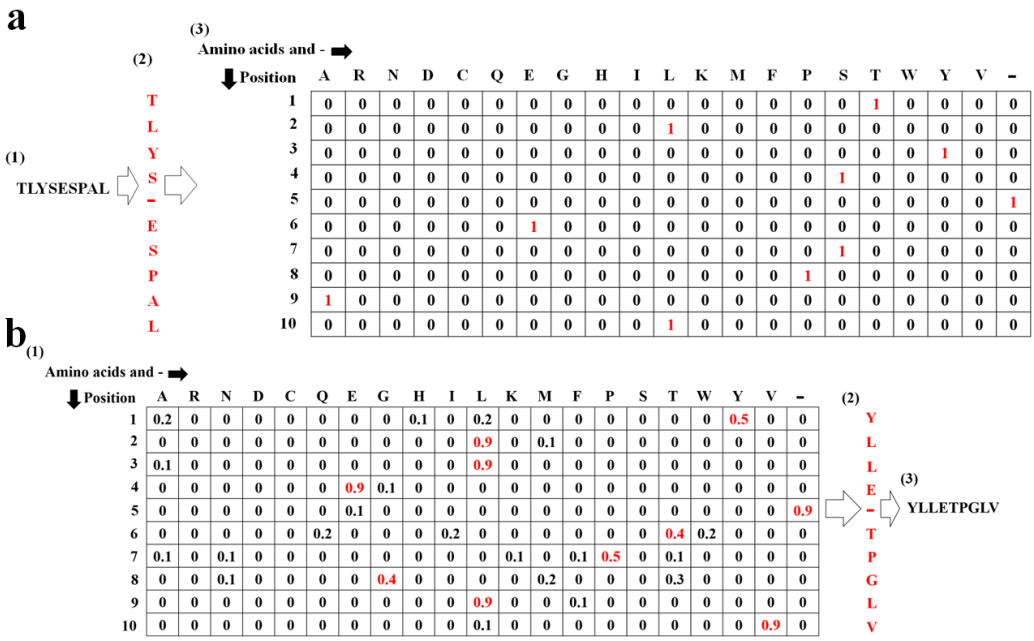

Figure 7: The encoding and the decoding methods in our goal-directed Wasserstein Generative Adversarial Network with Gradient Penalty (WGAN-GP) adopted from [19]. In (**a**), the peptide is encoded from a peptide sequence to a one-hot matrix x. **(1)** A 9-mer peptide sequence, such as TLYSESPAL, is prepared before input to the critic. **(2)** If the peptide sequence has only 9 amino acids, a placeholder '-' is inserted at the fifth position to extend the sequence to a length of 10. **(3)** The encoded peptide is represented as a one-hot matrix, where each row corresponds to a position in the peptide sequence, and columns represent amino acids along with a placeholder '-'. The element in the one-hot matrix is 1 if the corresponding position contains the amino acid or the placeholder at the corresponding column. For example, 'T' is at position 1, so the first row of the one-hot matrix will have 1 at the 17th column, corresponding to the character 'T', and 0 in the other columns. In (**b**), the peptide is decoded through the generated one-probability matrix $G_\theta(z)$. **(1)** The output of the generator $G_\theta(z)$ is a one-probability matrix, where each row represents the position of a peptide sequence, and columns represent amino acids along with a placeholder '-'. Each element in $G_\theta(z)$ is the probability that the position corresponding to the row contains the amino acid or the placeholder. **(2)** A peptide sequence is decoded by selecting the amino acid or the placeholder with the highest probability for each row. For instance, if 0.5 is the highest value in the first row, the first character in the decoded sequence is 'Y', corresponding to the column with the value 0.5 in the first row. **(3)** If the peptide sequence contains a placeholder, it is removed to form a peptide sequence shorter than 10.

C Detailed description of the network architecture

We present the details of the generator, the critic, the scorer, and the residual block in Table 1, Table 2, Table 3, and Table 4, respectively.

In the residual block shown in Table 4, the residual connection is defined to be

$$z_{res} = x_{res} + 0.3y_{res}, \tag{12}$$

where x_{res} is the input data in a residual block, y_{res} is the output from the fifth layer (convolution 1D) in the residual block, and z_{res} is the output of the residual connection.

