National Chiao Tung University
Spring 2019
Deep Learning
Instructor: Jen-Tsung Chien

Deep Learning HW1 Report

Alfons Hwu
Student ID: 0416324

alfons.cs04@g2.nctu.edu.tw
Dept of Comuter Science
Writing with LaTeXon Overleaf

Deep Learning HW1 Report

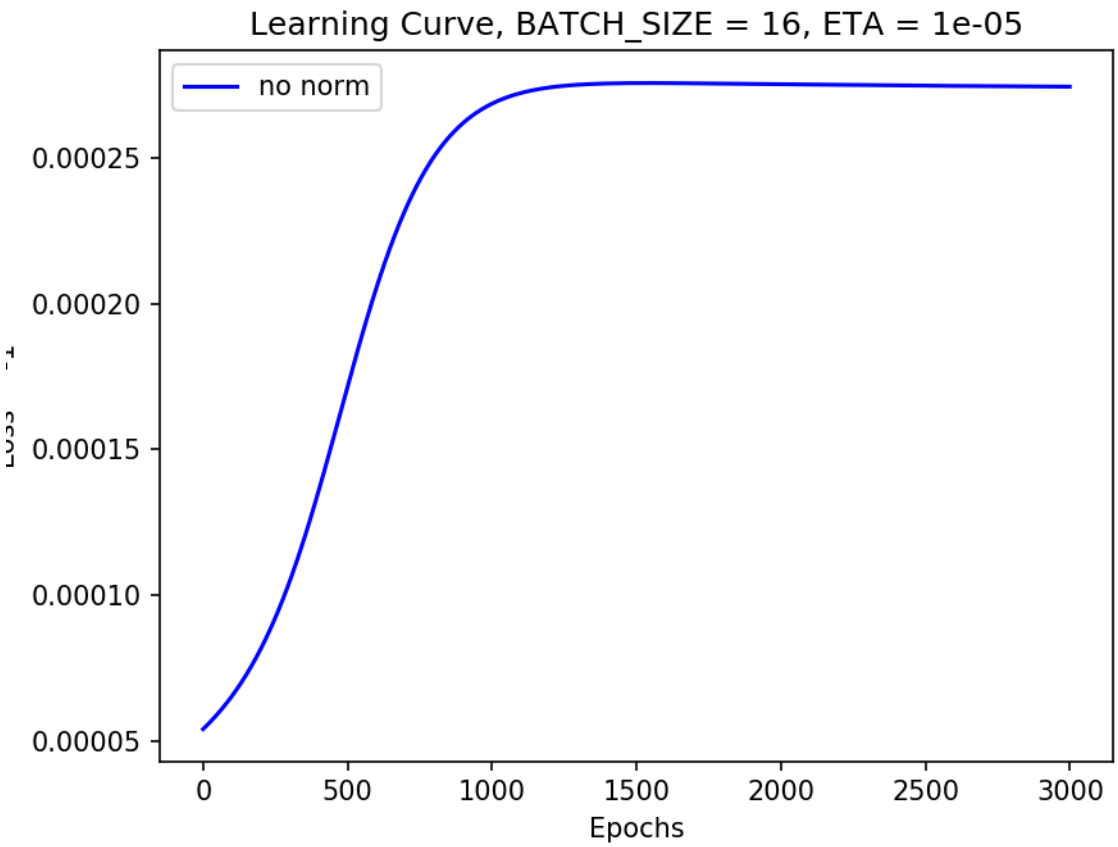
April 5, 2019

1 Self-designed DNN for binary classification

Loss is defined by

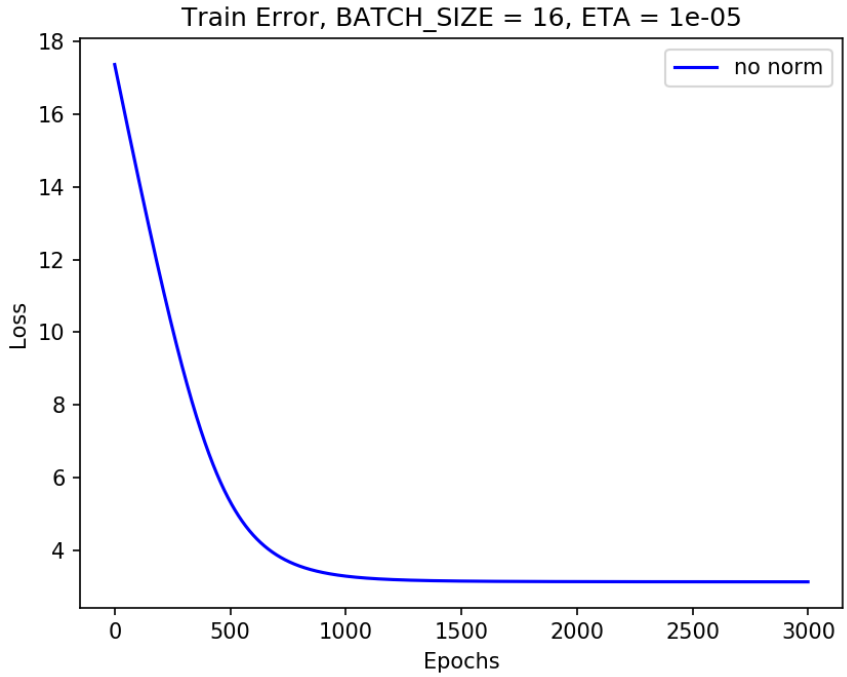
