Bioinformatics, YYYY, 0–0
doi: 10.1093/bioinformatics/xxxxx
Advance Access Publication Date: DD Month YYYY
Manuscript Category

Subject Section

ThermalGen: a sequence-based thermal stable protein generator trained using unpaired data

Hui-Ling Huang[1,†], Chong-Heng Weng[2], Torbjörn E. M. Nordling[3,4], Yi-Fan Liou[5,*, †]

[1]International Program of Health Informatics and Management, Chang Gung University, 7F,College of Management,No.259, Wenhua 1st Rd., Guishan Dist., Taoyuan City 33302, Taiwan, [2]Department of Computer Science and Information Engineering, National Central University, Taoyuan, Taiwan, [3]Department of Mechanical Engineering, National Cheng Kung University, No. 1 University Road, Tainan City 701, Taiwan, [4]Department of Applied Physics and Electronics, Umeå University, 90187 Umeå, Sweden, [5]Department of Vitual-Reality Interaction with Artificial Intelligence Technology, Coretronic Reality Incorporation, No. 5, Wenhua Rd., Hukou Township, Hsinchu County 303, Taiwan

*To whom correspondence should be addressed.

† Equal contribution

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Abstract

Motivation: Synthesis of proteins with novel desired functions is challenging but sought after by the industry and academy. The dominating approach is based on trial and error inducing point mutations, assisted by structural information or predictive models built with paired data that are difficult to collect. This study proposes Sequence-Based Unpaired-Sample of Novel protein Inventor (SUNI), to build ThermalGen for generating thermally-stable proteins according to sequence information only.

Results: The ThermalGen generator could mutate the input sequence with median number reaching 32 residues, an infeasible number to test using the conventional point mutation method. This study generated a thermally-stable form of a known normal protein, 1RG0. By superimposing the thermally-stable form on the wild-type, high similarity is shown, indicating that the basic function was conserved despite 51 mutated residues. Several molecular dynamics simulations indicated that the thermal stability increased.

Conclusion: This proof of concept demonstrated that transfer of a desired protein function from one set of proteins is feasible.

Availability and implementation: The source code of ThermalGen can be freely accessed at https://github.com/markliou/ThermalGen/ with MIT license.

Contact: yifanliou@gmail.com

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Creating a new protein with novel functions is useful for various practical applications. The point mutation method, which creates novel functions without changing the original activities of proteins, is mostly used for new protein production. However, to generate novel proteins, it is necessary to identify the key positions of proteins and replace the residues with correct ones. Most mutations lead to protein instability (Upadhyay *et al.*, 2019). Hence, protein stability is a key issue influencing the application of proteins, including medical, industrial, and

experimental uses. The low thermal stability of proteins is often a drawback during biocatalysis (Upadhyay *et al.*, 2019).

Traditional methods rely on obtaining certain recombinant proteins and then screening the resulting proteins after verifying their impact on the target property. To overcome such huge laboratory consumption, bioinformatics provides another view of this task, analyzing proteins before producing them in the real world. Researchers have adapted traditional statistical methods and machine learning to provide rational modifications for protein engineering (Pucci *et al.*, 2016). The most intuitional way to solve this problem is to collect the pairing data, including the wild-type protein and their corresponding mutations. Another way to solve this problem is to obtain structural information on the target proteins. However, most proteins do not contain the corresponding mutations, and obtaining structural information is a difficult task. Hence, it is necessary to develop a method to handle unpaired mutation data based only on protein sequences.

Recently, deep neural networks (DNNs) have made many breakthroughs in various fields, such as computer vision and natural language processing. DNN can be applied to solve biological problems. (Webb, 2018; Angermueller *et al.*, 2016); however, for handling sequences, the recurrent neural network (RNN), which is mostly used in natural language processing, is the most considered owing to its good performance in sequential data; additionally, RNN is widely used in natural language processing. For example, Li et al. used a 1-dimension convolution neural network to encode the regional information of proteins for feeding into an RNN. (Li *et al.*, 2021) The protein-lipid interaction residues have also been predicted using RNN (Katuwawala *et al.*, 2021). Although RNNs are known to work well with such sequence data, they are not easy to train. Therefore, feedforward neural networks have been considered. For example, after applying the condition gated-convolution, the convolutional neural network (CNN) works well in a sequential generative model. (Oord *et al.*, 2016)

Among various neural networks, generative models have more potential applications than predictive models, and the generative adversarial neural network (GAN) has already provided many state-of-the-art works. Since GAN has gradually drawn attention from bioinformatics researchers, an increasing number of applications of GAN have emerged recently. For protein prediction, GAN effectively recovers missing parts of the protein sequence (Li *et al.*, 2018; Anand and Huang, 2018). In DNA prediction, Feedback GAN (Gupta and Zou, 2018) has created a framework to produce genes with desired properties. By using an RNN-based function analyzer to evaluate its similarity to the known DNA with desired properties, feedback GAN gradually shifts the distribution to sample the right DNA.

However, traditional GAN itself only randomly generates new samples along the axis of a high-dimensional space and not necessarily towards the desired properties we want in real world. To solve this problem, CycleGAN (Zhu *et al.*, 2017), which is known to handle unpaired data, has been proposed. CycleGAN is good at handling image translation, which is not suitable for processing bio-sequence data; hence, a new method, sequence-based unpairing-sample of novel protein inventor (SUNI), based on CycleGAN, has been proposed.

Before introducing SUNI, it is important to know that CycleGAN is best known to handle an image style transfer task, that is, making a photo similar to a Van Gogh's painting with astonishing details. SUNI treats the origin sequences of amino acids as original photos, and the target sequences with desired properties are a Van Gogh's painting. Thus, the properties of target sequences are now Van Gogh's style. SUNI transfers the origin and target sequences to each of its opposite domains. Subsequently, it transfers them back to each of its origin domain, and the similarities to origin sequences are estimated. This procedure extracts important information from both domains. Thus, vital information of the target domain is the desired property. MolGAN (DeCao and Kipf, 2018) uses GAN to produce valid molecules and adjust its properties using reinforcement learning. In addition to using an unconstrained GAN, Mol-CycleGAN (Maziarka et al., 2020) uses the conditional GAN property to generate a molecule with desired properties.

