XGenomes: Lambda-phage T7-phage Classification

Group 25: Jeffrey Leong, Yu-chun Hsiao, Linhui Jiang, E, Chengyuan
{jeefle, ychsiao, lhjiang, ecyecy}@bu.edu

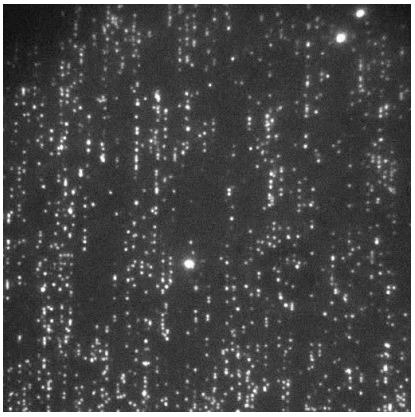

Figure 1. Sample bacteriophage to determine whether or not it is a lambda phage or T7 phage

1. Project Task

The goal of our project is to differentiate between two bacteriophages as seen in Fig. 1. Specifically, we are to differentiate if a genome sample is a Lambda phage or a T7 phage. The difficulty of this task includes localizing the image and processing it such that individual phages can be identified clearly.

2. Related Work

Random forest classification for bacteriophage was proposed in [1] implemented by a tool named MARVEL. However this technique is based on a dataset of metagenomic bins. In short, this data is not in the type we are provided.

In this proposal our task is to identify and process images in which deep convolutional neural networks have been proposed to implement image and pattern recognition [2].

3. Dataset and Metric

We are provided 5000 sample images of each bacteriophage along with XY coordinates and signal intensities for each image. The processed data directory is divided into the T7 and Lambda directories. Both directories contain 5000 images of the T7 and Lambda phage, respectively. Minimal preprocessing is needed because this is a binary classification problem and we are given two training sets of the two types to be classified.

Our metric of success lies in our accuracy. We are targeting an accuracy of greater than 90% in classifying if an image is or is not a T7 or a Lambda phage.

4. Approach

Since our goal is to differentiate between Lambda and T7, we treat this task as a binary classification problem.

It could be described as below: Given x, want to predict $\hat{y} = p(y = 1|x)$

Input (denoted as x) : images of Lambda and T7. Since each image is 150 pixel * 150 pixel. Each image will be reshaped into a 67500(150*150*3) * 1 column vector, where 3 denotes the RGB channel.

Output (denoted as $\hat{y}$): $\hat{y} = \sigma(w^T x + b)$. $\sigma(z) = 1/(1+e^{-z})$. w is a column vector which has the same dimension as x and b is constant number. Since both w and x are column vectors, their product will be a number, which could be added by b. Since our output is the probability that x is lambda or T7, it should be in between 0 and 1. Therefore, we use sigmoid function $\sigma(z) = 1/(1+e^{-z})$ to keep our output value in between 0 and 1.

Loss function:

We are using binary cross-entropy loss function, which is listed as below. For each individual $\hat{y}$, $J(w,b)$ will be smaller when

$$J(w,b) = \frac{1}{m}\sum_{i=1}^{m} y^i * log\hat{y}^i + (1-\hat{y}^i)log(1-\hat{y}^i)$$

m will be the training sample size.

We use 3000 of our 5000 images as a training set, 1000 images as a validation set, and finally 1000 images as a test set. Given the relative small size of our training dataset, we also implemented data augmentation. We applied a rotation, horizontal/vertical shift, shear, zoom, and horizontal flip to each training image, providing us with a much larger data sample to work with. This effectively reduces the the amount of overfitting that occurs during training.

Libraries:

- Numpy
- Matplotlib
- PIL and scipy
- Tensorflow
- Keras

Coding:

- Import training dataset
- Define the model structure
- Initialize the model's parameters
- Create a loop for calculating current loss; calculating current gradient and update parameters.
- Test with test sample

Software:

- Python
- Thunderstorm using ImageJ

5. Timeline and Roles

Task	Deadline	Lead
Data preprocessing/setup	11/10/18	Jeffrey
Design network architecture	11/20/18	Chengyuan
Determine hyperparameters	11/30/18	Linhui
Training	12/5/18	Yu-chun
Prepare report and presentation	12/11/18	all

6. Preliminary Setup

Each of the .jpg images provided to us represents a frame/snapshot of an experiment. Within each frame there are specific DNA binding sites that our model will look at. These features are what are identified in training and testing.