We denote the output matrix of the Convolution 1D in the residual block as a two-dimensional matrix Y^{res}. The element at the i-th row and the j-th column of Y^{res} is denoted Y^{res}_{ij}. The input

Table 1: Details of the generator network adopted from [19]

Layer	Type	Kernel size	Filter	Stride	Padding	Output shape	Number of parameters
1	Input	-	-	-	-	128	-
2	Fully Connected	-	-	-	-	1280	165120
3	Reshape	-	-	-	-	128×10	-
4	Residual block	-	-	-	-	128×10	98560
5	Residual block	-	-	-	-	128×10	98560
6	Residual block	-	-	-	-	128×10	98560
7	Residual block	-	-	-	-	128×10	98560
8	Residual block	-	-	-	-	128×10	98560
9	Convolution 1D	1	21	1	no	21×10	2709
10	Transpose	-	-	-	-	10×21	-
11	Gumbel-Softmax	-	-	-	-	10×21	-

Table 2: Details of the critic network adopted from [19]

Layer	Type	Kernel size	Filter	Stride	Padding	Output shape	Number of parameters
1	Input	-	-	-	-	10×21	-
2	Transpose	-	-	-	-	21×10	-
3	Convolution 1D	1	128	1	no	128×10	2816
4	Residual block	-	-	-	-	128×10	98560
5	Residual block	-	-	-	-	128×10	98560
6	Residual block	-	-	-	-	128×10	98560
7	Residual block	-	-	-	-	128×10	98560
8	Residual block	-	-	-	-	128×10	98560
9	Reshape	-	-	-	-	1280	-
10	Fully Connected	-	-	-	-	1	1281

matrix of the Convolution 1D in the residual block is denoted as X^{res}. In the residual block, the dimension of the input to the Convolution 1D is defined to be $X^{res} \in \mathbb{R}^{r_X^{res} \times c_X^{res}}$, where $r_X^{res} = 128$ and $c_X^{res} = 10$ are the number of rows and columns of the input matrix, respectively. The input matrix with one zero padding is defined as $X^{pad} = [\mathbf{0}^{res} \quad X^{res} \quad \mathbf{0}^{res}] \in \mathbb{R}^{r_X^{res} \times (c_X^{res}+2)}$, where $\mathbf{0}^{res} = [0 \quad 0 \quad \cdots \quad 0]^T \in \mathbb{R}^{r_X^{res} \times 1}$ is an all zero column vector with dimension r_X^{res}. The dimension of the output matrix after the Convolution 1D will be $Y^{res} \in \mathbb{R}^{f_{res} \times c_X^{res}}$, where $f_{res} = 128$ is the number of kernels. Each element in the output matrix of the Convolution 1D layer in the residual block is computed by

$$Y_{i,j}^{res} = \sum_{m=1}^{r_X^{res}} \sum_{n=1}^{c_w^{res}} (k_{m,n}^{res,i} \cdot X_{m,n+j}^{pad}) + b^{res,i}, \tag{13}$$

where $1 \leq i \leq f_{res}$, $1 \leq j \leq c_X^{res}$, $w^{res,i} \in \mathbb{R}^{r_X \times c_w^{res}}$ is the i-th convolutional kernel, $c_w^{res} = 3$ is the number of columns in the kernel, and $b^{res,i} \in \mathbb{R}$ is the bias for the i-th kernel.

For The Convolution 1D in the 9-th layer of the generator, $X^{gen} \in \mathbb{R}^{r_X^{gen} \times c_X^{gen}}$ is denoted as the input matrix, where $r_X^{gen} = 128$ and $c_X^{gen} = 10$ is the number of rows and columns of the input matrix, respectively. The output matrix will be $Y^{gen} \in \mathbb{R}^{f_{gen} \times c_X^{gen}}$, where $f_{gen} = 21$ is defined to be the number of kernels for this Convolution 1D. The Convolution 1D in the 9-th layer of the generator is defined to be

$$Y_{i,j}^{gen} = \sum_{m=1}^{r_X} \sum_{n=1}^{c_w^{gen}} (k_{m,n}^{gen,i} \cdot X_{m,n+j}^{gen}) + b^{gen,i}, \tag{14}$$

where $1 \leq i \leq f_{gen}$, $1 \leq j \leq c_X^{gen}$, $w^{gen,i} \in \mathbb{R}^{r_X \times c_w^{gen}}$ is the i-th convolutional kernel, $c_w^{gen} = 1$ is the number of columns in the kernel, and $b^{gen,i} \in \mathbb{R}$ is the bias for the i-th kernel.