Since many protein applications are highly dependent on thermal stability, this study attempts to use SUNI to transfer the current proteins with target properties to new proteins with a new add-on property. The thermal stability character as an example and a generator, named ThermalGen, has been proposed to create a new thermally stable form of original proteins. To validate whether the sequences generated from ThermalGen are more stable than their corresponding wild-type sequences, this study used the wild-type protein with 3D structure information and produced its thermally stable forms generated using ThermalGen for molecular dynamic (MD) simulations.

2 Methods

This study proposed ThermalGen, which is trained using SUNI, a CycleGAN-based method (Zhu *et al.*; Hoffman *et al.*), to generate artificial thermal-stable protein (ATSP) sequences. The training process requires two generators and two discriminators. One of the generators created an ATSP sequence from a normal protein (NP) sequence, and the other created an artificial normal protein (ANP) sequence from a thermally stable protein (TSP) sequence. The MD simulations were effective for protein stability in our previous study (Liou *et al.*, 2016). Then several MD simulations were performed to evaluate the structural stability of ATSP.

2.1 Data sets

CycleGAN is known to solve the domain-transferring problem using unpaired data from different target domains. To generate TSPs from NPs using CycleGAN, the normal protein dataset and thermally stable protein dataset were collected.

In this study, normal proteins were defined as the proteins found in organisms having normal optimal growth temperatures (OGTs) between 20 and 40 °C. TSPs were collected from organisms with OGTs above 40 °C. The OGTs of species were directly obtained according to a previous study (Zeldovich *et al.*, 2007). The overall species used in this study can be found in Sup.1.

Thirty-two and 172 species have been defined as thermophiles and normal organisms, respectively. Detailed information is provided in Sup1. The sequences were downloaded from UniProt database (UniProt release 2018_11) according to the taxonomy ID. After cross-checking the taxonomy IDs, two species, *Shewanella denitrificans* and *Desulfotalea psychrophila*, were found to have no recorders. The crude protein dataset has 32,958 and 3,963,711 sequences of TSPs and NPs, respectively.

Since the amount of NPs is much larger than that of TSPs, down sampling is needed for the training process. Further, subsampling for NPs was implemented using Usearch (Edgar, 2010) with a 20% threshold. The final TSP dataset contained 32,439 sequences whereas the NP dataset contained 421,437 sequences.

2.2 SUNI

SUNI is a CycleGAN-based neural network. The architecture of CycleGAN includes two generators and two discriminators. To train a

CycleGAN, a dataset containing more than one domain is necessary. CycleGAN solves the problem of unpaired-data generation, which is most commonly observed in the real world. For example, to create a thermally stable protein from a normal protein, the predictor created using regular machine learning methods would need a pairing dataset that collects normal proteins and their corresponding thermally stable mutations. In CycleGAN, the unpaired data is the input of one of the generators, and the output data are the inputs of other generators for evaluating cycle consistency.

As shown in Fig.1., generators G1 and G2 were used to transform thermally stable proteins to normal ones and normal proteins to thermally stable ones. Discriminators D1 and D2 were used for evaluating whether the given protein sequences were real normal proteins or ones created from G2 and G1.

In SUNI, the architecture of CycleGAN is thought to be a special case of an auto-encoder (Hinton and Salakhutdinov, 2006), which has the same input and output, and latent variables of the auto-encoder should be subjected to distribution of another domain. Notably, the inputs of generators are discrete, and the gradients of back propagation are interrupted. To overcome this problem, straight-through estimation (Bengio *et al.*, 2013) was applied. The generators and discriminators were constructed using a 1D fully convoluted neural network. The architecture is provided in Sup2. The kernel size was set to 9. The activation functions use the Mish (Misra, 2019), which is defined as

$$\mathrm{f(x)} = \mathrm{x}\ \tanh(\ln(1+\mathrm{e}^{\mathrm{x}})) \tag{1}$$

where x is the output of the previous layer.

Instance normalization, which is known to be helpful in style-transfer tasks (Ulyanov *et al.*, 2016), is used in generators, and batch normalization (Ioffe and Szegedy, 2015) is used in discriminators.

As the padding operation provides more space for inputs, some information is hidden in the padding region by generators. The padding penalty is also used:

$$\begin{gathered}\mathcal{L}_{\text{padding}}(G, X_{padding}) = \\ \mathbb{E}_{x\sim p_{\text{data}}(x)}\left[X_{padding}\log G\left(X_{padding}\right)\right] + \\ \mathbb{E}_{x\sim p_{\text{data}}(x)}\left[\left(1-X_{padding}\right)\log G\left(1-X_{padding}\right)\right]\end{gathered} \tag{2}$$

where G is the generator and $X_{padding}$ is the padding of input sequences.

To translate samples from the original domain to another domain, generator G can translate X to Y, and the adversarial loss is expressed as follows:

$$\begin{gathered}\mathcal{L}_{\text{GAN}}(G, D, X, Y) = \\ \mathbb{E}_{y\sim p_{\text{data}}(y)}[\log D(y)] + \mathbb{E}_{x\sim p_{\text{data}}(x)}\left[\log\left(1-D\left(G(x)\right)\right)\right]\end{gathered} \tag{3}$$

where D is the discriminator to distinguish the samples from generators or real sample datasets. The cycle-consistency loss used to evaluate sample recovery is

$$\begin{gathered}\mathcal{L}_{\text{cyc}}\,(G1, G2) = \\ \mathbb{E}_{x\sim p_{\text{data}}(x)}\left[x\log G2\left(G1(x)\right)\right] + \mathbb{E}_{y\sim p_{\text{data}}(y)}\left[y\log G1\left(F2(y)\right)\right]\end{gathered} \tag{4}$$

where G1 and G2 indicate the forward and reverse generators, respectively.