In figure two we see how Thunderstorm identifies each binding site.

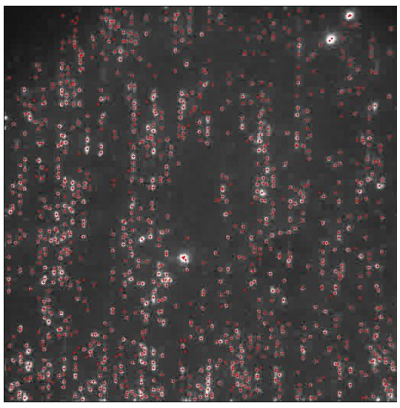

Figure 2. Bacteriophage with identified binding sites.

7. Network Architecture

We are working with a convolutional neural network architecture, and through cross validation, have identified hyperparameters that maximize our classifier's accuracy.

NN Architecture:

Convolutional Layer
|
Max Pooling
|
Convolutional Layer
|
Max Pooling
|
Convolutional Layer
|
Max Pooling
|
Convolutional Layer
|
Max Pooling
|
Flatten
|
Dropout
|
Dense
|
Dense(output)

Activation Function: ReLu/Sigmoid
Loss function - Binary-Cross entropy
Optimizer - RMSprop
Metric - Accuracy

We have tested the following hyper Parameters:
Conv Layers: 2,3,4
Learning rate: 1e-4, 1e-3, 1e-5
Activation function: ReLu, Sigmoid
Epochs: 1,50,100
Dropout: On vs. Off
Strides: 1,2,4
Padding: On vs. Off

Final Hyperparameters used:
Conv Layers: 4
Learning rate: 1e-4
Activation function: ReLu, Sigmoid
Epochs: 1
Dropout: On
Strides: 1
Padding: Off

Conclusion:

This project is focused in modeling a binary classifier that can be used to differentiate between lambda and T7 phages. We construct our model by applying CNN with one input layers, six hidden layers and one output layer. We treat each individual image in the dataset, which is our input, as a column vector and expect to output the probability that the image being either lambda or T7.

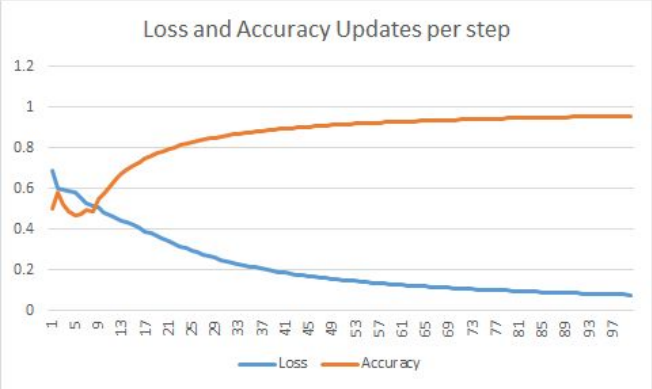

Fig 3. Figure 3 shows how our loss is correlated with accuracy throughout the training of our model.

Based on the test accuracy, this project provides an efficient classifier for lambda and T7 with high-accuracy. The test accuracy we achieved in our project is 95.7%, which is reasonable since all given images are very similar. Therefore, the low deviation in input matrices gives excellent performance in recognizing the image. For future thoughts, if we can obtain datasets of more kinds of bacteria, then we can build a classifier applied to those as well. Moreover, if we can find a efficient way to cut each strand of a DNA, we could switch our input from image to strand, and therefore identify each kind of bacteria that DNA contains.

This project shows how machine learning could be sufficiently applied to different fields as long as we can define the problem clearly and preprocess the data in way that machine can process.

References

[1] Amgarten, Deyvid Emanuel, et al. "MARVEL, a Tool for Prediction of Bacteriophage Sequences in Metagenomic Bins." *Frontiers in genetics* 9 (2018): 304.

[2] Wang, Chen, and Yang Xi. "Convolutional Neural Network for Image Classification." *Johns Hopkins University Baltimore, MD* 21218.

[3] Ovesný, Martin et al. “ThunderSTORM: a comprehensive ImageJ plug-in for PALM and STORM data analysis and super-resolution imaging” *Bioinformatics (Oxford, England)* vol. 30,16 (2014): 2389-90.