Trabalho Prático I

Igor Lacerda Faria da Silva

igorlfs@ufmg.br

1 Introdução

O objetivo deste trabalho é montar uma rede neural para identificar dígitos escritos à mão, a partir do banco de dados do `MNIST`. Este relatório compara diferentes variações dessa rede neural. A rede possui apenas 3 camadas e as variações exploram diferentes algoritmos, taxas de aprendizado e número de neurônios na camada oculta.

A comparação é dividida em 3 partes: inicialmente são fixados o algoritmo e a taxa de aprendizado, modificando-se o tamanho da camada oculta. Similarmente, a taxa é alterada e, por fim, o algoritmo. Conforme a especificação, os 3 algoritmos implementados foram: *Stochastic Gradient Descent*, *Gradient Descent* e *Mini-Batch*. A implementação consistiu apenas em modificações de parâmetros simples da biblioteca.

A biblioteca utilizada foi o `TensorFlow`. Ela permite carregar diretamente o banco de dados do MNIST e possui uma `API` conveniente para criar os modelos.

2 Desenvolvimento

Foram analisadas duas métricas: a acurácia e a perda.

2.1 Tamanho da Camada Oculta

Foi fixada uma taxa de aprendizado $l_R = 1$ e um algoritmo de *Mini-Batch*, com o tamanho da *batch* igual a 50. A escolha desses parâmetros foi empírica: dentre as opções descritas no enunciado, o desempenho final da rede é intermediário.

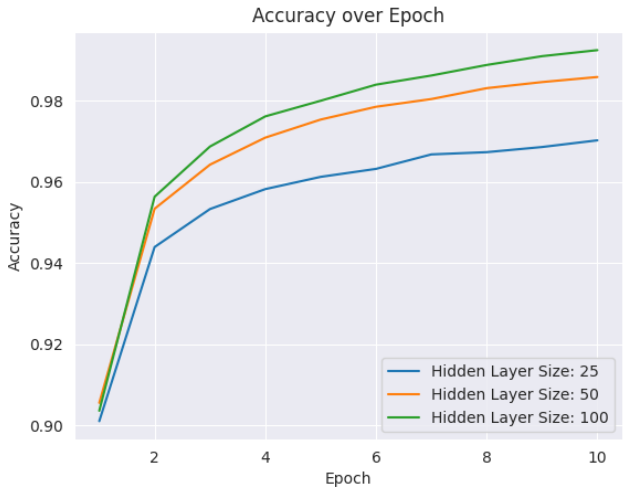

Figura 1: Acurácia por época, variando o tamanho da rede.

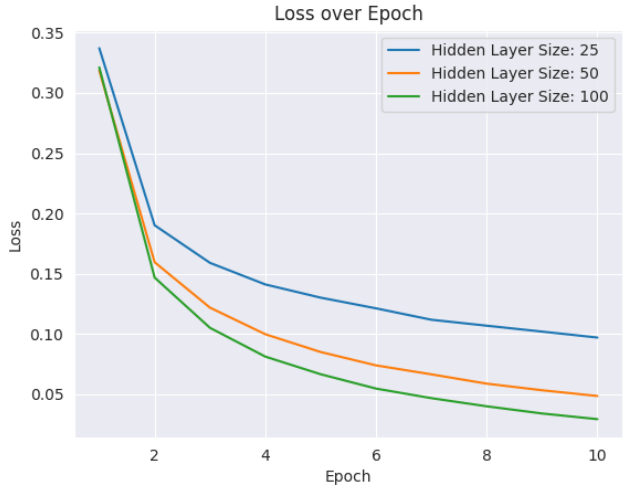

Figura 2: Perda por época, variando o tamanho da rede.

Como é de se esperar, o aumento do tamanho da rede melhorou a performance. Com mais neurônios é possível fazer ajustes mais sensíveis aos dados. De fato, o desempenho da rede é tão alto para a camada com 100 neurônios que é possível que tenha ocorrido um *overfitting*.

2.2 Taxa de Aprendizado

Foi fixada um tamanho de camada oculta igual a 50, e o algoritmo de *Mini-Batch*, também com tamanho de *batch* igual a 50. Novamente, a escolha desses parâmetros foi experimental, com base nas alternativas da especificação.

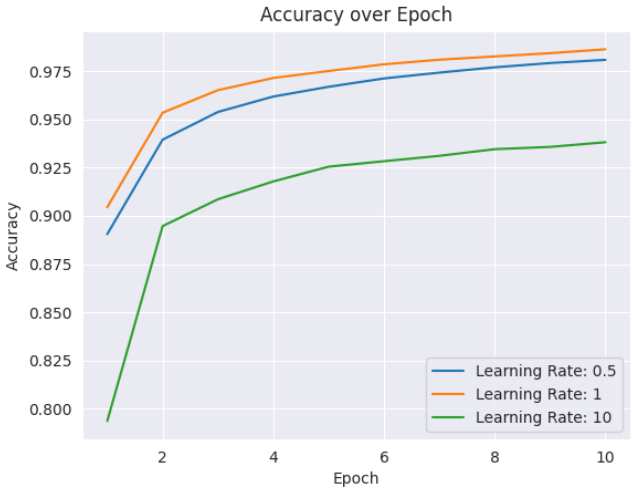

Figura 3: Acurácia por época, variando a taxa de aprendizado.

