Classificação de Série Temporal

João Vicente Dornas

13/04/2020

Sobre o Modelo:

Uma pesquisa realizada no artigo [1] envolvendo milhares de modelos de *DeepLearning* para classificação de séries temporais, ficou evidente a melhor performance do modelo citado no artigo [2]. Esse modelo se compõe de três camadas compostas em sequência por uma convolução de 1 dimensão, um *batch normalization* e uma ativação ReLu. Após essas três camadas, aplica-se um *Global Pooling* e por último a função *Softmax*. Abaixo segue ilustração do próprio artigo:

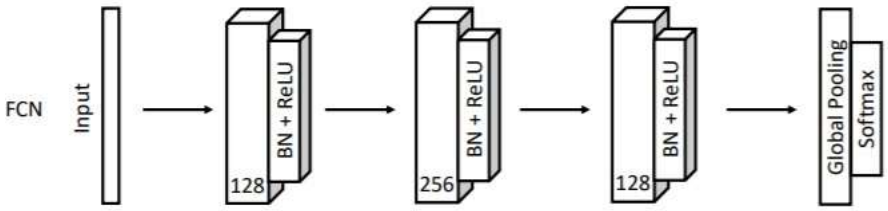

Figure 1: Modelo FCN.

Esse modelo é conhecido como *Fully Convolutional Nerwork*.

Sobre o Algoritmo:

Foi desenvolvida uma solução com Visual Studio 2017 usando um projeto Python 3.6. As bibliotecas usadas foram as seguintes:

Tkinter (inslado junto com o Python for Windows)

Opencv-python (cv2) (4.2.0.34)

Numpy (1.18.2)

Matplotlib (3.0.3)

Tensorflow (2.1.0)

Sklearn (0.0)

Seaborn (0.8.1)

Scipy (1.4.1)

Mat4py (0.4.2)

O algoritmo está comentado.

Sobre a Execução do Algoritmo:

Ao se executar o algoritmo, uma pequena tela, como a mostrada abaixo, irá surgir:

