Universidade Federal do Ceará
Campus de Itapajé

Redes Neurais Artificiais

TRABALHO COMPUTACIONAL
Mini-projeto I: Reconhecimento de Digitos com MLP

Prof. Dr. Hitalo Nascimento

hitalo.nascimento@ufc.br

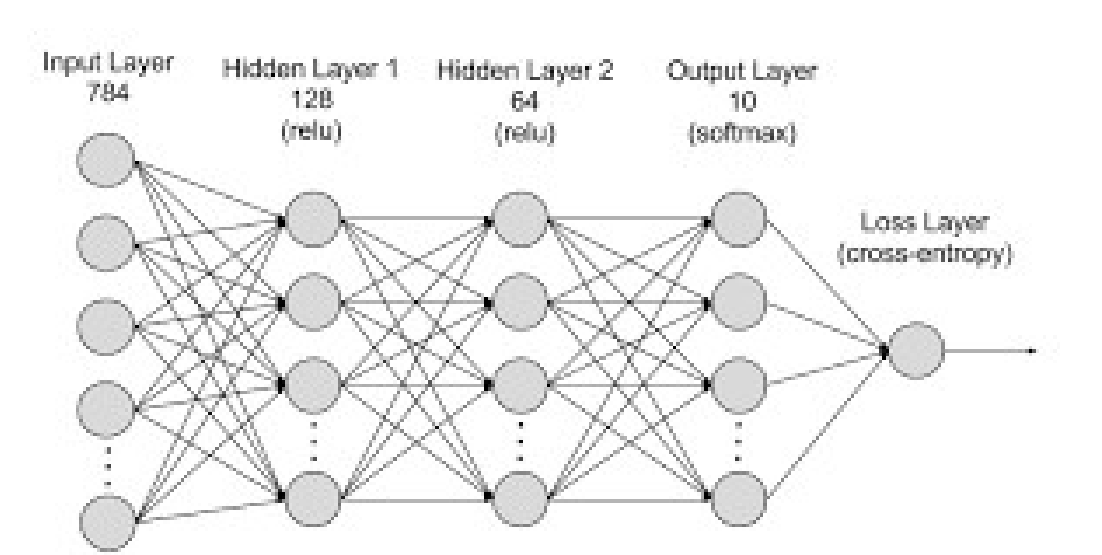
Descrição

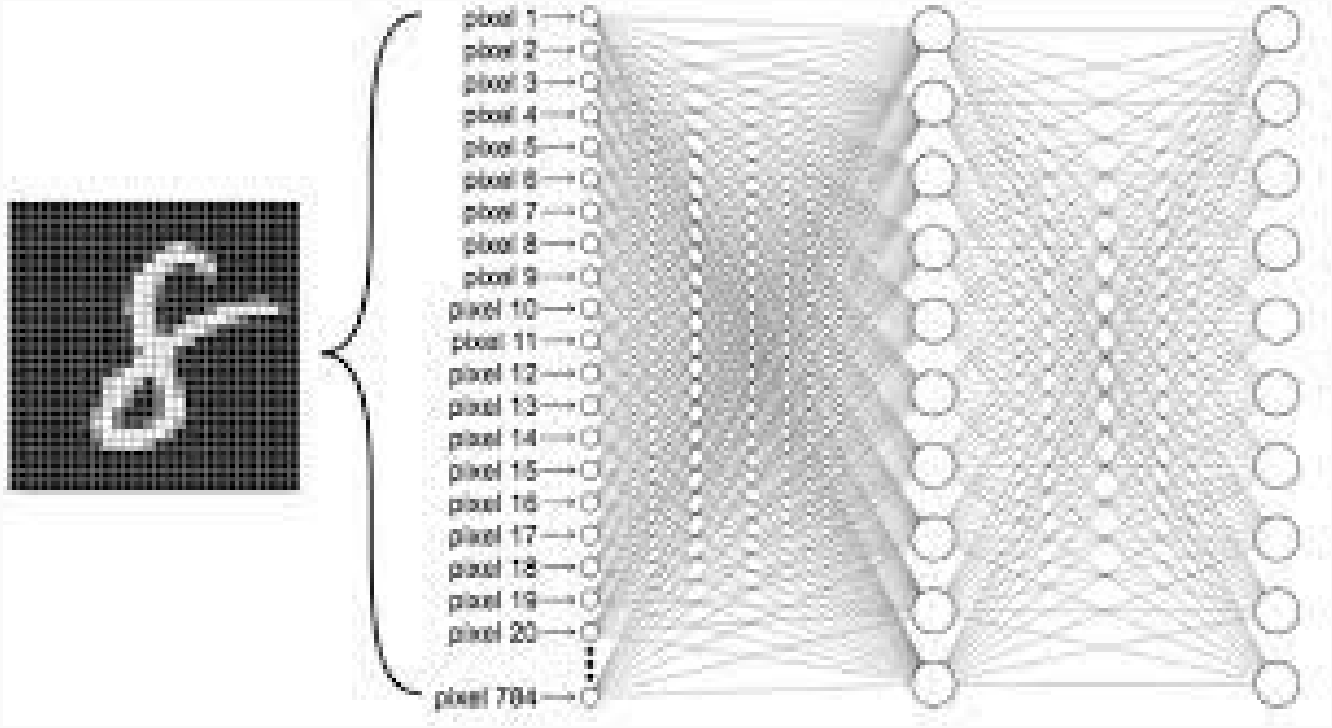
- Reconhecimento de dígito a partir de uma rede MLP;
- Implementar uma RNA MLP para classificar imagens entre 10 classes possíveis, em que e cada classe representa um digito manuscrito entre 0 e 9. Na etapa de treinamento e teste deverá ser utilizado o dataset MNIST, que consiste de uma base de dados de dígitos padronizadas para caber em uma matriz 28×28 pixels, onde cada pixel representa um nível de escala de cinza. O MNIST contém 60.000 imagens de treino e 10.000 imagens de teste.