$$E(w) = \sum_{n=1}^{N} \sum_{k=1}^{K} = t_{nk} \ln y_k(X_n, w)$$

1.1 Learning Curve



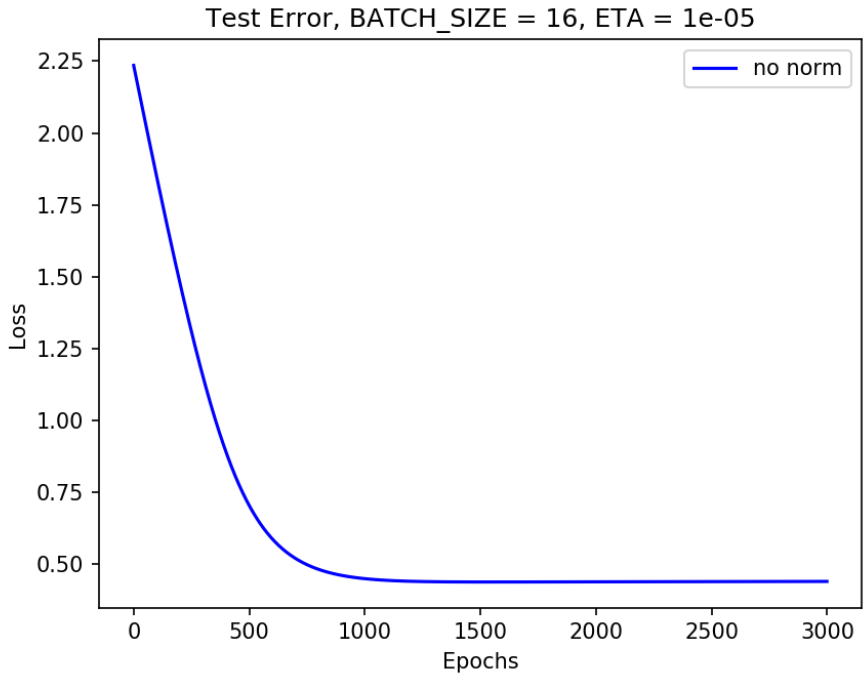
The Y-axis represents 1/Loss, telling the learning curve(loss is dropping and finally converge)

1.2 Training Loss



The Y-axis represents loss, telling the learning curve(loss is dropping and finally converge)

1.3 Testing Loss



The Y-axis represents loss, telling the learning curve(loss is dropping and finally converge)

