**Relatório Trabalho 01 - Redes Neurais Convolucionais para Visão Computacional**

Marcelo Cabral Ghilardi

Rodrigo/Jônatas

Fiz a implementação solicitada de uma rede de 2 camadas, nos arquivos **Neuralnet.ipynb** e **neural\_net.py**.

O notebook **Neuralnet.ipynb** pode ser rodado por inteiro se quiser, e no final há a leitura do meu melhor modelo com a acurácia de (Treino: 0.89605%, Validação: **0.582%**).

Tenho 249 modelos salvos, ocupam 15GB, estou enviando apenas o melhor.

Meu melhor modelo e seu treinamento se baseou na rede com 2 camadas. Mas tenho uma implementação teste de 3 camadas em **Neuralnet3.ipynb** e **neural\_net3L.py**. Todavia, não treinei o suficiente para atingir melhores resultados, portanto, o que vale é a rede de 2 camadas.

Todo o treinamento e suas etapas estão no arquivo **desafio.py**, foram testados modelos com:  
hidden\_size: 50,100,200,400,800,1600,3200,4000  
optimizer: SGD, Adagrad e Adam  
dropout, learning\_rate, learning\_rate\_decay, reg: muito variado.

Foi descrever abaixo a implementação da nova função da treinamento e como cheguei no melhor modelo.

**NOVO MÉTODO DE TREINAMENTO**

No arquivo **neural\_net.py** há dois métodos de treinamento:  
def **train**( ... ) => que possui suas características originais do trabalho

e um novo chamado:  
def **trainMarcelo**(self, X, y, X\_val, y\_val,   
**epochs**=100, //Número de épocas  
batch\_size=200, learning\_rate=1e-3, learning\_rate\_decay=0.95, reg=1e-5, verbose=False,   
**earlyStopping**=0, //para em X épocas sem melhoras de LossVal LossTrain AccVal   
**optimizer**="SGD", //SGD, AdaGrad, Adam  
**dropout**=0.0) //Dropout  
  
No método **trainMarcelo**, diferente do método **train**, além dos novos parâmetros, a cada época todos os dados de treinamento são lidos, reorganizados de forma randômica, e separados em batchs também de forma randômica.

Outra coisa importante, é que o retorno do treinamento é sempre os pesos da melhor época, e não da última época.

**MELHOR MODELO**

Meu melhor modelo foi treinado com 3200 neurônios, optimizador AdaGrad, os pesos foram iniciados de outro modelo, que teve seus pesos iniciado de outro, e assim por diante.

A ordem toda, de forma decrescente, foi:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nome do modelo** | **Épocas** | **Optimizador** | **LearnRate** | **Reg** |
| model\_0.582000\_3200\_0.654387\_0.000032\_25-R7 | 25 | AdaGrad | 0.654387 | 0.000032 |
| model\_0.581500\_3200\_0.689206\_0.000664\_75-R6 | 75 | AdaGrad | 0.689206 | 0.000664 |
| model\_0.580800\_3200\_0.768194\_0.000031\_21-R4 | 21 | SGD | 0.768194 | 0.00031 |
| model\_0.580100\_3200\_0.678687\_0.000673\_66-R3 | 66 | SGD | 0.678687 | 0.000673 |
| model\_0.575500\_3200\_0.750000\_0.002000\_135 | 135 | SGD | 0.75 | 0.002 |
| TOTAL | 322 \* |  |  |  |

\* 322 até o melhor modelo, mas foram mais 10 ou 15 épocas em cada treinamento.