Final Project

BMI 203

Winter 2018

github repo: https://github.com/christacaggiano/-neural-net

Assignment- Distinguish real binding sites of a transcription factor (RAP1) from other sequences.

Part 1 - autoencoder

Feed forward 3 layer neural network with standard sigmoidal units.

8x3x8 autoencoder use:

```
from main import autoencoder
matrix_size = 8
print(autoencoder(matrix_size))
```

~~~
Optimization terminated successfully.
         Current function value: 0.000004
         Iterations: 123
         Function evaluations: 202
         Gradient evaluations: 5666
         Hessian evaluations: 0
[[1. 0. 0. 0. 0. 0. 0. 0.]
 [0. 1. 0. 0. 0. 0. 0. 0.]
 [0. 0. 1. 0. 0. 0. 0. 0.]
 [0. 0. 0. 1. 0. 0. 0. 0.]
 [0. 0. 0. 0. 1. 0. 0. 0.]
 [0. 0. 0. 0. 0. 1. 0. 0.]
 [0. 0. 0. 0. 0. 0. 1. 0.]
 [0. 0. 0. 0. 0. 0. 0. 1.]]

 ```
~~~

```
## Part 2 - Learn transcription factor binding sites

I chose to continue to use my artificial neural network (ANN) to learn transcriptio

### encoding
In order to feed DNA sequence information in my ANN, I used one-hot encoding. I ass
nucleotides (i.e `d = {"A": 0, "T": 1, "C": 2, "G": 3}`) to integer encode the sequ
Since this encoding suggests an implicit ranking of the nucleotides, like `G==3 > A
my neural network towards G's. For the purpose of this assignment I assumed that al
```

python

A T C G

[[1. 0. 0. 0.]
[0. 1. 0. 0.]
[0. 0. 1. 0.]
[0. 0. 0. 1.]]

```
Since I have 137 positive test values, the resulting data would be of shape `(137,
`(68, 1)` matrix. In the example above, our matrix would become
```

python
[1. 0. 0. 0. 0. 1. 0. 0. 0. 0. 1. 0. 0. 0. 0. 1.]

```
I chose to break up the negative training label into 17-mers, to match the positive
the beginning of each fasta file, and discarded remaining bases that did not evenly

### neural network design

My ANN takes in a 2D array. I chose to set the hidden layer size at 3, as qualitati

### training

For training, a batch gradient optimization was used. Cost was calculated as the sq

To prevent negative training data from overwhelming the positive data, I just took

### validation

Random samples of the data were chosen for training and testing to evaluate the mod
```

```
I found that the parameter that my model was most sensitive to was the optimization

Sample output for `hidden_size = 3, num_subsample=500` and 50% of the data reserved
```

python

Optimization terminated successfully.

Current function value: 0.000004

Iterations: 23

Function evaluations: 27

Gradient evaluations: 199

Hessian evaluations: 0

[[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]

[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.001]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]

[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]

[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]

[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]
[0.]

```
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 0. ]
[ 1. ]
[ 1. ]
[ 1. ]
[ 1. ]
[ 1. ]
[ 1. ]
```

[1.]
[1.]
[1.]
[0.983]
[1.]
[1.]
[1.]
[1.]
[1.]
[1.]
[1.]
[1.]
[1.]
[1.]
[1.]
[1.]
[1.]
[1.]
[1.]
[0.999]
[1.]
[1.]
[1.]
[1.]
[1.]
[1.]
[0.996]
[1.]
[1.]
[1.]
[1.]
[1.]
[1.]
[0.999]
[1.]
[1.]
[1.]
[1.]
[1.]
[0.923]
[1.]
[1.]

```
[ 1. ]
[ 1. ]
[ 1. ]
[ 1. ]
[ 1. ]
[ 1. ]
[ 1. ]
[ 1. ]
[ 1. ]
[ 0.999]
[ 0.637]
[ 1. ]
[ 1. ]
[ 1. ]
[ 1. ]
[ 1. ]
[ 1. ]
[ 1. ]
[ 1. ]
[ 0.978]
[ 1. ]
[ 1. ]
[ 1. ]
[ 1. ]]

Process finished with exit code 0
```

Sources consulted:

- Stephen Welch, Neural Network Demystified
- Suriyadeepan Ram, The Principle of Maximum Likelihood
- @giantneuralnet, Beginner Intro to Neural Networks
- Jon Como, Flowers, a simple neural net tutorial
- Sebastian Rudder, An overview of gradient descent algorithms
- Erik Lindernoren, ML From Scratch