The generated sequences should be brought back to the original sequences using F, that is, $x \rightarrow \mathrm{G1(x)} \rightarrow \mathrm{G2(G1(x))} \approx \mathrm{x}$. Hence, the full objective is

$$\begin{gathered}\mathcal{L}(G1, G2, D1, D2) = \\ \mathcal{L}_{\text{GAN}}(G1, D1, X, Y) + \mathcal{L}_{\text{GAN}}(G2, \mathrm{D2}, Y, X) + \mathcal{L}_{\text{cyc}}(G1, G2) + \\ \lambda\mathcal{L}_{\text{padding}}\left(G1, X_{padding}\right) + \lambda\mathcal{L}_{\text{padding}}\left(G2, y_{padding}\right)\end{gathered} \tag{5}$$

where λ was set to 0.1 in this study. The weights of cycle-consistency loss are different from the original CycleGAN due to information balancing. After this adjustment, the weights of cycle consistency can be

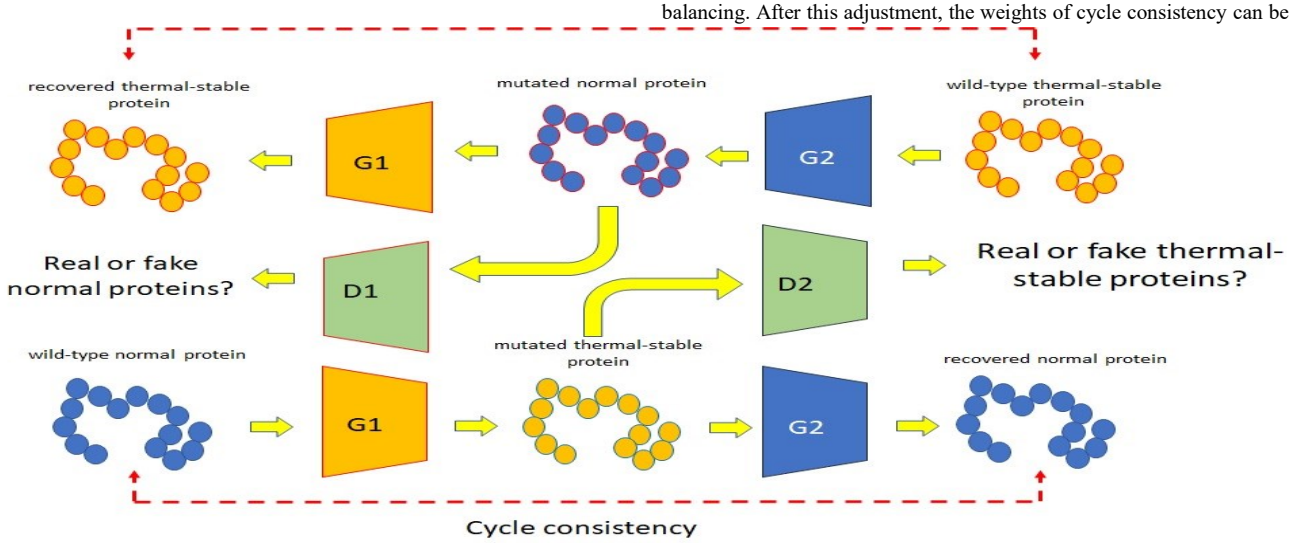


Fig. 1. The architecture of SUNI and its application, ThermalGen. The basic architecture of SUNI contains two generators and two discriminators. Both the generators and discriminators use the sequence information with one-hot representation. The generators transform the sequences from one domain to another domain, and the discriminators identify whether the sequences are real or generated. ThermalGen, the example used in this study, consisted G1 and G2, which transform normal proteins to thermally stable proteins and thermal proteins to normal proteins. D1 and D2 are the discriminators that are used to identify whether the sequence is generated from G2 or G1.

set to 1, which was 10 in the original CycleGAN.

2.3 ThermalGen

ThermalGen, which is a SUNI-based generator, generates thermal proteins from normal proteins using protein sequences only.

The TrEMBL data indicated that most of the proteins contained less than 400 residues in the database (data not shown). In this study, the input length of sequences was set to 512. All proteins were applied with zero padding or they were trimmed if the proteins had fewer or more than 512 residues, respectively. Before feeding into the generators and discriminators, the protein sequences were embedded into 128 dimension representation and position embedding was applied. The embedding parameters were learned using a gradient decent. The architectures of generators and discriminators can be found in Sup. 2.

RMSprop was used to optimize the weights of ThermalGen with a batch size of 32. The learning rates of generators and discriminators were 0.001 and 0.002, respectively, with polynomial decays of 0.0001 and 0.0002, respectively. Gradient clipping and weight averaging were applied to stabilize the training process. The warm-up and training iterations were set to 50 and 50,000, respectively. All training processes were run with an Intel(R) Core(TM) i7-6700K CPU and Nvidia-RTX2080ti GPU.

2.4 Substitution matrices

Substitution matrices aim to estimate the variances between original and mutated sequences generated using generators. The Needleman-Wunsch dynamic programming algorithm was used. The Pairewase2 module of Biopython was used for implementation. The gap opening and extension penalties, which are the same as the default settings from EMBOSS (Rice *et al.*, 2000), were set to 10 and 0.5, respectively. The paddings of input sequences were removed before alignment. The conservation positions, which are defined as the positions having identity residues, were discarded in the substitution matrices because the mutation positions were much lesser than the conservation positions and the mutated positions were of interest in this study.

2.5 MD simulations

The entire dataset with redundant sequences was used to select candidate samples. All sequences were checked for their corresponding PDB information. Protein conformation reliability can be measured using the B-factor, which is highly correlated with entropy (Octav Caldararu *et al.*, 2019); therefore, structure 1RG0, with the lowest B-factor, 1.8, was selected.

1RG0 sequence is fed into the ATSP generator, and structures of the novel ATSP sequence were predicted using Alphafold 2 (Jumper *et al.*, 2021). The ATSP and original 1RG0 forms were visualized using Discovery Studio 2020 Client. These two structures were superimposed for further comparison of secondary and tertiary structures, which were optimized using Amber force field, while other predicting parameters were set as default. The structure with the highest score was used to perform MD simulations. Gromacs was used for MD simulations. To accelerate computation, Gromacs with a tag of 2021-beta3, provided by NVIDIA, was used for acceleration using GPUs.