1.4 Explanation of my neural network architecture

- Layer: Input 6unit (input including 6 features, which is 6 dimensions).
  Hidden 4unit, since in my opinion, what influences the result most lies in the learning rate, the neuron set 4 will be optimal for computation (too many neurons merely increases the matrix computation time)
  Click for ref
  Output: 1unit for alive or death.
  For the following two items, I select them based on the combination with shell script, and comparing the optimal results using the **same random seed** for benchmarking the learning curve vs batch size and learning rate. (Choose the normal learning rate first (such as 0.5) and slightly decrease by empirical method (automated training with shell script and comparing with pyplot figure).
  According to this article the **high learning rate will cause the non-convergent of learning curve.**
  With the aforementioned site, I start up with learning rate 0.5 and empirically dropped down to about **0.00001 +- 0.000005**

```
if [ $sel -eq 1 ];
then
    batch_size=$2
    for learning_rate in 0.000001 0.000003 0.000005 0.000008 0.00001 0.000
    do
        echo $learning_rate
        python3 dnn_1.py $1\_$batch_size\_$learning_rate $batch_size $lea
    done
else
    python3 dnn_1.py $1_8_0.000005 8 0.000005
fi

mkdir -p $1_1
mv $1*\.png $1_1/
```

- Batch size: 16 for optimal, (too large will consume too much memory resource, and too small will cause the unstable learning curve. Although more randomness, less chance to converge)
  Ref link

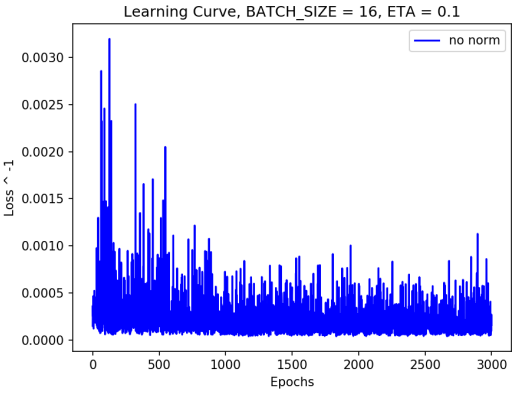


Figure 1: Learning rate = 0.1

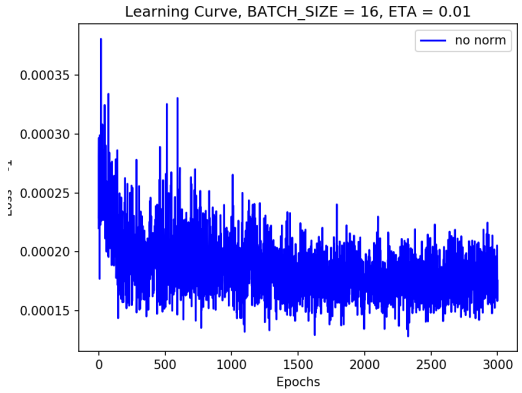
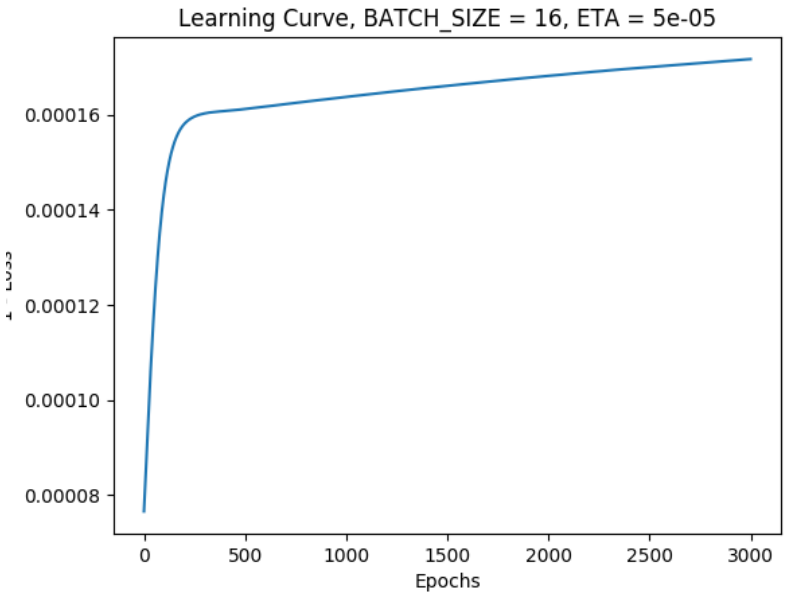


