**CS535 Deep Learning , Assignment 2, Dongkyu Kim**

**1) and 2) → please see the source code.**

**3) The best hypter-parameter tuned.**

Below parameters were decided by the results of 5) Turing Parameters.

mini_batch_size = 64
learning_rate = 0.0008
hidden_units = 2000
seed = 10
num_epochs = 100
momentum = 0.9

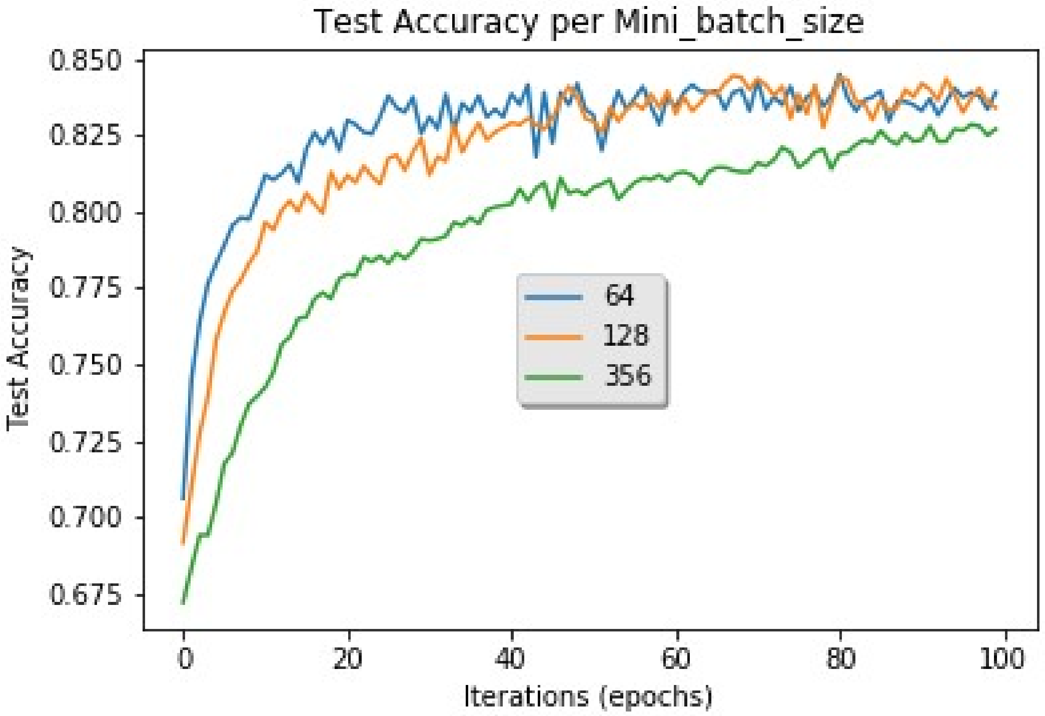
```
Train[Epoch 75  mini-batch 156  Cost = 5.849]
Train[Avg.cost: 0.0005849087498108696   Accuracy: 99.77%]
Test[Cost: 0.4332420980283863   Accuracy: 85.75%]


 ..............................

Train[Epoch 99  mini-batch 156  Cost = 3.335]
Train[Avg.cost: 0.00033352909675913553   Accuracy: 99.97%]
Test[Cost: 0.4684568376083723   Accuracy: 85.35%]
```