The simulation temperatures were set to 300 K, 400 K, and 500 K, and all simulation times were set to 10 ns. Each protein was placed in a cubic box soaked in SPC/E water (Berendsen *et al.*, 2002). All bond lengths were constrained using the LINCS algorithm. The box edges were at least 10 nm away from the proteins with a periodic boundary condition (PBC) in all directions. Sodium and chloride counter ions were used in this study. The entire MD system was then treated with minimization, which uses the steepest descent method for 50,000 iterations. The cutoff was 1 with the Particle Mesh Ewald (PME) condition. The NPT and NVT equilibrations were sequentially run for 100 ps with 2 fs steps. Before the main MD simulation, an energy minimization of 100,000 steps was performed. Each main simulation with different temperatures was then performed with 2 fs steps.

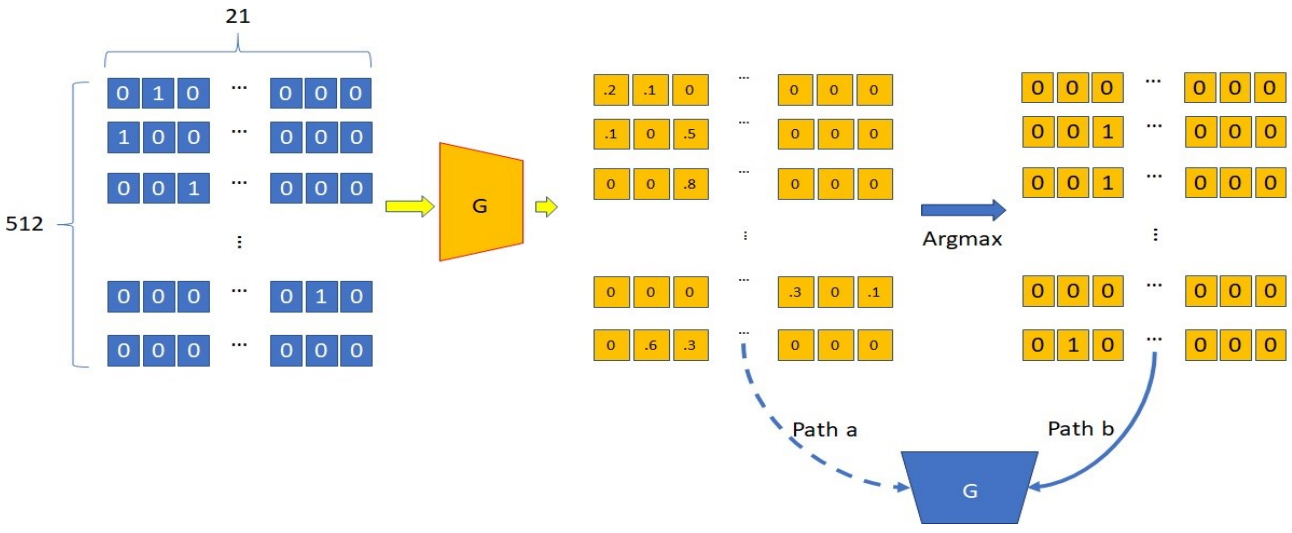


Fig. 2. Comparison of the generated sample delivering from one generator to another generator. The generators use the one-hot represented sequences as inputs, and they generate the sequences with non-one-hot representation mostly using Softmax function. Traditional CycleGan will directly feed the generated sequence to another generator, as shown with the dash-line arrow (Path A). In SUNI, the generated sequences are further treated to make the one-hot representations, and they are then fed into another generator (Path B).

2.6 Simulation analysis

The simulation process was recorded every 100 ps. The secondary structures were defined using DSSP (Joosten *et al.*, 2011), and the standard tools were provided by Gromacs (gmx do_dssp). The 3_{10} helix, α-helix, π-helix, and β-sheets were observed to evaluate the stability of proteins. The root-mean squared deviations (RMSDs) were also considered to evaluate differences between the original and current structures. C_α RMSD was calculated as follows:

$$RMSD_T = \sqrt{\frac{\sum_{l=1}^{n} (\hat{C}_l - C_l)^2}{n}} \quad (6)$$

where n and T indicate the length of the sequence and structure snapshot at time T, respectively and C_l and $\hat{C}_l$ are the positions of C_α at T=0 and T=T, respectively.

3 Results and discussion

This section presents the training status, followed by an analysis of the proposed sequences and molecular dynamic simulation analysis.

3.1 Differences between the original CycleGAN and SUNI

As the CNN is already known to work well on sequential data (Oord *et al.*, 2016), this study attempted to use the CNN for handling protein sequences. Compared to conventional supervised learning methods that require pairing data to train the models, the dataset was created by collecting as many target proteins and their mutations as possible. However, it is difficult to create such a dataset due to laboratory consumption. Using an unpairing dataset, which collects data from different domains, would solve such problems, and CycleGAN is best known to handle this type of task.

Conventional CycleGAN uses a generator trained with a certain domain to generate targets that are fed directly to other generators to recover them back to the original domain. Most generators use the softmax function to deliver output. Softmax provides a continuous gradient for training; however, it hides information into the softmax output. This hidden information is called "dark knowledge" (Hinton *et al.*, 2015), which can be learned using a neural network. If dark knowledge is learned using a generator, "self-adversarial attack" (Bansal *et al.*, 2018) would happen. Several studies have utilized various methods to reduce the effects of self-adversarial attacks. For example, Bansal *et al.* (Bansal *et al.*, 2018) and Bashkirova *et al.* (Bashkirova *et al.*, 2019) used recycle-reconstruction and noising inputs, respectively, to eliminate self-adversarial attacks. Since dark knowledge causes the problem of reconstructing, it is necessary to erase the information. As shown in Fig. 2, the inputs generated from previous generators will be further handled to give only the maximum argument rather than directly giving the softmax results.