Figure 2: Learning rate = 0.01

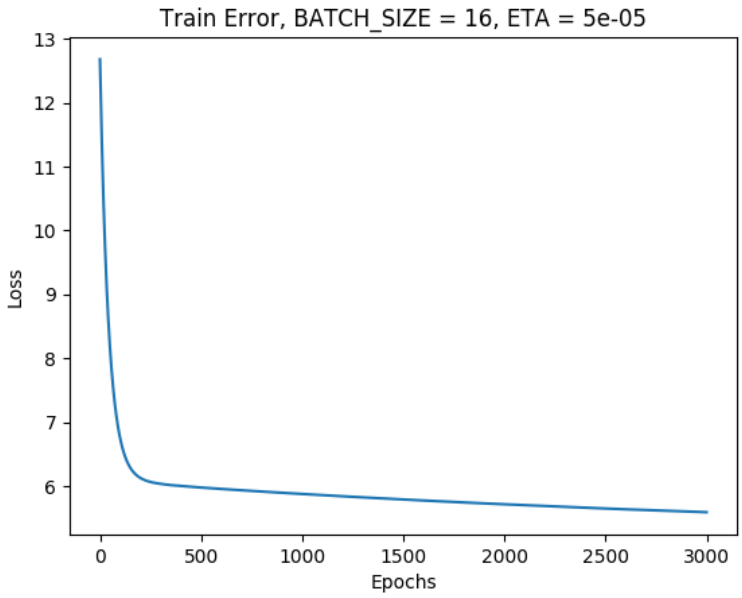
- Learning rate: 0.00001 (1e-5) will be optimal (The above figures shows some high learning rate causing non-convergence of learning curve)

2 TA-designed DNN

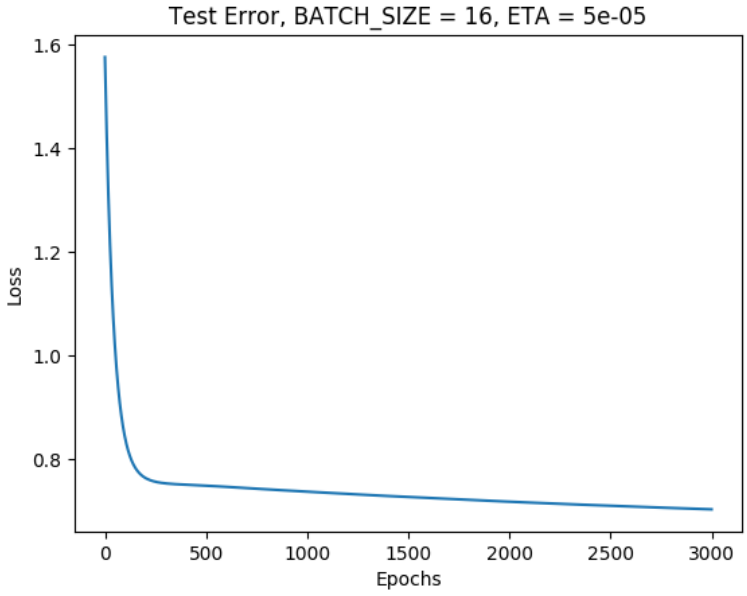
The selection of learning rate and training batch size is the same as above



Learning curve



Training error



Testing error

2.0.1 And a subsubsection