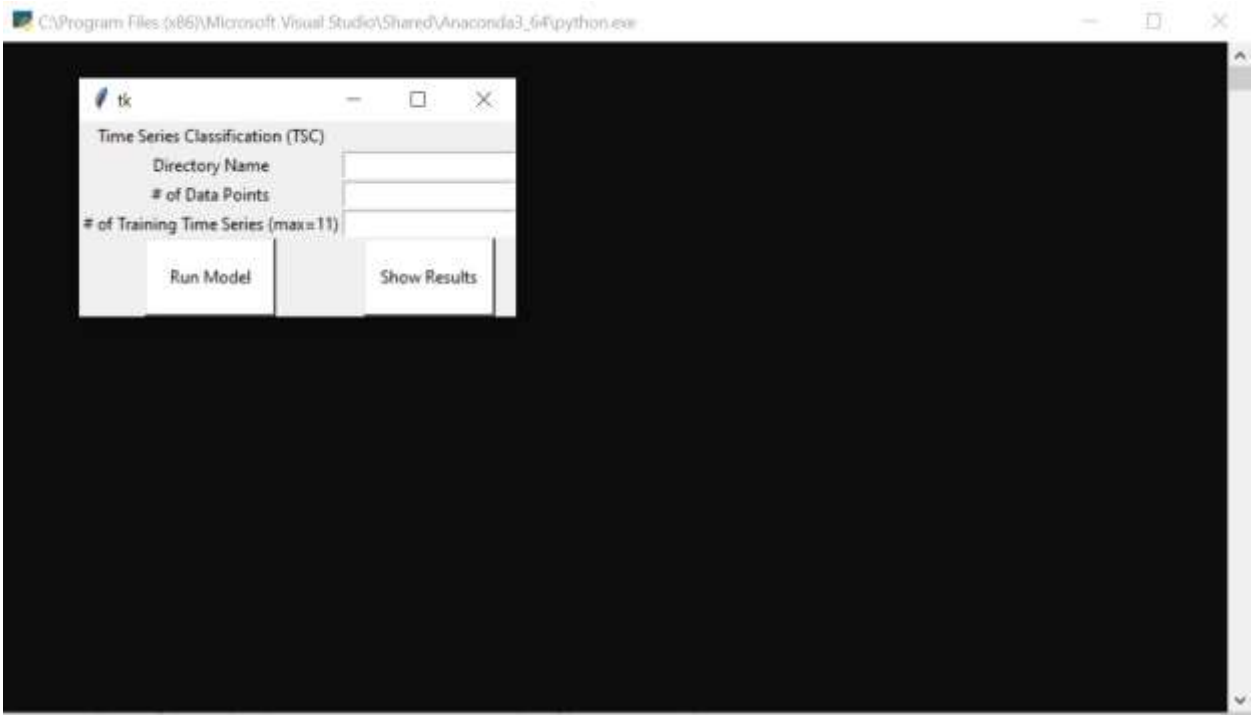

Figure 2: Tela do software.

Para se executar um modelo completo, com treinamento, deve-se preencher os três campos e clicar no botão *Run Model*. O primeiro campo define o nome do diretório onde será salvo as informações do modelo, geradas pelo Tensorflow, bem como uma figura do modelo, um gráfico *Model Loss,* outro gráfico *Model Accuracy*, um gráfico com *random trials*, um para cada subcondição do experimento, e uma *Confusion Matrix*, mostrando o desempenho do modelo. O segundo campo define o número de pontos a serem usados em consideração na série temporal do dado bruto. O dado bruto vem com 2 milhões de pontos, para reduzir o consumo de memória, é aplicada uma função de *downsample* para o número de pontos escolhidos nesse campo. O valor usado nos meus treinamentos foi de 100 mil pontos. O terceiro campo define quantas séries temporais serão usadas para treinamento. Como o total são 12. Se forem

escolhidas 6 séries temporais nesse campo, 6 séries temporais serão usadas como treinamento e 6 como teste.

Quando se clica no botão *Run Model*, o algoritmo carrega os dados (que deve estar no mesmo diretório que o algoritmo, dentro de uma pasta chamada *datafiles*), preprocessa, salva uma imagem do modelo, aplica a função *model.fit* do Tensorflow, aplica em seguida a função *model.evaluate*, que mostra a acurácia do modelo e então salva a *Confusion Matrix*.

Quando se clica no botão *Show Results*, o algoritmo apenas mostra em uma imagem todos os gráficos gerados após o treinamento de um modelo. Para executar essa função, basta preencher no nome do diretório onde o modeo foi salvo.

Sobre os Resultados:

Os dados do experimento disponibilizaram 12 séries temporais por classe. Sendo assim, repeti o treinamento do modelo em três situações: usando 6 séries temporais como treinamento, usando-se 8 e usando-se 10. Os resultados seguem-se logo abaixo:

