COVID-19 Mutation Prediction Using Generative Adversarial Networks

Tiff Lyman and Matthew Knotts
Dr. Chuxu Zhang
COSI 165B: Deep Learning
May 2021

1. Introduction:

In developing effective treatments for pathogens, one major fear is the pathogen's ability to develop resistance to that treatment. At the most basic level, drug resistance is conferred through genetic mutation. This change can lead to downstream changes of the pathogen, possibly rendering previous treatments ineffective. This fear is especially prevalent in RNA viruses, whose single-stranded genetic code is more susceptible to random mutation than DNA based pathogens, such as bacteria and eukaryotic parasites. In the course of the COVID-19 pandemic, many variants have arisen, sparking fear that these variants may be resistant to the current vaccines. While it has not currently been found that these variants have resistance to any of the current vaccines, the possibility of a resistant strain emerging is still very real. One way to curb this possibility is to stop the spread of the virus. While mass vaccination is underway in affluent countries, most other areas do not have the resources or programs in place to vaccinate their population at the required rate. The next best solution is to predict how COVID-19 will mutate in order to develop effective vaccines before the mutant strain emerges. Deep learning methods have recently been explored in current computational biology research, specifically using neural networks for mutation and protein-protein interaction prediction. A 2020 paper implemented a GAN to predict Influenza A mutations, called MutaGAN, with 97.6% sequence generation accuracy [2]. Thus, implementing such a GAN is of interest in predicting COVID-19 mutations.

Due to time and computing restrictions, however, it is not feasible to predict mutations over the whole genome of COVID-19. Instead, a subsequence of the genome, which encodes for the receptor binding domain (RBD) of the spike (S) protein of COVID-19 will be used for the model. This subsequence is especially relevant because it is the target of both the Pfizer and Moderna mRNA vaccines. Thus, it is the region with the most concern of mutation, as it may render these vaccines ineffective.

2. Model:

2.1. Data

COVID-19 sequence data was acquired from NCBI GenBank. All sequences were labeled for "Surface Glycoprotein", corresponding to the S protein. All sequences were dated from January 2020 - April 2021. Results were further filtered to have a length of 1273, the length of the consensus sequence of the S protein [3], returning 69, 976 sequences. All sequences were then aligned with the S protein consensus sequences, with calculated alignment scores (1273 indicating a full match) distributed in figure 1.

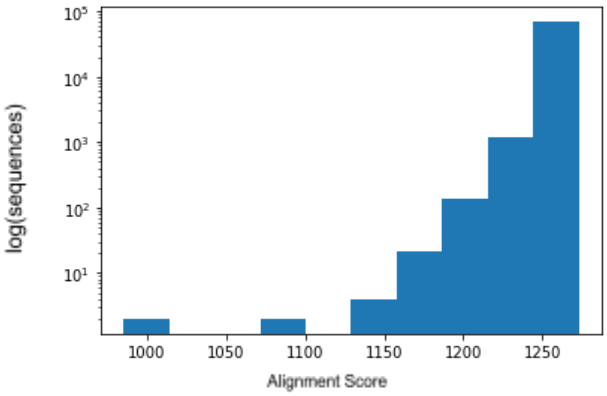


Figure 1: Log-scale histogram of sequence alignment scores. A higher score corresponds to a closer match to the consensus sequence, with 1273, the length of the consensus sequence, indicating an exact match

The S protein sequences were then further filtered for a 70 character-long substring that corresponds to the RBD. For the complete spike protein, there are 25 unique characters, while there are only 23 unique characters for the RBD.

2.2. Generator:

To simulate mutations in COVID-19, we'll be using a long-short-term memory (LSTM) based generator called seq2seq [1]. The generator consists of two parts: an encoder and a decoder. The encoder takes in sequences (shape = [sequence_length]) and encodes them using an embedding (shape = [sequence_length, num_features]) followed by a series of bi-directional LSTM Cells. The encoded product and states then pass through the decoder which uses a series of LSTM layers to produce an embedded sequence (shape = [sequence_length, num_features]), where the maximum feature value represents the feature in that position of the sequence. The states (h, c) for each encoding LSTM cell are passed to their respective LSTM layer in the discriminator. The generator uses negative-log likelihood for its loss function, a hidden LSTM dimension of 256, and a dropout with probability 0.5 between the LSTM Cells.