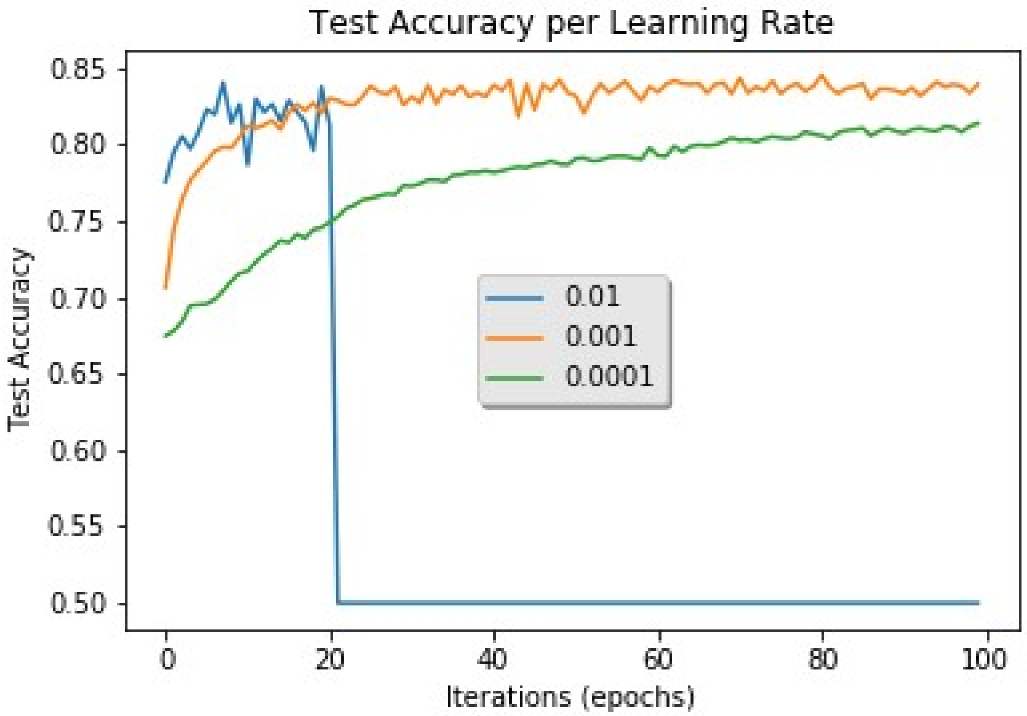
**4) below is a part of the results**

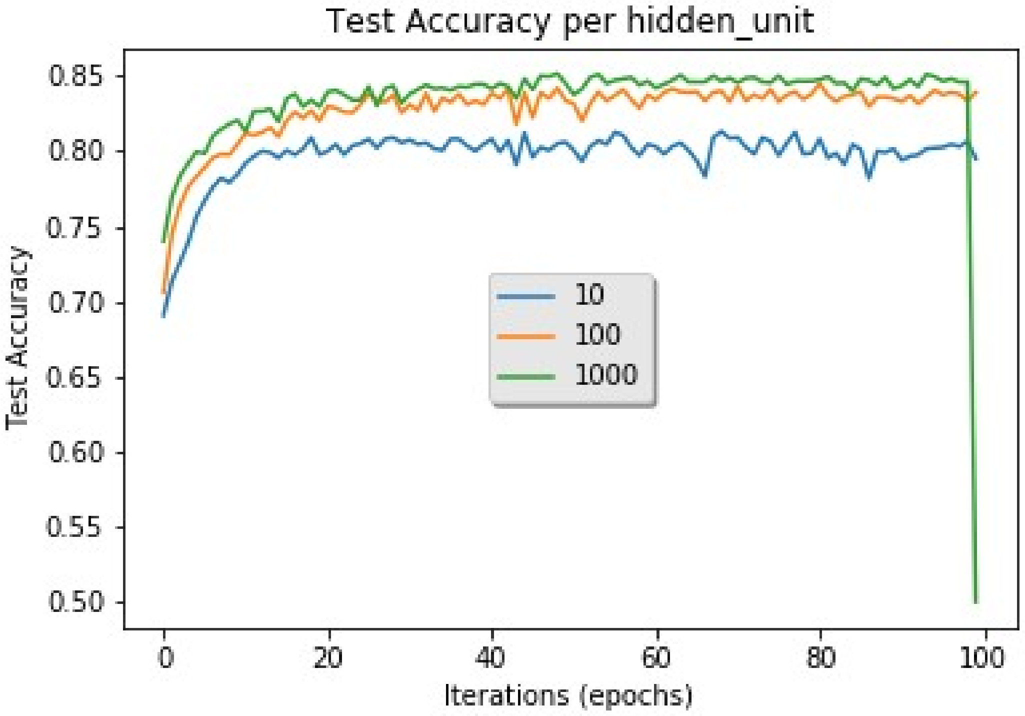
```
 ..............................
Train[Epoch 62  mini-batch 156  Cost = 9.902]
Train[Avg.cost: 0.00099018636592678   Accuracy: 99.15%]
Test[Cost: 0.4216214591585031   Accuracy: 84.55%]

Train[Epoch 63  mini-batch 156  Cost = 9.712]
Train[Avg.cost: 0.0009711819528149317   Accuracy: 99.13%]
Test[Cost: 0.41480091597693647   Accuracy: 85.30%]

Train[Epoch 64  mini-batch 156  Cost = 9.411]
Train[Avg.cost: 0.0009411072369252203   Accuracy: 99.23%]
Test[Cost: 0.41709766101330514   Accuracy: 85.10%]

Train[Epoch 65  mini-batch 156  Cost = 9.002]
Train[Avg.cost: 0.0009001835106140099   Accuracy: 99.36%]
Test[Cost: 0.42023565611280844   Accuracy: 84.30%]

Train[Epoch 66  mini-batch 156  Cost = 8.834]
Train[Avg.cost: 0.000883444180390107   Accuracy: 99.33%]
Test[Cost: 0.4296430719418355   Accuracy: 84.35%]

Train[Epoch 67  mini-batch 156  Cost = 8.584]
Train[Avg.cost: 0.0008584292426807811   Accuracy: 99.42%]
Test[Cost: 0.421883786647761   Accuracy: 85.25%]
 ..............................
```

