XGenomes: Classification of Bacteriophage Genome

Group 18: Alex Barrett, Kaiyu Yang, Po-Yu Hsieh, Tanmoy Pal, Theresa McNeil
{acbarret, kyyang, pyhsieh, tanmoy, tnmcneil}@bu.edu

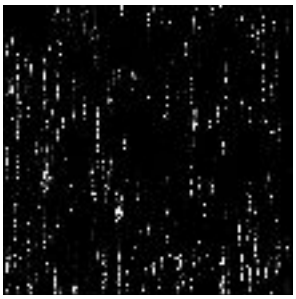


Figure 1. Genome containing enterobacteria phage λ

Figure 2. Genome containing bacteriophage (T7 phage)

1. Project Task

Our goal is to identify whether a given image of a DNA sample similar to the figures shown above contains the genome of bacteriophage T7 (T7 phage) or enterobacteria phage λ (lambda phage) based on its binding activity. This is a binary classification problem. Each experiment shows the binding behaviour of many copies of a phage's genome to an 'oligo'. Thus, the signals can be thought as indicating the location of a special "word", which in this case is a sequence of DNA that binds the oligo.

The input is an image representing the binding behavior of the genome, as seen in the figures above, and the output is the class of either lambda phage or T7 phage.

This is a challenging problem because the differences in binding patterns of the oligo to the DNA molecules are not easily distinguishable to human eye.

2. Related Work

Nyugen, Tran, Ngo, Phan [1] outline a method for classifying DNA sequences using a CNN. The input data are combinations of DNA bases analogous to words, where order of the bases is preserved in the "words", in a 2D matrix derived from one hot vectors of each of the "words".

3. Approach

We see three ways to interpret the data we have, and based on this have four separate approaches.

The simplest interpretation is to treat each extracted frame as an individual data point. We fear this will lead to overfitting of the training data, and the model will be less likely to work well in identifying oligo bindings in DNA samples with different molecule distributions, or different oligo density levels.

Our fear of overfitting is due to the consistent molecule location and density throughout an experiment. We want to avoid the model learning features specific to an experiment that are not actually relevant to the oligo. The important feature for the model to learn in this case is the vertical patterns in each "column" of the image, representing the binding behavior along a single molecule, and not the horizontal associations of the molecule placement.

Ideally we would like to use the high resolution image aggregated from the individual frames of each experiment as input. They are of much higher image quality, and illustrate the binding behavior of the genome to the molecule over time, as opposed to at a single point in time which the frames represent. However, due to the limited dataset we were given this would mean we only have eight independent data points, so this is not a feasible interpretation at this point in time.

Our way around this is to extract the individual bindings from the aggregate image. For this we need to develop an efficient and accurate way of extracting the individual columns of the images, which is difficult because the columns are not necessarily parallel to each other and some may overlap. Because of this we will aim to only extract the "good" molecules, being the columns which we are confident only contain one molecule.

Our final method of preparing data for the model input is to generate "fake" data using a tool built by us. The tool will use DNA sequence data from NEBcutter, and generate the oligo binding behavior based on that. This also allows us to adjust experiment parameters such as the density levels of the oligo and genome, and the distribution of the molecule.

Now, based on these interpretations we present our four approaches.

First, we suggest training a CNN on the individual frames from the experiments. Second, we suggest generating a variety of fake data to use as input to train a CNN, and then testing the model on the real aggregate images. We acknowledge that this is not practical for a long term solution, as these artificial data points do not capture irregularities present in real data. Third, we suggest cutting the aggregate image into smaller pieces using a uniform sliding window, and using these as input for a CNN. Finally, we suggest extracting the individual molecule strands, and using these as input for a CNN.

From here, the goal in each approach is to classify whether lambda phage or T7 phage is present.

We will use the cross-entropy loss function, defined as

$$J(\theta) = \frac{-1}{m}\sum_{i=1}^{m}\left[y^{(i)} ln\left(h_\theta(x^{(i)})\right) + (1-y^{(i)}) ln\left(1-h_\theta(x^{(i)})\right)\right]$$

Where h is the class result of our model, y is the ground truth and m is the total number of samples.

We are consistent in our choice of using a CNN to model the data because of its important quality of translational invariance. This is advantageous to us because we aim to identify the pattern of oligo binding within the genome, regardless of where the binding is located.

4. Dataset and Evaluation Metric

We have data from 4 experiments for each class. We suggest four possibilities of data input, which are listed by example and class breakdown below.

Table 1. Dataset summary example

	Training (lambda/ T7)	Validation (lambda/ T7)	Testing (lambda/ T7)
Frames	7064 / 5,600	1009 / 800	2018 / 1600
Fake data	700 / 700	100 / 100	100 / 100
Sub-images	1350 / 1350	150 / 150	500 / 500
Molecules	555 / 327	79 / 46	218 / 93

The original data is given in the form of 8 high resolution raw videos showing the oligo binding activity of the genome samples. From these videos we can extract the individual frames for use in the first approach. In total we have 10,092 frames containing the lambda phage and 8,000 frames containing the T7 phage. The images are processed using ThunderSTORM and can be used to create an aggregate of all the frames in a single experiment, also known as a super-resolution image.