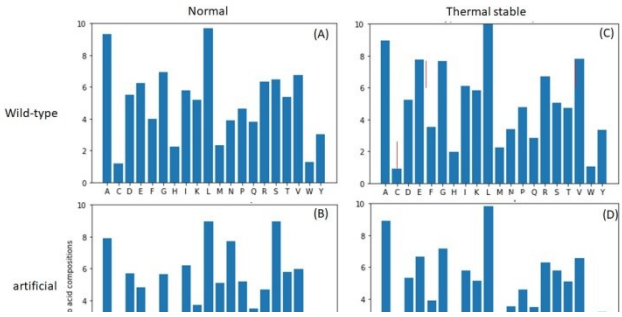


Fig. 3. Amino acid compositions (AACs) of the wild-type and generated proteins. (A) AACs of wild-type normal proteins. (B) AACs of wild-type thermally stable proteins. (C) AACs of artificial normal proteins generated based on thermal-stable proteins. (D) AACs of artificial thermal stable proteins generated based on normal proteins.

On the other hand, CycleGAN is well known to work with images; however, only a few studies have focused on time series data, such as natural language processing or acoustic data processing. This study focused on generating protein sequences using only sequential primary structures. The difference between vision and language data is information density (He *et al.*, 2021). The language data, which are artificial, are highly sematic and more information-dense than images with heavy spatial redundancy. This characteristic also has an impact when vanilla CycleGAN is used for handling sequence data. To maintain output length, the padding token is most frequently used. Using vanilla CycleGAN, the generated sequences were found to be identical to the input sequences but the padding region was not. This scenario suggests that the generator hides information in the padding regions. Therefore, padding regularization terms are used to overcome such a situation.

During training, the learning rates of discriminators should be less than that of generators to avoid model collapse. The model collapse of protein sequence generators is similar to that of regular GANs. Sequences comprising almost the same residues at the same positions are generated with the generators trained using inappropriate learning rates. ThermalGen also tries to train the discriminators in lesser time as compared to generators, similar to the original GAN (Goodfellow et al., 2014); however, mode collapse still occurs. The final training strategy uses learning rates of 1E-4 and 2E-4 for discriminators and generators, respectively. The warm-up, which uses a learning rate of 1E-9 for the first 50 steps, is also essential for avoiding mode collapse. The training process was finally stopped at 50,000 iterations, which took approximately 70 h for training, and the models were used for generating protein sequences for further analysis.

3.2 Comparing the amino acid compositions of wild-type and artificial sequences

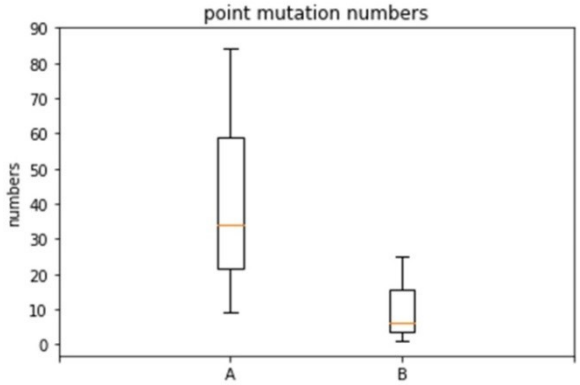
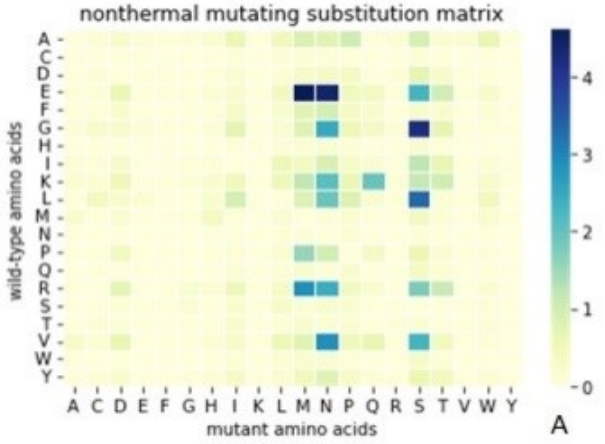


Fig. 4. The point mutation numbers of artificial proteins. Red line indicates the median. The top and bottom of the box indicate Q1 and Q3 numbers, respectively. The top and bottom horizontal lines denote the highest mutation number and least mutation number. A. Point mutation numbers of artificial thermal stable proteins. B. Point mutation numbers of artificial normal proteins.

Fig. 3 (A) and (B) show 20 amino acids whose NP sequences went through G1 and produced ATSP. Among the ATSP sequences, Ser(S) increased the most. This corresponds to a previous study (Haikarainen *et al*., 2013) that proposed that Ser(S) ratios are identified as potential reasons for thermal stability. Fig. 3 (C) and (D) show 20 amino acids, whose TSP sequences went through G2 and produced ANP. Among the ANP sequences, the amount of Glu(E) decreased the most. This corresponds with a previous experiment (Ding *et al*., 2007), where the negatively charged amino acid Glu(E) is needed when bacteroid heat resistance proteins form salt bridges. There are many factors that affect the thermal stability of proteins; however, salt bridge plays an important role. Salt bridge is the electrostatic force that lies between the negative and positive charges. Many studies have indicated that it significantly contributes to the thermal stability of proteins. Glu(E) is negatively charged and Ser(S) is polarized but does not dissociate, which grants structural freedom during electrostatic reaction of the salt bridge.

Fig. 4 (A) shows a box plot for calculation of the number of mutated amino acids of the ATSP sequence that is generated through the process of NP sequence going through G1. As shown, the median was 32, which means that an NP sequence mutated 32 amino acids and turned them into



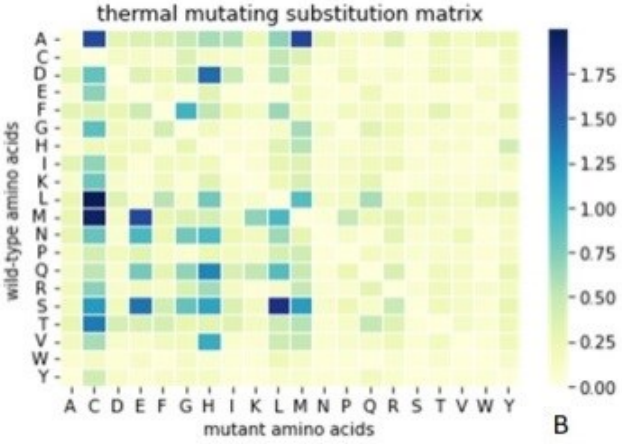


Fig. 5. Substitution matrices of artificial proteins and their corresponding template wild-type proteins. The colors indicate mutation numbers. The transition from light yellow to dark blue indicates the increasing number of substitution numbers. (A) the substitution matrix of artificial thermal stable proteins generated from normal proteins. (B) the substitution matrix of artificial normal proteins generated from artificial thermally stable proteins.