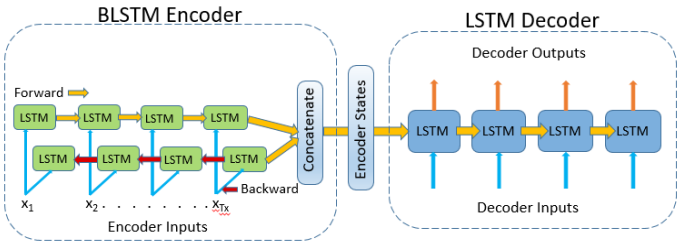


figure 2: Architecture of seq2seq generator

2.3. Discriminator:

Similar to the generator, the discriminator starts by encoding the input sequence using the same embedding to LSTM Cell architecture. The discriminator then uses a fully connected layer with sigmoid activation to reduce each sequence index down to a value between 1 and 0 indicating how realistic having that feature at that index is. Finally the realness of each index is averaged to give the whole sequence a "realness" score. If the average is greater than 0.5, it is considered genuine. The discriminator uses all the same features and parameters as the generator, except loss. Here we use a variant of negative-log loss as shown below

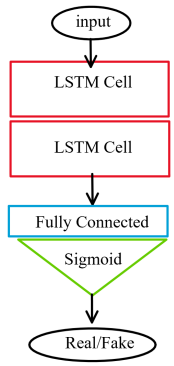


Figure 3: Discriminator Architecture

$$loss = -y \cdot \log(p) + (1-y)\left(\log(1-p)\right)$$

where y is the ground truth and p is the sequence's predicted score.

3. Experiments:

3.1. Main Results:

Both the generator and discriminator will use a batch size of 64 and the Adam optimizer with a learning rate of 0.0002. We also limit the gradient norm to under 5.0 so that changes aren't too drastic. After 20 epochs, the following results were obtained for accuracy:

Figure 4: Discriminator accuracy across 20 epochs. The percentage of fake sequences not detected is indicative of the generator's accuracy

Note that the accuracy of the generator is inversely represented by the percentage of fake samples detected; if fewer mutations are detected then the generator is doing better. What we see here is a visualization of the action-reaction process between the two networks. The discriminator appears to be bounded at about 80% accuracy, where each time the following epoch sees the generator make substantial improvements. We also have the loss for each network shown below:

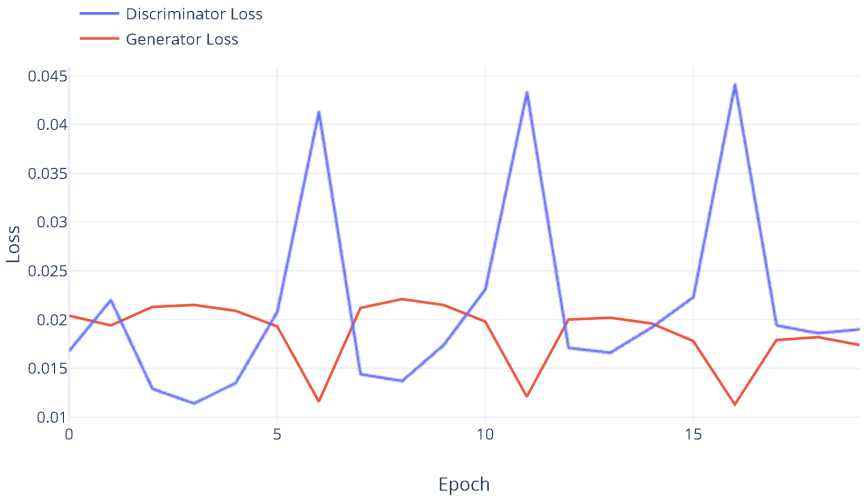


Figure 5: Loss values for generator and discriminator across 20 epochs. Note that the peaks correspond to peaks in accuracy

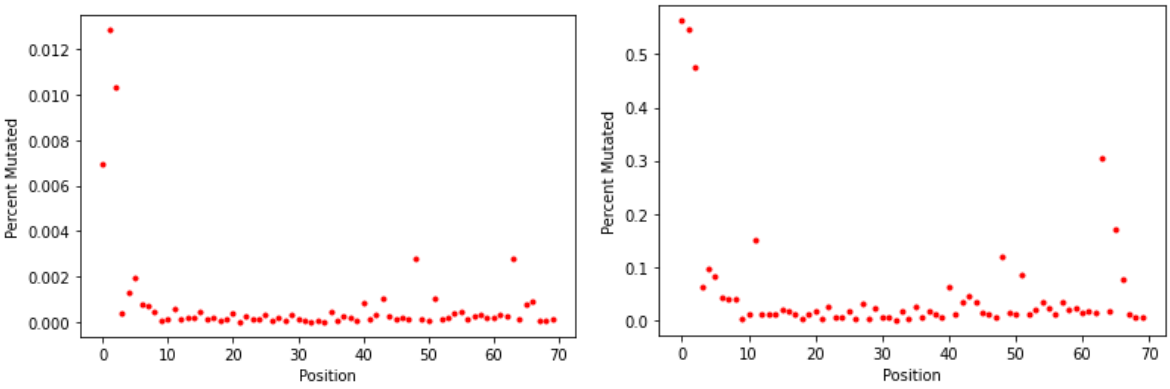
One interesting observation about the loss diagram is that there's a spike in discriminator loss right *before* the generator makes major improvements. Previously we were using a learning rate of 0.001, where we weren't seeing any patterns like the ones above. It turns out that the action-reaction cycle takes about 5 epochs at a learning rate of 0.0002 (1/5 the original learning rate), so we were seeing roughly the same stage of the cycle each epoch before we reduced the learning rate.