The second approach made use of fake data that was generated based off of what the oligo binding activity would be in a perfect experiment. Because we knew what the oligo binding sequence was, we were able to utilize a tool called NEBcutter, which generates where binding activity should occur for a given phage genome, to produce fake data very similar to the super-resolution images provided to us. We added noise to produce a diversity of fake training data to avoid any bias that could be detected by the model from properties such as molecule position and density.

The third approach required scripting a program to slide across each aggregate image, as visualized below in Fig. 4

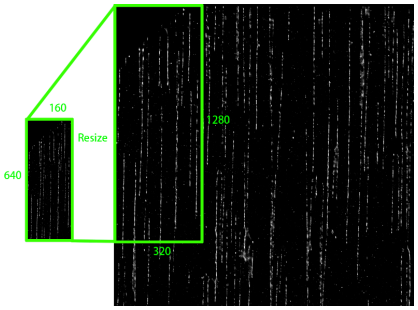


Figure 4. Uniform 160x 640 window slides across image, creating multiple data points from one image

Pre-processing for the fourth approach was the most rigorous. Commonly used line detection methods, such as Hough Transform, did not provide good results on our data, which we credit to the spotty pattern of oligo binding signals. To address this we used RetinaNet object detection network to identify the strands. RetinaNet is a single network structured by two parts. First, a pyramid-like structure is used to search for possible candidates under different resolution levels. Next, two “sub-networks” work to identify the object class (in our case this is the DNA strand) and form an accurate bounding box. Due to the narrow shape of our target strands, default height-width ratio settings for original RetinaNet will miss most targets in this case, so we used a window with half the width of the default (16~256px, compared to 32~512px), along with higher height-width ratios, which makes the search windows fit better to our target. We found that the network did not do well in discriminating between molecules with different binding patterns, so we modified the network to do one-class object detection with the goal of simply extracting molecules rather than classifying them, to be used as input for a separate

network. Our detection network uses resnet50 as backbone network, and was trained on 6000 of the annotated fake images with slightly skewed DNA signals to strengthen distortion tolerance on the network.

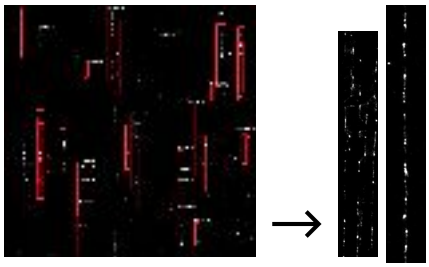


Figure 5. Example of bounding boxes on aggregate image and molecules extracted

Metric:

We will evaluate success by precision, recall, F1 score, and accuracy. Once the model is tested the predictions are separated into true positives, false positives, true negatives and false negatives. Note we defined lambda phage as the positive class and T7 phage as the negative class.

We hope to show higher than a random guess baseline which differs by experiment, depending on which data is used, varying between 50-58% and is reflected in the results section.

$$Accuracy = \frac{TP+FN}{TP+FP+TN+FN} \qquad Precision = \frac{TP}{TP+FP}$$

$$Recall = \frac{TP}{TP+FN} \qquad F1 = 2 * \frac{Precision * Recall}{Precision + Recall}$$

5. Results

The first approach did not do well, which was somewhat expected. A CNN was built with two convolution layers and one fully connected layer, making use of a 10x50 rectangular kernel as opposed to a traditional square kernel, in order to force the model to converge faster and learn the vertical patterns of the bindings.

This was optimized using adam algorithm with a learning rate of 10e-4. We attempted two different ways of splitting the data into training, testing and validation sets. First was to randomly shuffle all the frames and use a traditional 70/20/10 split, second was to use the frames from three experiments as training data with 10% of these reserved for validation, and then test on the remaining experiment. The motivation behind this is because we would ultimately hope the model could perform well in classifying the phage from experiments which it has not previously seen. Neither method was successful, and the results were no better than random.

It is important to note here that while each experiment is independent, the frames within one experiment are not, because the spatial arrangement and density of the molecule remain constant throughout the experiment. Additionally, there is little to no noticeable difference between each frame. We attribute this to being the reason why the model was unable to converge.

The second approach implemented the same CNN architecture as above and had success with classifying the fake data with a variation of molecule position, density, and noise of the fake data. However this did not generalize well to real data as you can see in the results below.

	Adjusted distribution	Adjusted density	Adjusted molecule size & noise	Real data
Accuracy	94.00%	99.50%	91.50%	26.00%
Recall	88.46%	99.00%	100.00%	0.00%
Precision	100.00%	100.00%	85.55%	0.00%
F1	93.88%	99.49%	92.91%	0.00%
Random F1	50.00%	50.00%	50.00%	50.00%

While we did our best to make the fake data look as real as possible, there is a certain amount of randomness in real data that simply can't be simulated, which is why the model did poorly in classifying the real images. However, the success in classifying the fake images gives us confirmation that our algorithm has the potential to work well. We believe that if we were given experiments on the order of thousands we could attain comparable results to the results of the fake images. A big success here is the model's ability to generalize the classification over the variety of distributions and densities.