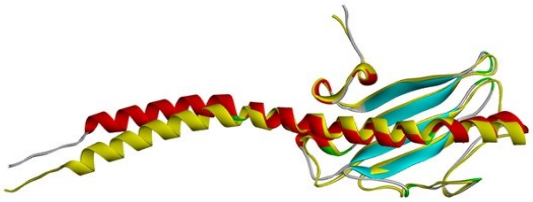


Fig. 6. Superimposition of 1RG0 and its ATSP form. Wild-type 1RG0 is shown in the ribbon representing colored structures according to the secondary structures. The red, blue, and green colors indicate the α-helices, β-sheet, and coiled structures, respectively. The ATSP form structures are colored with yellow.

the ATSP sequence. The system automatically identifies 32 amino acid sites to mutate into ATSP. This exceeded the limit of 12 amino acid sites performed manually (Lee et al., 2014). Fig. 4 (B) shows a box plot for calculation of the number of mutated amino acids of the TSP sequence generated through the process of ANP sequence going through G2. As shown, the median was 8, which means that a TSP sequence mutated 8 amino acids and turned them into an ANP sequence. Compared to Figure 4 (a), the TSP can be turned into ANP with only ¼ of the mutation points. The main goal is to lessen the mutation and conserve the domain of functioning protein while losing its function through mutation. Through this process, we found that the most efficient mechanism for Cycle-CANs is incompatible with the fact that many weak polar bounds need to be placed correctly to make it thermally stable but only some of them are destroyed to make it normal.

3.3 Substitution matrices of mutated sequences

SUNI is based on the CycleGAN model. Under the condition of cycle-consistency loss, the generated sequences should be returned to the original sequences. Figure 5 shows that the two substitution matrices are a collection of scores for aligning amino acids with one another, each coming from G1 (shown in Fig.5A) and G2 (shown in Fig.5B), respectively. As shown in Figure 5A, the mutation of E(Glu) to M(Met) was the highest; in Figure 5B, the mutation of L(Leu) to C(Cys) was the highest. It has been proposed that amino acids can form distorting residues and symmetrizing residues (Shalit and Tuva-Arad, 2020). We

explained how G1 and G2 satisfy the optimal mutation choice for the CycleGAN model: sequence x, $x \rightarrow G1(x) \rightarrow G2(G1(x)) \approx x$. E(Glu) and M(Met) are distorting residues, which usually contain long or polar side chains. L(Leu) and C(Cyst) are symmetrizing residues that usually feature short or aromatic side chains.

3.4 Substitution matrices of mutated sequences

The major difference between paired and unpaired data training methods is that the sample exists in the real world. To validate the generated samples, image-to-image tasks can be observed and evaluated directly using human eyes. In contrast, the generated proteins cannot be easily seen only by the human eye to evaluate the function of proteins. Using other machine learning models is an alternative; however, GANs, including SUNI, already have discriminators that are also classifiers made using machine learning. Thus, an evaluation method that can provide more direct results than machine learning methods is essential. Hence, this study uses MD simulations at different temperatures.

To check whether ATSPs can perform better than their corresponding NP forms, the NP sequence (Uniprot ID: P17838) with a tertiary structure (PDB ID:1RG0) and the lowest B-factor among the training dataset was selected. The structures of 1RG0 and its secondary structure are shown in Fig. 6. A β-sheet embeds an α-helix whereas both the N- and C-terminals form random structures. The primary structures of 1RG0 were fed directly into ThermalGen to obtain the ATSP form. The length of 1RG0 was 120. ThermalGen provided a sequence with the same residue numbers excluding 51 mutated residues. The similarity between NP and ATSP forms was 67.5%. To further check the structures of these two forms, the ATSP form of sequence is directly assigned to AlphaFold 2 without changing any parameters for predicting structures, especially the protein template, to confirm that the structures are not influenced by human cause. The final structures predicted using AlphaFold 2 are shown in Fig. 6. The NP and ATSP forms were superimposed, and the main structures of α-helix and β-sheets mostly overlapped. This suggests that even though 51 residues were different, the structures that were directly related to the protein functions were still similar between these two forms. To further estimate the thermal stability of these two forms, the proteins were placed in simulated environments at different temperatures, such as 300 K, 400 K, and 500 K. Each MD simulation had a simulation time of 10 ns; hence, a total of 60 ns simulations were performed. As shown in Fig. 7, the β-sheet structures were stable in both simulations of wild-type and ATSP forms at 300 K. Until the end of the simulation at 300 K, the β-sheet structures still existed. The α-helix structures were less stable than the β-sheet, and they were unfolded approximately 2 ns and 4 ns of wild-type and ATSP, respectively. This suggested that the α-helix structure of ATSP was more stable than that of the wild-type. To further estimate the stabilities of β-sheet, more simulations with higher temperatures (400 K and 500 K) were performed. The β-sheet remained stable until the end of both simulations at 400 K; however, it was disrupted at 500 K. The wild-type retained only the β-sheet structures for 3 ns; however, ATSP maintained its structure for 6 ns. The scenarios indicated the same results that the a-helices and β-sheets of ATSP were more stable than the wild-type ones.