We also see that loss on the discriminator tends to increase with each cycle while the generator loss is decreasing. Because the RBD sequence is only 70 characters, the data suggests that we would start to see a downward curve in discriminator accuracy if training continued. The last local maximum for

average accuracy was about 4 points lower, the discriminator's reaction appears to be less intense across each trough, and the margin between fakes detected versus real samples accepted grows smaller. Eventually the sequences would likely be identical, so there might not even be a mutation.

3.2. Mutation Frequency

With the mutations that were successful at deceiving the discriminator, we wanted to know where these mutations were occurring. The real rbd sequences are almost all identical, but a few of them have variations. For that reason we plotted the indices where these variations occur and compare them with the indices where mutations are most frequent



3.3. Results on Spike Data

Due to time constraints we were only able to evaluate the full spike sequence over 2 epochs:

Value	Epoch 0	Epoch 1
Discriminator Accuracy	51.06%	43.42%
Real Samples Accepted	56.55%	44.91%
Fake Samples Detected	45.57%	41.93%
Generator Loss	0.0113	0.0104
Discriminator Loss	0.0227	0.0226

Table 1: First 2 epochs on spike sequence

Each epoch took about 6 hours so we couldn't conclude much, but one interesting observation is that accuracy went down after the first epoch. The spike and RBD sequences had similar initial accuracies and with a slight favoring towards accepting, But the RBD accuracy went up almost 20 points while the spike accuracy went down 8. However, the difference in accuracy between real and fake samples at the second epoch of the spike sequence was considerably smaller. RBD had an accept-real percentage about 10 points higher then its detect-fake percentage, while the spike sequence accuracies differ by less than 3 points. One last observation is that the generator loss for both epochs is lower than any RBD generator loss. This is likely because the discriminator didn't have enough time to identify differences and treated the fake samples similarly to the real ones.

3. Conclusion:

As shown, high mutation frequencies are concentrated towards the beginning and ends of the sequences. The high mutation frequency at the beginning of the sequences can be explained by the sequence acquisition method, as in practice sequencing tends to be less accurate at the beginning where the primer initially anneals. What is interesting, however, is the high mutation rate towards the end of the sequence, specifically around position 64. By analyzing a 3-D structure of the RBD binding to ACE-2 from the Protein Data Bank, it shows that this corresponds to an area of the protein far away from the binding interface. Since treatment-resistance is usually dependent on changing the interaction at the binding interface, it is unlikely that the predicted mutated sequences would correspond to COVID-19 strains resistant to current treatment methods.

One thing we would change if we were to try this again would be to use a better computer. Neither of us could afford to run this on google cloud so we were limited to a local environment. Evaluating the RBD sequence wasn't too time consuming (1 epoch = ~90min) but the spike sequence was too much for a laptop without a GPU to effectively handle. Having more computing power would allow us to verify the trend towards identical RBD sequences and further investigate the entire spike sequence. Additionally, we would want to stagger the learning rates to investigate how that affects the otherwise cyclic accuracy.

Sources:

1. Sutskever, Ilya, et al. "Sequence to Sequence Learning with Neural Networks." *CoRR*, vol. 1409, no. 3215, 2014. https://arxiv.org/abs/1409.3215
2. Berman, Daniel, et al. "MutaGAN: A Seq2seq GAN Framework to Predict Mutations of Evolving Protein Populations." *ArXiv*, 2020, arxiv.org/abs/2008.11790.
3. Massacci, A., Sperandio, E., D'Ambrosio, L. *et al.* Design of a companion bioinformatic tool to detect the emergence and geographical distribution of SARS-CoV-2 Spike protein genetic variants. *J Transl Med* **18,** 494 (2020). https://doi.org/10.1186/s12967-020-02675-4