For the Convolution 1D in the 3-rd layer of the critic and the scorer, $X^{critic} \in \mathbb{R}^{r_X^{critic} \times c_X^{critic}}$ is denoted as its input matrix, where $r_X^{critic} = 21$ and $c_X^{critic} = 10$ is the number of rows and

Table 3: Details of the scorer network

Layer	Type	Kernel size	Filter	Stride	Padding	Output shape	Number of parameters
1	Input	-	-	-	-	10×21	-
2	Transpose	-	-	-	-	21×10	-
3	Convolution 1D	1	128	1	no	128×10	2816
4	Residual block	-	-	-	-	128×10	98560
5	Residual block	-	-	-	-	128×10	98560
6	Residual block	-	-	-	-	128×10	98560
7	Residual block	-	-	-	-	128×10	98560
8	Residual block	-	-	-	-	128×10	98560
9	Reshape	-	-	-	-	1280	-
10	Fully Connected	-	-	-	-	1	1281

Table 4: Details of the residual block adopted from [19]

Layer	Type	Kernel size	Filter	Stride	Padding	Output shape	Number of parameters
1	Input	-	-	-	-	128×10	-
2	ReLU	-	-	-	-	128×10	-
3	Convolution 1D	3	128	1	1	128×10	49280
4	ReLU	-	-	-	-	128×10	-
5	Convolution 1D	3	128	1	1	128×10	49280
6	Residual connection	-	-	-	-	128×10	-

columns of its input matrix, respectively. The output matrix will be $Y^{critic} \in \mathbb{R}^{f_{critic} \times c_X^{critic}}$, where $f_{critic} = 128$ is defined to be the number of kernels for this Convolution 1D. The operation of the Convolution 1D in the 3-rd layer of the critic and the scorer is similar to (14) but with the change of the input size and the number of kernels and it is defined as

$$Y_{i,j}^{critic} = \sum_{m=1}^{r_X} \sum_{n=1}^{c_w^{critic}} (k_{m,n}^{dis,i} \cdot X_{m,n+j}^{critic}) + b^{dis,i}, \tag{15}$$

where $1 \leq i \leq f_{critic}$, $1 \leq j \leq c_X^{critic}$, $w^{dis,i} \in \mathbb{R}^{r_X \times c_w^{critic}}$ is the i-th convolutional kernel, $c_w^{critic} = 1$ is the number of columns in the kernel, and $b^{dis,i} \in \mathbb{R}$ is the bias for the i-th kernel.

In the critic and the scorer, the reshape function aligns each row into a row vector. The reshape function acted on the matrix $A \in \mathbb{R}^{p \times q}$ can be represented by

$$vec(A) \triangleq [A_{r1} \quad A_{r2} \quad \cdots \quad A_{rp}], \tag{16}$$

where A_{ri} is denoted as the i-th row vector in the matrix A.

In the generator, the reshape function at the third layer converts a vector $B \in \mathbb{R}^{p \cdot q}$ into a matrix $\hat{B} \in \mathbb{R}^{p \times q}$ by putting the p-th q elements of B to the p-th row of the matrix $\hat{B}$. It can be represented as

$$\hat{B} = \begin{bmatrix} B_1 & B_2 & \cdots & B_q \\ B_{q+1} & B_{q+2} & \cdots & B_{2q} \\ \vdots & & & \\ B_{(p-1)q+1} & B_{(p-1)q+2} & \cdots & B_{pq} \end{bmatrix}, \tag{17}$$

where B_i is denoted as the i-th elements in the matrix B.