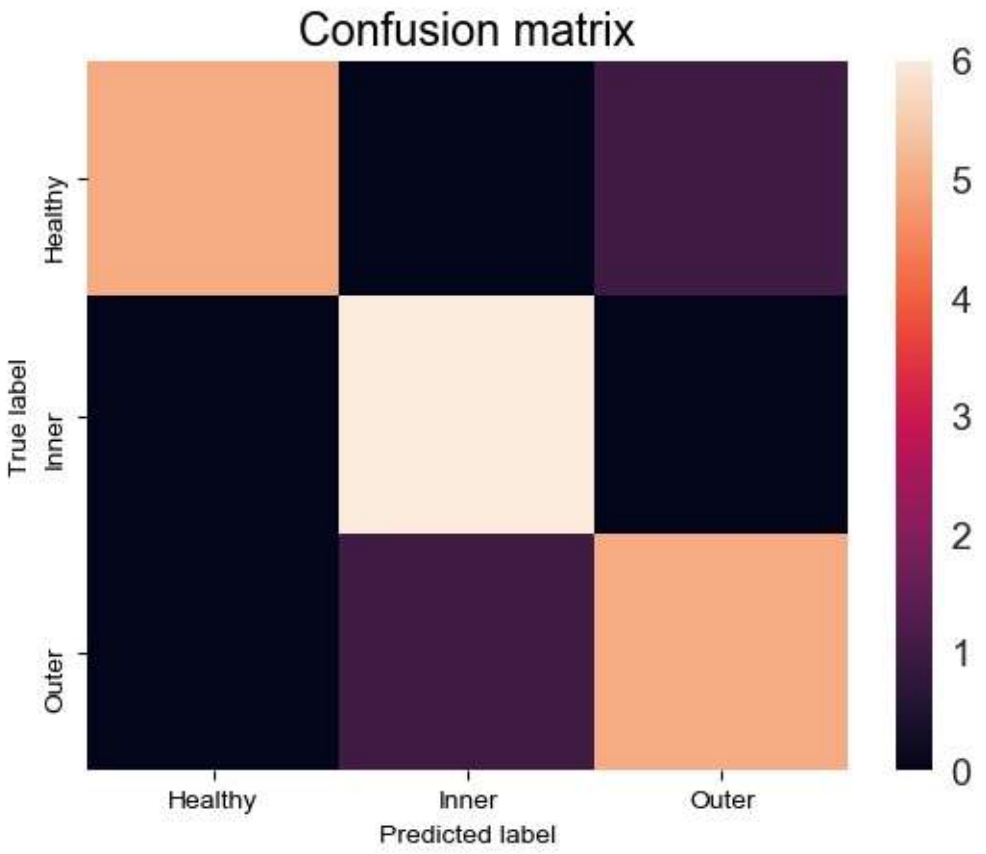

Figure 3: 88% de acurácia (6 séries de temporais para treinamento)