Descrição

- Abaixo um imagens do conjunto de treinamento do MNIST.

Descrição


Input Layer
784
Hidden Layer 1
128
(relu)
Hidden Layer 2
64
(relu)
Output Layer
10
(softmax)
Loss Layer
(cross-entropy)


pixel 1
pixel 2
pixel 3
pixel 4
pixel 5
pixel 6
pixel 7
pixel 8
pixel 9
pixel 10
pixel 11
pixel 12
pixel 13
pixel 14
pixel 15
pixel 16
pixel 17
pixel 18
pixel 19
pixel 20
pixel 784

Descrição

Mais detalhes sobre esse dataset pode ser obtidos em: http://yann.lecun.com/exdb/mnist/

Além disso, foi disponibilizado no google class, os aquivos em csv para treinamento e treino da RNA.

Descrição

É possível ainda importar o dataset em questão usando o framework keras, Veja o exemplo completo:

```
from keras.datasets import mnist
from matplotlib import pyplot

#loading
(train_X, train_y), (test_X, test_y) = mnist.load_data()

#shape of dataset
print('X_train: ' + str(train_X.shape))
print('Y_train: ' + str(train_y.shape))
print('X_test:  '  + str(test_X.shape))
print('Y_test:  '  + str(test_y.shape))

#plotting
from matplotlib import pyplot
for i in range(9):
pyplot.subplot(330 + 1 + i)
pyplot.imshow(train_X[i], cmap=pyplot.get_cmap('gray'))
pyplot.show()
```

Descrição

A Rede neural deve ser treinada até que consiga atingir uma acurácia de pelo menos 96% em relação aos testes.

Parte I: Defina e implemente os componentes necessários para a rede neural (784 unidades de processamento referentes às entradas para o treinamento da rede; 20 unidades de processamento referentes às camadas escondidas; 10 saídas para os valores de 0 a 9; taxa de aprendizado; função de custo; o número de iterações e épocas);

Descrição

Parte II: Implemente a propagação para frente e calcule o custo;

Parte III: Calcule o gradiente da função de custo;

Parte IV: Implemente a retropropagação. (Use a regra de atualização para o gradiente descendente);

Parte V: Teste diferentes números de camadas intermediárias, e verifique se há uma melhora na qualidade dos resultados no teste. Qual o número de camadas que ideal?

Descrição

- O trabalho pode ser feito em dubla;
- Deve ser apresentado em sala de aula até o dia 16/05;
- Deve ser implementado em Python;
- É necessário evidenciar a execução do código.

Referências

http://yann.lecun.com/exdb/mnist/

https://www.askpython.com/python/examples/load-and-plot-mnist-dataset-in-python

https://en.wikipedia.org/wiki/MNIST_database

https://towardsdatascience.com/exploring-how-neural-networks-work-and-making-them-interactive-ed67adbf9283

http://neuralnetworksanddeeplearning.com/chap1.html