Generating Antimicrobial Peptide Sequences using RankGAN

1st Abhimanyu Agarwala
Department of Mathematics
Stevens Institute of Technology
Hoboken, United States of America
aagarw16@stevens.edu

2st Shantanu Kolekar
Department of Mathematics
Stevens Institute of Technology
Hoboken, United States of America
skolekar@stevens.edu

***Abstract*—Our research employs RankGAN, an advanced generative adversarial network, to synthesize antimicrobial peptide (AMP) sequences. This innovative approach leverages Negative Log Likelihood and Ranking functions, coupled with cosine similarity measures, to ensure the generated AMPs closely align with the properties of high-efficacy natural peptides. Our methodology harnesses deep learning to mimic the structural complexity of AMPs, aiming to contribute significantly to the development of new therapeutic agents. Encouraging preliminary results demonstrate that our generated sequences exhibit substantial similarity to effective AMPs, marking a promising advance in peptide synthesis technology. This project not only demonstrates the feasibility of using GANs for complex biological sequence generation but also sets a foundation for future explorations in the field. The code and methodologies behind our innovative approach are shared on our GitHub repository: https://github.com/shant-kolekar/rankGAN/**

I. Introduction

Our project emerges with a focus on the generation of antimicrobial peptide (AMP) sequences using a deep learning model, RankGAN, originally known for its application in image generation. The versatility of RankGAN presents an exciting opportunity to explore its utility beyond its conventional realm. Our venture into this uncharted territory aims to leverage the model's generative capabilities to produce AMP sequences that are not only novel but also biologically viable.

AMPs are a unique class of therapeutic agents, distinguished by their broad-spectrum activity and reduced likelihood of developing resistance. They are a cornerstone in the innate immune system and are known for their potential as next-generation antibiotics. The design of AMPs, however, is not trivial due to their complex nature and the intricate balance of their properties which dictate their efficacy. Recognizing this, our project does not attempt to revolutionize the industry overnight but instead makes a methodical advance towards augmenting the process of AMP design using AI.

In order to use RankGAN for our objective, we started by adapting its architecture to handle the sequential nature of peptide chains. The model was fine-tuned to learn from the structure and properties of existing AMP sequences. This required a meticulous process of encoding the amino acid sequences into a suitable format for the generative network to process. The training phase involved feeding the model with data on known AMPs, allowing it to understand and eventually replicate the complex patterns within these sequences.

As we progressed, the generated AMP sequences were evaluated for their likeness to naturally occurring peptides. The evaluation metrics and methodology focus on the relative ranking of sequences, highlighting the distinctiveness and effectiveness of RankGAN in sequence generation tasks. We utilize an innovative technique that integrates Negative Log Likelihood and Ranking functions alongside cosine similarity evaluations. This method is pivotal in ensuring the generated antimicrobial peptides closely replicate the characteristics of naturally occurring, effective peptides.

To generate antimicrobial peptide sequences with RankGAN, we encountered and overcame several challenges. Configuring TensorFlow and ensuring compatibility with other packages. We also fine-tuned the sequence generation to restrict the character set to the twenty standard amino acids, discarding the remaining six alphabetic characters. Furthermore, we used the model's ability to produce variable lengths to more accurately reflect natural peptide diversity. These adaptations were critical in aligning our computational model with biological realities.

II. Contribution

In our collaborative project, the first team member focused on the selection and architectural understanding of RankGAN, distinguishing its merits over other models. He implemented token generation using NLTK and crafted functions to format the amino acid sequences appropriately. His work extended to generating and reconverting tokens into sequences. The second team member dedicated their efforts to fine-tuning the model parameters, ensuring robust sequence generation. He also took charge of evaluating the generated sequences.

III. Description of Dataset

Our dataset comprises unique antimicrobial peptide (AMP) sequences, each representing a potential candidate for curative application. One such sequence, "AACSDRAHGHICESFKSFCKDSGRNGVKLRANCKKTCGLC," exemplifies the complexity and diversity within our collection. These sequences vary in length and composition, reflecting the natural variability found in AMPs. Excluding unnatural amino acids (B, J, O, U, X, and Z), they contain only the 20 standard amino acids, crucial for maintaining biological relevance. Our data serves as a rich foundation for training our RankGAN model, with the goal of generating new, biologically viable AMP sequences. The dataset can be found at https://cbbio.online/AxPEP/?action=dataset.

IV. Data Pre-processing

In our project, we devised a multifaceted approach to preprocess and convert antimicrobial peptide (AMP) sequences into a format amenable for RankGAN training. Utilizing the Natural Language Toolkit (NLTK), we initially tokenized each sequence to its constituent amino acids, thereby simplifying the subsequent steps. Our custom function, ‘add_spaces‘, took these tokenized sequences and systematically inserted spaces between each amino acid, enhancing readability and processing efficiency.

Next, we engineered a two-way mapping between each amino acid and a unique numerical code through ‘get_dict‘, establishing a ’dictionary’ that served as the foundation for encoding and decoding during the model’s learning phase. The ‘text_to_code‘ function then transformed the tokenized text into a sequence of numerical codes, ensuring each sequence had a consistent length by padding shorter sequences with an end-of-file code.

Additionally, ‘get_tokenlized‘ and ‘get_word_list‘ functions were implemented to extract and deduplicate the tokens from our dataset, facilitating the creation of a comprehensive vocabulary of amino acids used in the model training. Finally, ‘text_precess‘ was the overarching function that brought together all these elements, managing the tokenization, dictionary creation, and encoding processes, while also handling the separation of training and evaluation datasets, if provided. This meticulous pre-processing pipeline was pivotal for the effective training of our RankGAN model on AMP sequences.