**CS535 Deep Learning , Assignment 2, Dongkyu Kim**

## 5) Tuning Parameters







## 6) Discussion

- Vanishing gradient

  As you can see above figures regarding to learning rates and hidden units in 5), the accuracy drops to 50%. Also, the information of training-testing monitoring shows as below:

  ```
  Train[Epoch 22  mini-batch 156  Cost = nan]
  Train[Avg.cost: nan   Accuracy: 50.00%]
  Test[Cost: nan   Accuracy: 50.00%]

  Train[Epoch 99  mini-batch 156  Cost = nan]]
  Train[Avg.cost: nan   Accuracy: 62.13%]
  Test[Cost: nan   Accuracy: 50.00%]
  ```

  Cost values become NaN, and it could be an evidence of vanishing gradient problem. The problem is that in some cases, the gradient will be vanishingly small, effectively preventing the weight from changing its value. In the worst case, this may completely stop the neural network from further training.

- Since it performs mini-batch gradient descent, graphs do not show linear lines. Each batch has a different a cost(loss) value, thus its accuracy could be fluctuated. When mini-batch size becomes the total sample size, it works as a batch gradient descent, and it will take too long per iteration. On the other hand, when mini-batch size is too small, it works as a stochastic gradient descent, and  it will lose speed up from the vectorization. The result also shows a tendency that the accuracy goes up until certain points as iteration(epochs) increases.

**CS535 Deep Learning , Assignment 2, Dongkyu Kim**

- In terms of the learning rate change, the bigger a learning rate is, then the wider fluctuation is, but it reaches to a minimum area but not exactly the minimum earlier. It might oscillate around the minimum(optimal) point. On the other hand, when a learning rate is small, it searches the minimum point sophisticatedly. Thus, it reaches the higher accuracy slowly as shown in the figure, and it reduces the effect of the momentum.

- Regarding hidden units, the last figure in 5) shows that when the number of hidden units increases, the neural network's performance will be improved.

- Considering these findings and several experiments with other parameters such as momentum and number of epochs, I derived the best tuned parameters as shown in 3)