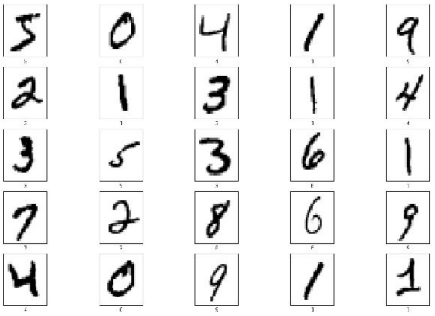
Bruno Bacelar Abe
9292858

Classificando a base MNIST, utilizando Tensorflow juntamente com Keras

Baseando-se no tutorial dado no enunciado do trabalho, criei uma aplicação que prediz dígitos depois de um treinamento, em uma CNN.
Primeiramente, houve a importação da base de dados MNIST, dividindo-a em duas, uma para testes e outra para treinamento. Na figura abaixo, podemos ver os 25 primeiros números da base importada.



(Figura 1: Base de treino)

Com as bases em mãos, houve a configuração da rede para que o treinamento fosse realizado. Nessa rede, utilizei a função de ativação ReLU e Softmax para extração de características e avaliação.

```
model = keras.Sequential()
model.add(keras.layers.Conv2D(64, kernel_size=(2, 2),
                 activation='relu',
                 input_shape=(input_shape)))

model.add(keras.layers.Conv2D(128, (2, 2), activation='relu'))
model.add(keras.layers.MaxPooling2D(pool_size=(4, 4)))
model.add(keras.layers.Dropout(0.25))

model.add(keras.layers.Flatten())
model.add(keras.layers.Dense(256, activation='relu'))
model.add(keras.layers.Dropout(0.5))

model.add(keras.layers.Dense(num_classes, activation='softmax'))
```

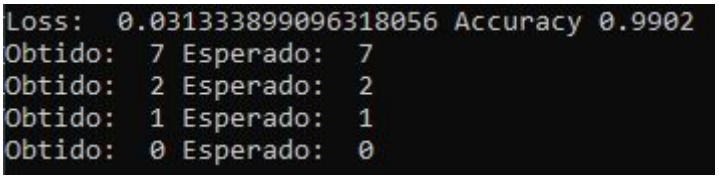
(Figura 2: Configuração do modelo para treino)

No exercícios, são pedidos os seguintes itens:

1. Alterar a quantidade de mapas de características em cada camada;
2. Modificar o tamanho dos filtros na cama de convolução;
3. Adicionar uma camada totalmente conectada.

Como é possível analisar na imagem acima, há uma camada totalmente conectada (a última), enquanto as outras possuem um dropout. Os tamanhos dos filtros também foram modificados.
Após realizar os ajustes na rede, é necessário compila-la , para então realizar o treinamento. Estou usando o otimizador “Adadelta”.
Após o treino, apuramos a acurácia do modelo treinado - que superou os 98% na maior parte dos testes - e também realizamos a inferência de dados para predição.A inferência foi

realizada em 4 números diferentes (7, 2, 1 e 0). Para realizar a inferência, utiliza-se o método “model.predict(Base de teste)”, que retorna um vetor com todos os resultados. Para cada um dos resultados, houve a comparação com o resultado esperado.




(Figura 3: Resultados obtidos)

Para o treinamento da rede, utilizei 5 épocas, Python 3.6.5, Windows 10 x64 e uma GPU Nvidia GeForce 1050ti.