The third approach performed well, but for the wrong reasons. As described, each aggregate image was cut into around 500 sub images. These were used as input for a CNN with similar architecture to the one used to train the fake data with the kernel changed to 8*4. The data was split by experiment as previously described. When trained on the first two experiments and tested on the third we achieve very good results. After receiving data from the fourth experiment we tested the model on it and as you can see below, the performance drops considerably.

	Test on 3rd Experiment	Test on 4th Experiment
Accuracy	90.2%	43.6%
Recall	84.77%	94.31%
Precision	92.35%	44.21%
F1	88.40%	59.95%
Random F1	50.00%	50.00%

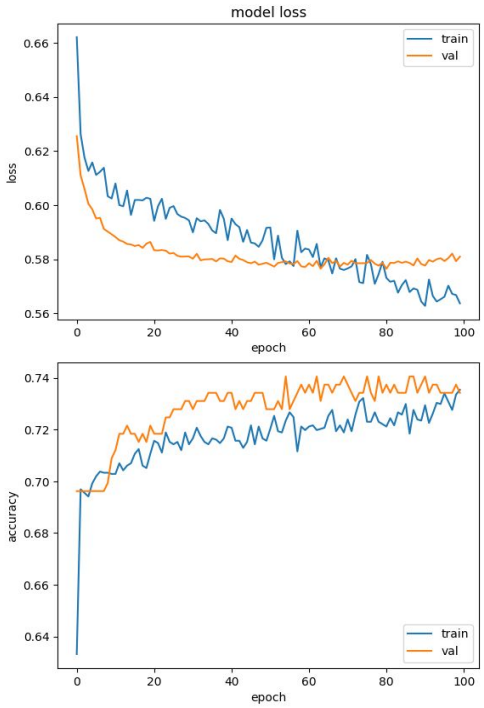
We attribute this difference in performance to a false discriminating feature present in some of the experiments, namely the density of the molecule. In the experiments we were presented, all four genomes containing lambda phage were similarly quite dense and one genome containing T7 phage had a comparable density. However the remaining three genomes with T7 phage were extremely sparse, making the density the most noticeable difference between the bacteria, even though it is unrelated to the actual binding behavior.

The results of testing on the fourth confirm our fears that the model learned to classify based on the density. This is behind the reason for the high recall and low precision when tested on experiment four, because it defaults to classifying the images as lambda based on the high density.

Accuracy	70.7%
Recall	94.1%
Precision	72.4%
F1	82.1%
Random F1	58.43%

		Actual class: lambda	Actual class: T7
Predicted class	lambda	205	78
	T7	13	15

For the fourth approach a CNN was implemented with two convolution layers, this time making use of a 3x3 kernel, followed by a fully connected layer. The input was the molecules extracted from each aggregate image, with a train/test/validation split of 70/20/10 from the shuffled data since every strand within each experiment is independent. The molecule images were not uniform in size, so we padded the numpy array representation with zeroes to counteract this. Due to the density differences explained there was a large imbalance of molecules extracted for each class, and several of the molecules extracted from the lambda class actually contained several molecules within the frame. The results below show we were able to beat our random baseline in precision, recall and F1, although the accuracy metric is close to random.



The biggest challenge we faced in this project was figuring out an effective way to use the data we were provided. Given our success with the model that classified fake data with high success, we believe we may have been able to achieve similar success if we had more experiments. If we could generate our own experiments we would be sure to include a varying density of molecules for both classes to prevent the model from learning based on density, as seen in the results of method 3.

While our results were not as good as we were hoping for, we managed to surpass the random baseline we were aiming to beat in the final three approaches.

6. Timeline and Roles

Task	**Lead**
Image processing	Alex, Tanmoy, Kaiyu
Object detection using RetinaNet	Po-Yu
CNN for frame, molecule approaches	Theresa, Tanmoy
CNN for sliding window approach, “fake data” approach	Kaiyu
Prepare report and presentation	all

References

1. Nguyen, N. , Tran, V. , Ngo, D. , Phan, D. , Lumbanraja, F. ,Faisal, M. , Abapihi, B. , Kubo, M. and Satou, K. (2016) DNA Sequence Classification by Convolutional Neural Network. *Journal of Biomedical Science and Engineering*, **9**, 280-286.

2. T. Lin, P. Goyal, R. Girshick, K. He and P. Dollar, "Focal loss for dense object detection," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
doi: 10.1109/TPAMI.2018.2858826

3. Vincze, T., Posfai, J. and Roberts, R.J. *NEBcutter: a program to cleave DNA with restriction enzymes,* Nucleic Acids Res. 31: 3688-3691 (2003)