MATE21 - 2018.2*

Adeilson Silva

I. Utilização das imagens

As imagens da base de dados possuem dimensão de 71 x 77 pixels, portanto, para serem servidas às redes construídas, elas foram transformadas em vetores unidimensionais de 5467 características. As imagens foram carregadas em escala de cinza e normalizadas para o intervalo [0, 1].

O subset de treino foi particionado, sendo 75% das imagens utilizadas para treino e os outros 25% para validação.

II. Estrutura das redes criadas

A. Regressão Logística

Para a implementação da regressão logística foi criada uma rede com uma única camada, que recebe como entrada uma imagem e retorna a classe estimada para aquela imagem. Para tal, a função softmax foi utilizada como função de ativação:

$$\sigma(z) = \frac{e^z}{\sum_{k=1}^{K} e^{z_k}} \tag{1}$$

A equação (1) permite transformar a saída da regressão linear em uma distribuição de probabilidades (*i. e.* a soma do vetor de saída do softmax é igual a 1.).

Para avaliar a proximidade entre as predições feitas pela rede e a classe real das imagens de treino, a função de entropia cruzada foi utilizada, como visto na equação (3). Para utilizar esta função, criamo um vetor *one_hot*, que consiste em um vetor com todas as posições zeradas, exceto na posição correspondente à classe real da observação.

$$CE = -\sum_{k=1}^{N} y_i * log(\hat{y}_i) \tag{2}$$

Nesta etapa, foi utilizado a versão *mini batch* do gradiente descendente, com um batch contendo 32 imagens aleatórias do conjunto de treino. Para cada época, o batch é resorteado e o gradiente acumulado em relação as imagens do batch. Após isso, utilizamos o gradiente para atualizar os pesos e o bias.

Na figura 1 podemos visualizar a relação entre a quantidade de épocas de treino e o decrescimento da função de custo, além do aumento da acurácia do sistema.

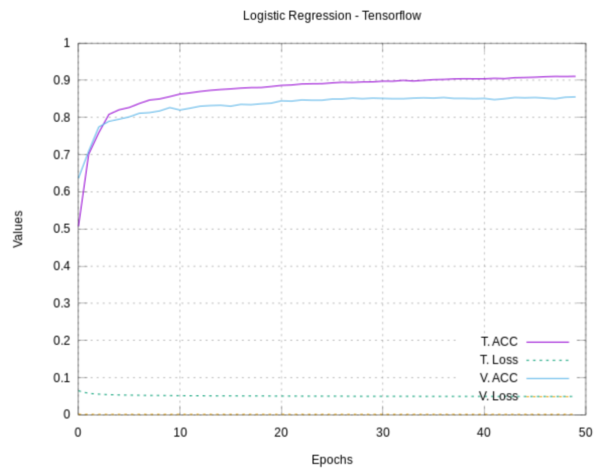

Figura 1. Relação entre custo, acurácia e quantidade de épocas.

B. Multilayer Perceptron

Para esta implementação, foi criada uma rede com uma camada de entrada, uma camada de saída e uma camada oculta. Para a camada oculta, a função sigmoide foi utilizada como função de ativação:

$$S(x) = \frac{1}{1 + e^{-x}} \tag{3}$$

Desta vez utilizamos a ideia de *backpropagation* para realizar a atualização dos pesos em todas as camadas da rede. Isto consiste em atravessar todas as camadas, calcular o erro obtido e, utilizando a derivadas em cadeia, projetar os erros obtidos caminhando do final para o início da rede.

Vemos na 2 como a função de custo comportou-se em relação ao passar das épocas.

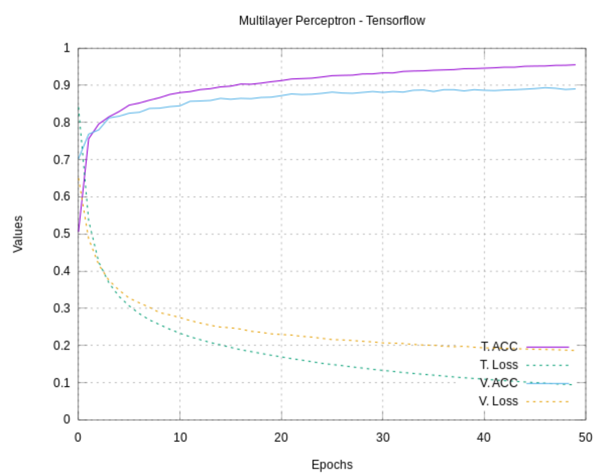

Figura 2. Relação entre custo, acurácia e quantidade de épocas.