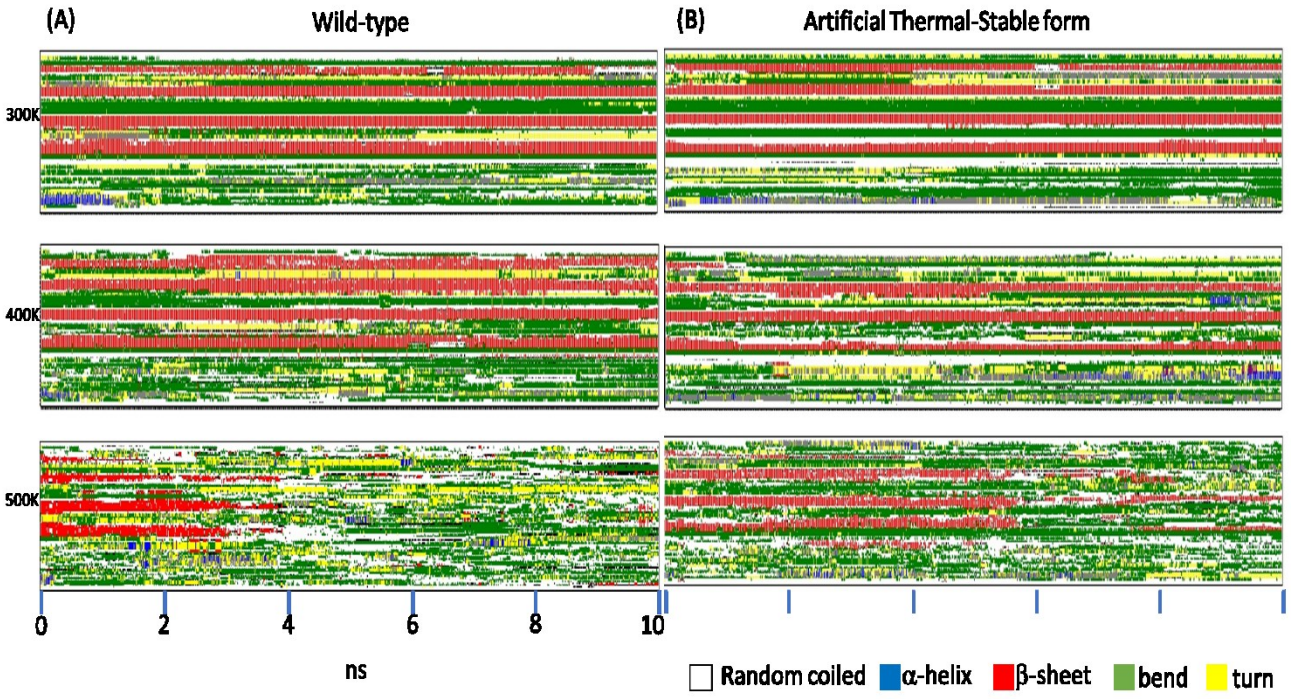


Fig. 7. Structural analyses of 1RG0 and its ATSP form in trajectories under 300K, 400K, and 500K. Secondary structures are defined using DSSP and the structures were analyzed every 100 ps. The colored blocks indicate the secondary structure of the residues at the simulation time. The white, blue, red, green and yellow indicate the coiled, α-helical, β-sheet, bend and turn structures, respectively. A. the results of the wild-type of 1RG0 simulated at 300K, 400K and 500K for 10 ns. B. the results of the ATSP form of 1RG0 simulated at 300K, 400K and 500K for 10 ns.

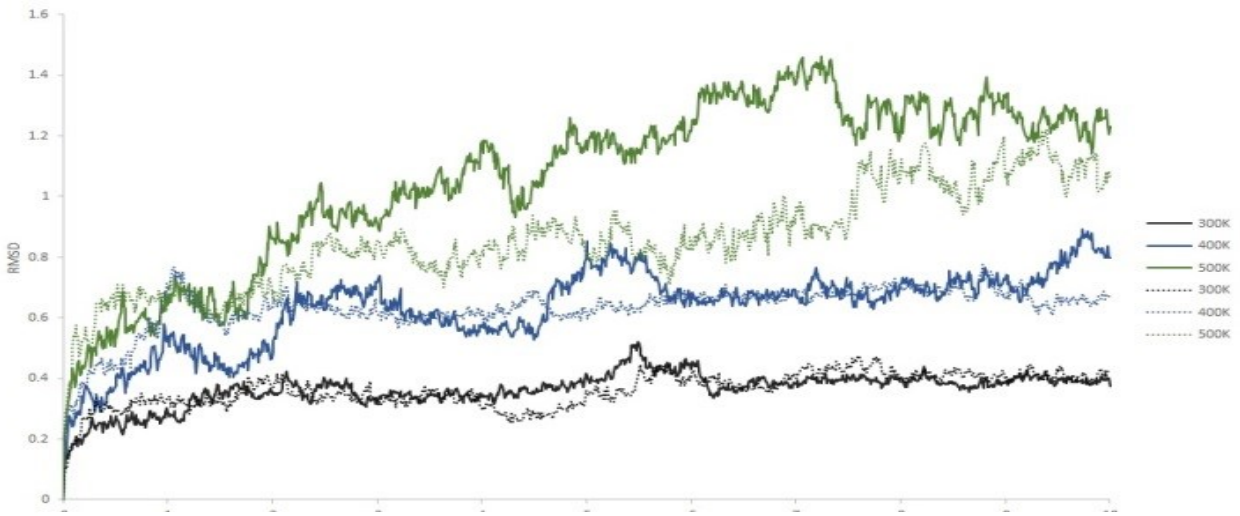


Fig. 8. Records of the backbone RMSD values with respect to the starting structure of 1RG0 and its ATSP form under different simulation temperatures. The solid and dash lines indicate RMSD of the 1RG0 and ATSP forms, respectively. The black, blue, and green lines indicate the trajectories under 300K, 400K, and 500K, respectively.

To evaluate the stability of the entire structure, RMSD was also used. As shown in Fig. 8, RMSD of wild-type and ATSP showed almost the same trends at 300 K. After increasing the simulation temperatures from 300 K to 400 K, RMSD also increased; however, it still showed similar trends. These simulation results indicated that the stabilities would be similar at 300 K and 400 K. However, the simulations under 500 K yielded significantly different results. ATSP had a lower RMSD than wild-type forms. RMSD of the wild-type significantly increased after a 2 ns simulation time; however, ATSP still had a steady RMSD of approximately 0.8. The RMSD of ATSP remained stable until 8 ns. This also suggested that ATSP had more stable structures than the wild-type one.