V. Methodology

We initiated the generative process using the RankGAN architecture to produce antimicrobial peptide (AMP) sequences. The RankGAN model is a type of generative adversarial network (GAN) specifically designed to optimize the generation process by ranking the quality of generated data. It consists of two main components: the Generator and the Discriminator.

The Generator’s role is to create sequences that mimic the true data distribution of AMPs. It was initialized with parameters such as embedding dimension, hidden layer dimension, and start token, all carefully chosen to reflect the complexity of AMP sequences. The Generator’s architecture ensures that the sequences are not only diverse but also maintain biological relevance.

The Discriminator, on the other hand, acts as a critic, evaluating the quality of sequences produced by the Generator. It uses convolutional neural network layers to capture the intricate patterns in the data and to classify sequences as either real (coming from the true dataset) or fake (generated by the Generator). By doing so, the Discriminator guides the Generator towards producing more realistic sequences.

Training of the Discriminator involved generating synthetic sequences and feeding them into the model along with real sequences, allowing it to learn the difference. The Discriminator was trained iteratively to improve its accuracy in distinguishing between the real and the synthetic AMP sequences.

The innovative aspect of RankGAN comes from the feedback loop between the Generator and the Discriminator. As the Discriminator evaluates sequences, it provides a reward signal to the Generator. This signal is based on the rank of the synthetic sequences against the real ones, thus encouraging the Generator to produce sequences that are closer to the real AMPs in terms of quality. The Generator updates its parameters based on this feedback, refining its sequence production in a direction that is more likely to deceive the Discriminator.

Evaluating the performance of the model involved several metrics. The cosine similarity score, commonly used in natural language processing, assessed the similarity between the generated sequences and the real AMP sequences. The EmbSim metric evaluated the embedding similarity, which is a measure of how well the generated sequences align with the vector space of the real sequences. Finally, the Nll (Negative Log Likelihood) provided a probabilistic measure of how well the Generator’s sequences fit the true data distribution.

The evaluation process was crucial as it provided quantitative feedback on the model’s performance. With each training epoch, we used these metrics to fine-tune both the Generator and the Discriminator, aiming to enhance the quality of the generated sequences. By iteratively improving through adversarial training, the model aimed to reach a point where the generated AMP sequences were indistinguishable from real sequences, in terms of the evaluated metrics. This rigorous training and evaluation cycle was central to our methodology, ensuring that the final model could generate high-quality AMP sequences with the desired characteristics.

VI. Tools & Technologies

Our project utilized PyCharm and Macbook M2 Air, leveraging its default GPUs for computation. We employed TensorFlow 1.15.3 for machine learning, NumPy for data manipulation, SciPy for scientific computations, NLTK for text processing, and , creating an effective toolkit for AMP sequence generation.

VII. Experiment

The focus was on adjusting hyperparameters for LSTM and CNN architectures. These include embedding dimensions and hidden dimensions, critical for model complexity. Filter sizes and the number of filters in CNNs influence feature extraction capabilities. Regularization and dropout probabilities are used to avoid overfitting, ensuring broader applicability of the model. Batch size is also fine-tuned to balance between training efficiency and model performance. These hyperparameters are central to the model’s ability to effectively learn and generalize from the sequence data.

In our project, we explored the efficacy of two models: GAN and RankGAN with enhanced ranking capabilities. While the traditional GAN framework provided a foundational understanding, the modified RankGAN introduced advanced ranking functionalities. This modification aimed to improve the model's ability to assess and generate more accurate antimicrobial peptide sequences, leveraging a more nuanced ranking mechanism. The comparative analysis of these models offered insights into the strengths and limitations of each approach.

VIII. Results

We compared the performance of GAN and RankGAN using the Negative Log Likelihood (NLL) metric. The GAN model yielded an NLL score of 1.1634, while the modified RankGAN demonstrated a more impressive score of 0.5419. This significant improvement in NLL for RankGAN highlights its enhanced capability in accurately generating antimicrobial peptide sequences. The results underscore the effectiveness of our modifications to the RankGAN model, particularly in its ranking capabilities, proving it to be a more efficient tool in sequence generation tasks compared to the standard GAN model.

IX. Problems and Issues

Our project faced compatibility challenges with TensorFlow 1.x, as it does not support Python versions higher than 3.6. This limitation necessitated careful selection and adjustment of other libraries, particularly numpy and pandas, to ensure they align with the Python version used. This compatibility issue was a crucial aspect of setting up our development environment, impacting both the selection of tools and the overall workflow of the project.

X. Conclusion

In conclusion, our project effectively demonstrates the potential of RankGAN in generating biologically relevant antimicrobial peptide (AMP) sequences. The comparative analysis with traditional GAN models, underscored by the impressive Negative Log Likelihood scores, highlights RankGAN's superior performance in sequence generation. This research not only contributes to the field of computational biology but also sets a precedent for future studies in AMP synthesis using advanced machine learning techniques.

References

[1] K. Lin, D. Li, X. He, Z. Zhang, and M.-T. Sun, "Adversarial ranking for language generation," *Advances in neural information processing systems*, vol. 30, 2017.

[2] F. Juefei-Xu, R. Dey, V. N. Boddeti, and M. Savvides, "Rankgan: a maximum margin ranking gan for generating faces," in *Computer Vision–ACCV 2018: 14th Asian Conference on Computer Vision, Perth, Australia, December 2–6, 2018, Revised Selected Papers, Part III 14*. Springer, 2019, pp. 3–18.

[3] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.