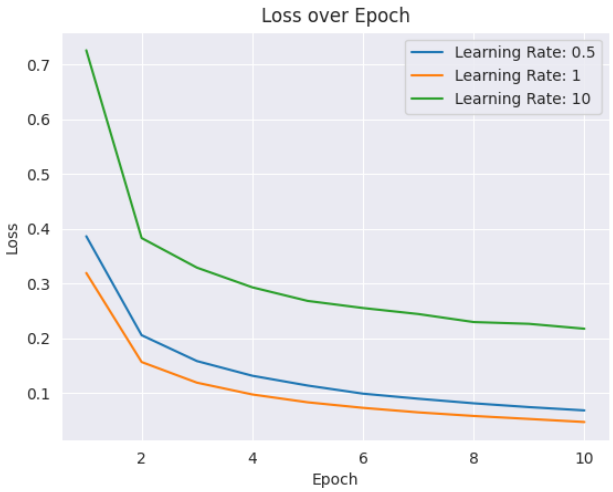

Figura 4: Perda por época, variando a taxa de aprendizado.

Como visto em aula, uma taxa de aprendizado muito alta pode prejudicar a qualidade do modelo. De fato, as taxas tradicionalmente usadas são muito

menores do que as descritas na especificação: na faixa de 0.0001 a 0.01. Uma taxa de aprendizado muito baixa pode aumentar o número de épocas para convergência. Por outro lado, uma taxa de aprendizado muito alta pode fazer o modelo não convergir. Esse não foi o caso, apesar de que para a taxa igual a 10, é nítido o efeito na qualidade do modelo.

2.3 Algoritmo

As variações de algoritmo foram controladas alterando-se o parâmetro *batch_size*, da função *fit()* do modelo. Em princípio, todos os algoritmos são o *Mini-Batch*. Para criar um SGD, basta usar uma *batch* de tamanho 1 e para criar um GD, é preciso usar uma taxa de aprendizado igual ao número de dados de treino. O número de épocas também precisou ser ajeitado: é preciso dar mais oportunidades de ajustes no GD, porque seus ajustes são muito infrequentes. Além disso, o tempo de execução do SGD é muito grande. Nos gráficos, o eixo x representa o "progresso" do algoritmo (normalizado pelo número de épocas), cada ponto corresponde a uma época. Para os algoritmos de SGD e MB foram feitas 15 épocas e, para o SGD, 100 épocas. A taxa de aprendizado, fixada em 1, e o número de neurônios, fixado em 50 foram escolhidos experimentalmente, como nos outros casos.

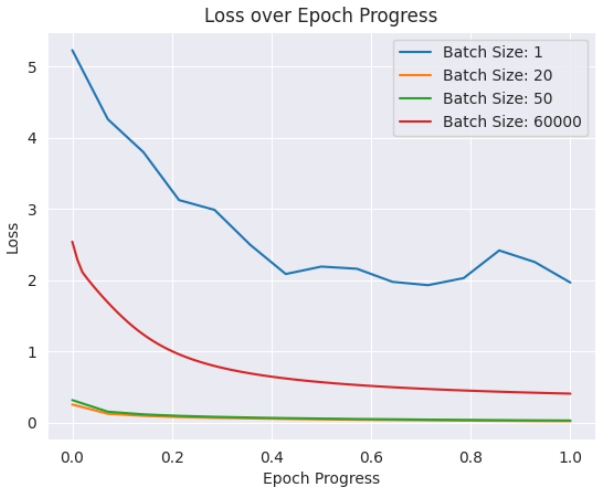

Figura 5: Acurácia por progresso em épocas, variando o algoritmo.

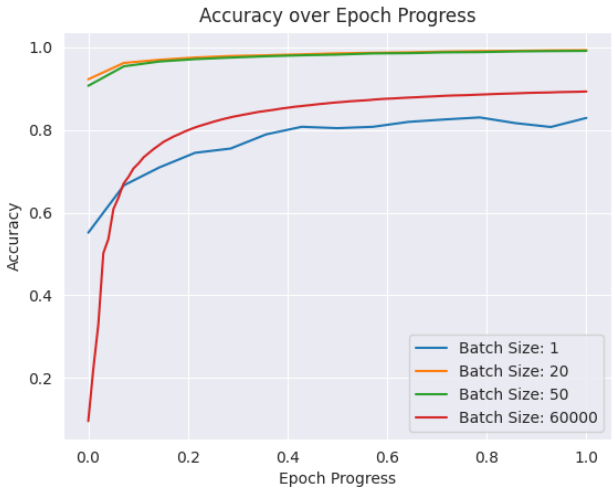

Figura 6: Acurácia por progresso em épocas, variando o algoritmo.

Todos os algoritmos tiveram um desempenho razoável. É possível que, com mais épocas, o SGD se saísse melhor, mas infelizmente ele demanda muito tempo de execução. Similarmente, com mais épocas, o GD poderia ter se saído um pouco melhor. No entanto, o MB certamente é superior nesse caso, convergindo muito mais rapidamente, independentemente do tamanho do *batch*. Isso acontece pois o MB é um equilíbrio razoável entre o acúmulo de ajustes constantes aos pesos com a quantidade de dados usada para fazer esses ajustes.