4 Conclusion

Creating a protein with novel functions is helpful for many application areas, such as the fermentation industry or for experimental purposes. However, traditional machine learning-based methods are built using paired data that are difficult to collect. This study proposed SUNI, a CycleGAN-based method that is known to handle unpaired data problems, and it creates ThermalGen to generate a thermally stable protein according to the given protein sequence. Simulations were performed at different temperatures to evaluate the generated proteins. The results suggested that the generated proteins had more stable structures than the wild-type. This method provides a way to create proteins with novel functions.

Funding

TEMN gratefully acknowledge support from the Ministry of Science and Technology in Taiwan (MOST 109-2224-E-006-003, 110-2326-B-006-001-MY3, and 110-2222-E-006-010).

Conflict of Interest: none declared.

Data availability

The training dataset can be accessed with the ThermalGen source code or directly downloaded from: https://github.com/markliou/ThermalGen/blob/master/ThermalGen_sourcecode/training_dataset.tar.bz2

References

Anand,N. and Huang,P. (2018) Generative modeling for protein structues. 6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings, 31.

Angermueller,C. *et al.* (2016) Deep learning for computational biology. Molecular Systems Biology, 12, 878.

Bansal,A. *et al.* (2018) Recycle-GAN: unsupervised video retargeting. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 11209 LNCS, 122-138.

Bashkirova,D. *et al.* (2019) Adversarial self-defense for cycle-consistent GANs. Advances in Neural Information Processing Systems, 32.

Bengio,Y. *et al.* (2013) Estimating or propagating gradients through stochastic neurons for conditional computation.

Berendsen,H.J.C. *et al.* (2002) The missing term in effective pair potentials. The Journal of Physical Chemistry A, 91, 6269-6271.

DeCao,N. and Kipf,T. (2018) MolGAN: an implicit generative model for small molecular graphs.

Ding, Y.-R. *et al.* (2007) Relationship between salt bridge and thermostability of complete genome microorganisms. Journal of Huazhong Agricultural University, 26(3).

Edgar,R.C. (2010) Search and clustering orders of magnitude faster than BLAST. Bioinformatics, 26, 2460-2461.

Goodfellow,I. *et al.* (2014) Generative adversarial networks. Communications of the ACM, 63, 139-144.

Gupta,A. and Zou,J. (2018) Feedback GAN (FBGAN) for DNA: a novel feedback-loop architecture for optimizing protein functions.

Haikarainen,T. *et al.* (2014) Crystal structure and biochemical characterization of a manganese superoxide dismutase from Chaetomium thermophilum.

Biochimica et Biophysica Acta - Proteins and Proteomics. 1844(2), 422-429.

He,K. *et al.* (2021) Masked autoencoders are scalable vision learners.

Hinton,G. *et al.* (2015) Distilling the knowledge in a neural network.

Hinton,G.E. and Salakhutdinov,R.R. (2006) Reducing the dimensionality of data with neural networks. Science, 313, 504-507.

Hoffman,J. *et al.* Cycada: cycle-consistent adversarial domain adaptation. In International Conference on Machine Learning, 1989-1998.

Ioffe,S. and Szegedy,C. (2015) Batch normalization: accelerating deep network training by reducing internal covariate shift. In 32nd International Conference on International Conference on Machine Learning, 1, 448-456.

Joosten,R.P. *et al.* (2011) A series of PDB related databases for everyday needs. Nucleic Acids Research, 39, D411.

Jumper,J. *et al.* (2021) Highly accurate protein structure prediction with AlphaFold. Nature, 596, 583-589.

Katuwawala,A. *et al.* (2021) DisoLipPred: accurate prediction of disordered lipid-binding residues in protein sequences with deep recurrent networks and transfer learning. Bioinformatics.

Lee, C.-W. *et al.* (2014) Protein thermal stability enhancement by designing salt bridges: a combined computational and experimental study. Plos One, 9(11).

Li,Y. *et al.* (2021) DELPHI: accurate deep ensemble model for protein interaction sites prediction. Bioinformatics, 37, 896-904.

Li,Z. *et al.* (2018) Protein loop modeling using deep generative adversarial network. Proceedings - International Conference on Tools with Artificial Intelligence, 2017-November, 1085-1091.

Liou Y.-F. *et al.* (2016) A hydrophobic spine stabilizes a surface-exposed α-helix according to analysis of the solvent-accessible surface area. BMC Bioinformatics, 17(19): 503.

Misra,D. (2019) Mish: a self regularized non-monotonic activation function. arXiv.

Octav Caldararu *et al.* (2019) Are crystallographic B -factors suitable for calculating protein conformational entropy? Physical Chemistry Chemical Physics, 21, 18149-18160.

Oord,A.van den *et al.* (2016) WaveNet: a generative model for raw audio.

Pucci,F. *et al.* (2016) Predicting protein thermal stability changes upon point mutations using statistical potentials: introducing HoTMuSiC. Scientific Reports, 6, 1-9.

Rice,P. *et al.* (2000) EMBOSS: The European Molecular Biology Open Software Suite. Trends in Genetics, 16, 276-277.

Shalit Y. and Tuvi-Arad I. (2020) Side chain flexibility and the symmetry of protein homodimers. Plos One 15(7).

Ulyanov,D. *et al.* (2016) Instance normalization: the missing ingredient for fast stylization.

Upadhyay,R. *et al.* (2019) RiSLnet: rapid identification of smart mutant libraries using protein structure network. Application to thermal stability enhancement. Biotechnology and Bioengineering, 116, 250-259.

Webb,S. (2018) Deep learning for biology. Nature, 554, 555-557.

Zeldovich,K.B. *et al.* (2007) Protein and DNA sequence determinants of thermophilic adaptation. PLoS Computational Biology, 3, e5.

Zhu,J.Y. *et al.* Unpaired image-to-image translation using cycle-consistent adversarial networks monet photos.

Zhu,J.Y. *et al.* (2017) Unpaired image-to-image translation using cycle-consistent adversarial networks. Proceedings of the IEEE International Conference on Computer Vision, 2017-Octob, 2242-2251.