Let $X^{GS} \in \mathbb{R}^{r_X^{GS} \times c_X^{GS}}$ be the input and $Y^{GS} \in \mathbb{R}^{r_X^{GS} \times c_X^{GS}}$ be the output of the Gumbel-Softmax in the 11-th layer of the generator network, where $r_X^{GS} = 10$ is the number of row for the input matrix X^{GS} and $c_X^{GS} = 21$ is the number of column in X^{GS}, respectively. The Gumbel-Softmax in the generator network is computed as

$$Y_{i,j}^{GS} = \frac{e^{X_{i,j}^{Gumbel}}}{\sum_{m=1}^{c_X^{GS}} e^{X_{i,m}^{Gumbel}}}, \tag{18}$$

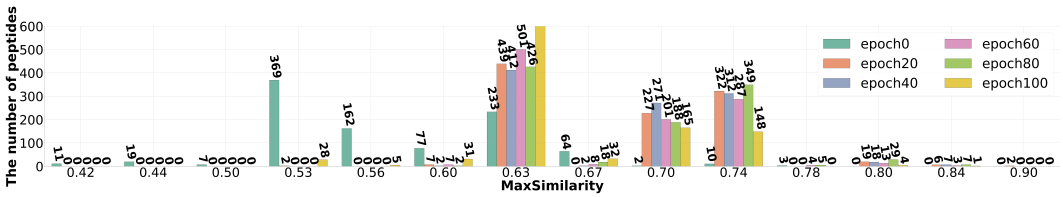

Figure 8: The histograms for the number of the peptides with respect to their maximum similarity. It shows that increasing the training epoch may increase the number of peptides with higher maximum similarity.

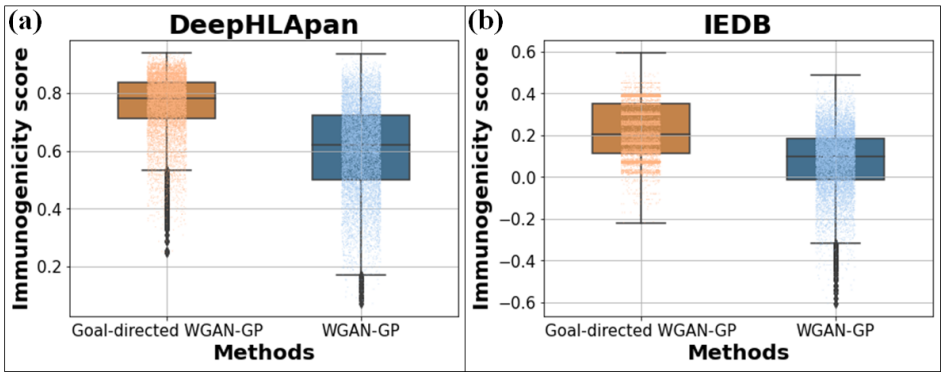

Figure 9: (**a**) The dots and box plot showing the median immunogenicity score of the 10000 brain peptides predicted using DeepHLApan 4.1 [35] and produced from the generator trained with our goal-directed WGAN-GP is about 0.78, and it is higher than the score of the 9837 peptides generated from the vanilla WGAN-GP with about 0.62. (**b**) The dots and box plot showing the median immunogenicity score of the 10000 brain peptides predicted using IEDB [28] and produced from the generator trained with our goal-directed WGAN-GP is about 0.2, and it is higher than the score of the 9837 peptides generated from the vanilla WGAN-GP with about 0.15.

where $1 \leq i \leq r_X^{GS}$, $1 \leq j \leq c_X^{GS}$, $Y_{i,j}^{GS}$ is denoted as the element of Y^{GS} at the i-th row and the j-th column, X^{Gumbel} is evaluated as

$$X^{Gumbel} = \frac{X^{GS} - ln(g^{GS})}{\tau}, \tag{19}$$

where $g^{GS} \in \mathbb{R}^{r_X^{GS} \times c_X^{GS}}$ is a matrix with its value generated by the exponential distribution $f(x) = \lambda e^{-\lambda x}$, $\lambda = 1$, and $\tau = 0.75$.

D Additional experiment results

Figure 8 depicts histograms illustrating the number of peptides generated from the generator trained with 0, 20, 40, 60, 80, and 100 epochs, categorized by their maximum similarity rounded off to the second decimal place. The results demonstrate a tendency for an increased number of peptides with higher maximum similarity as the training epoch progresses. These findings indicate that the generator, trained using our designed training scheme, learns the patterns of peptide sequences in the training dataset and generates similar but not identical sequences, as no sequence achieves a maximum similarity of 1.

In Fig.9, we have used both DeepHLApan 4.1 [35] and IEDB [28] to predict the immunogenicity scores. Our analysis confirms that peptides generated by our system consistently exhibit higher immunogenicity scores, in terms of the median, compared to those from the vanilla WGAN-GP.

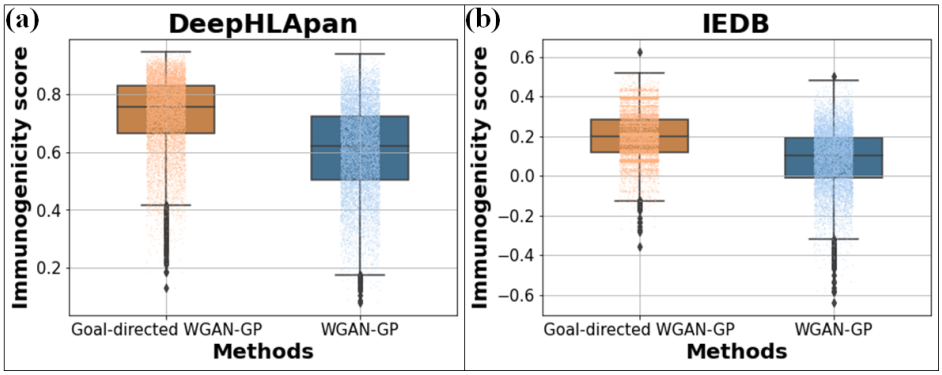

Figure 10: The averaged immunogenicity score of the brain peptide sequences generated from our designed goal-directed WGAN-GP (orange line) compared to the WGAN-GP without the immunogenicity predictor (blue line) through training and their cumulative average score in the brown dashed line and the purple dotted line, respectively.

E Experiment result using brain cancer epitopes

In this section, we present the comparison of our goal-directed WGAN-GP with the vanilla WGAN-GP using the brain cancer epitopes from TSNAdb [31]. The network architecture and hyperparameters are the same as in (2).

The cumulative averaged immunogenicity score in (10) is depicted as a brown dashed line and a purple dotted line in Fig.10. Notably, the result is similar to that of in Fig.3 as the immunogenicity scores of sequences generated from our goal-directed WGAN-GP continue to increase, reaching approximately 0.93. Conversely, sequences generated without the immunogenicity predictor in the GAN only reach around 0.7 and show no further increase.

To further validate that our goal-directed WGAN-GP generates brain peptides with higher immunogenicity scores compared to the WGAN-GP without the immunogenicity predictor, we employed the generator trained for 1000 epochs from both schemes to produce 10000 peptide sequences. Considering only 9 or 10-mers peptides, the resulting number of peptide sequences is 9895 for our goal-directed WGAN-GP and 9986 for the WGAN-GP. In Fig.11, the immunogenicity scores of the peptide sequences from both models are presented in dots and box plots, which is similar to the result in Fig.4.

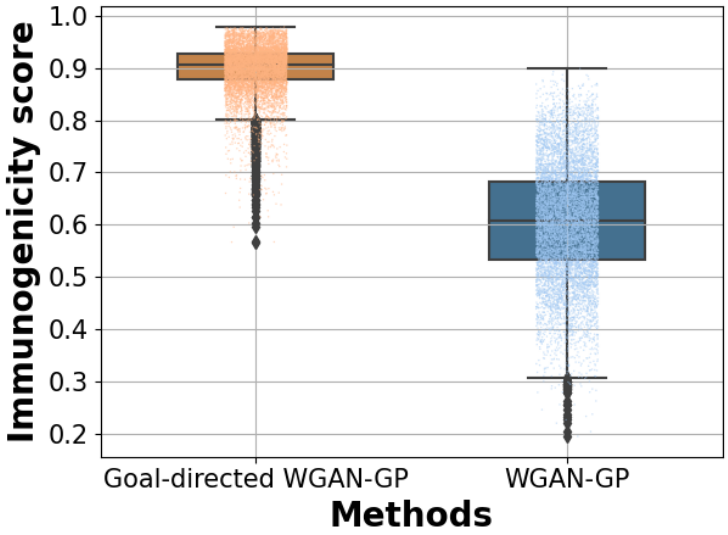

Figure 11: The dots and box plot showing the median immunogenicity score of the 10000 brain peptides produced from the generator trained with our goal-directed WGAN-GP is about 0.9, and it is higher than the score of the 9837 peptides generated from the vanilla WGAN-GP with about 0.6.