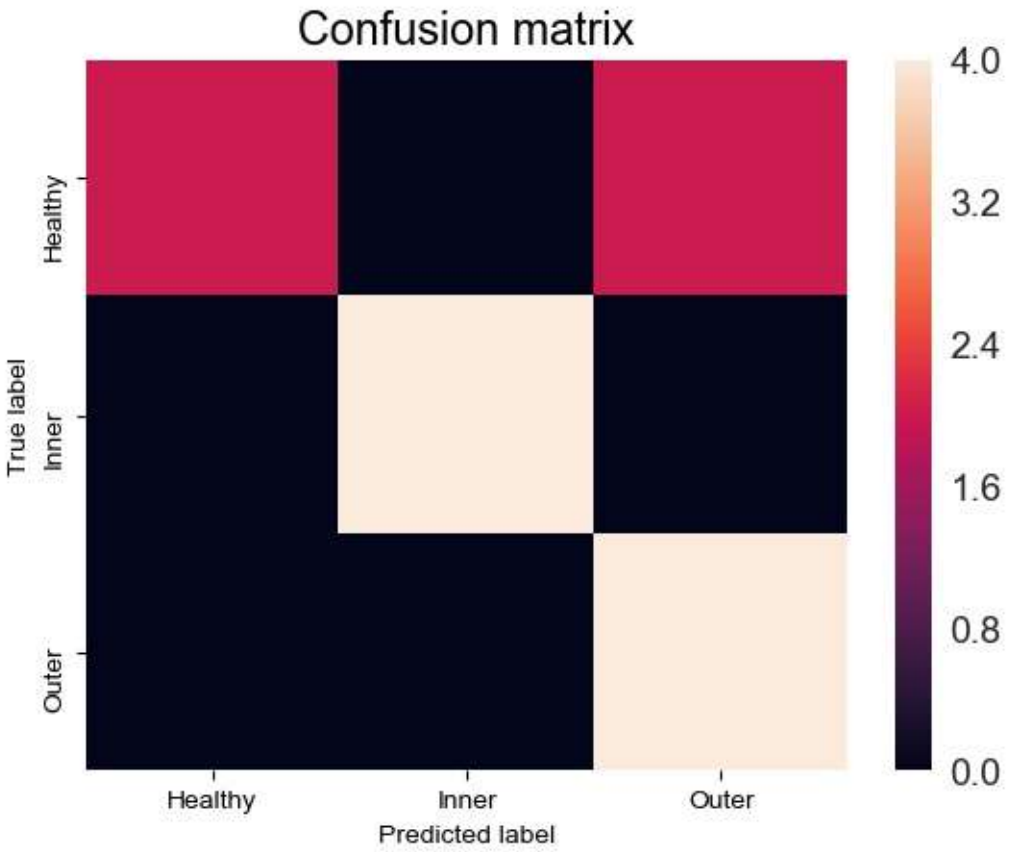

Figure 4: 83% de acurácia (8 séries temporais para treinamento)

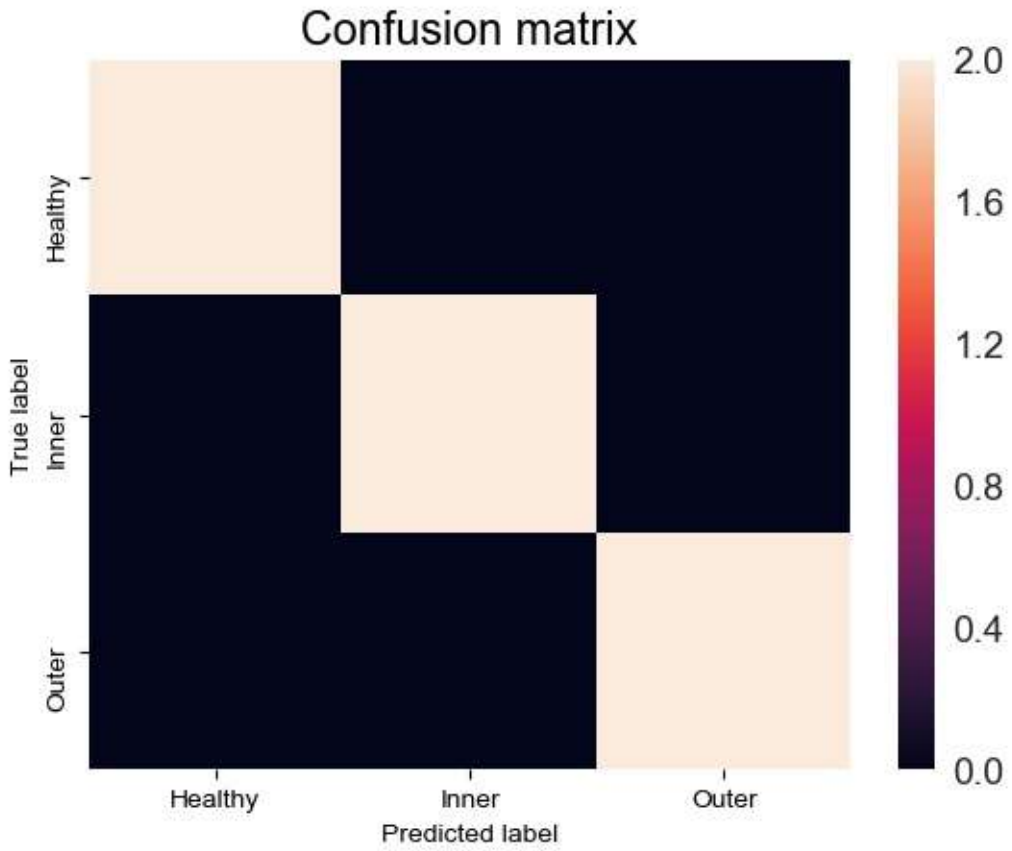

Figure 5: 100% de acurácia (10 séries temporais para treinamento)

Sobre os Modelos Testados:

Dentro do diretório do código fonte existem três diretórios chamados *Model_6ts, Model_8ts e Model_8ts*. Caso se digite o nome de um desses diretórios no primeiro campo da interface gráfica do algoritmo e clica-se em *Show Results*, uma tela semelhante a abaixo irá aparecer:

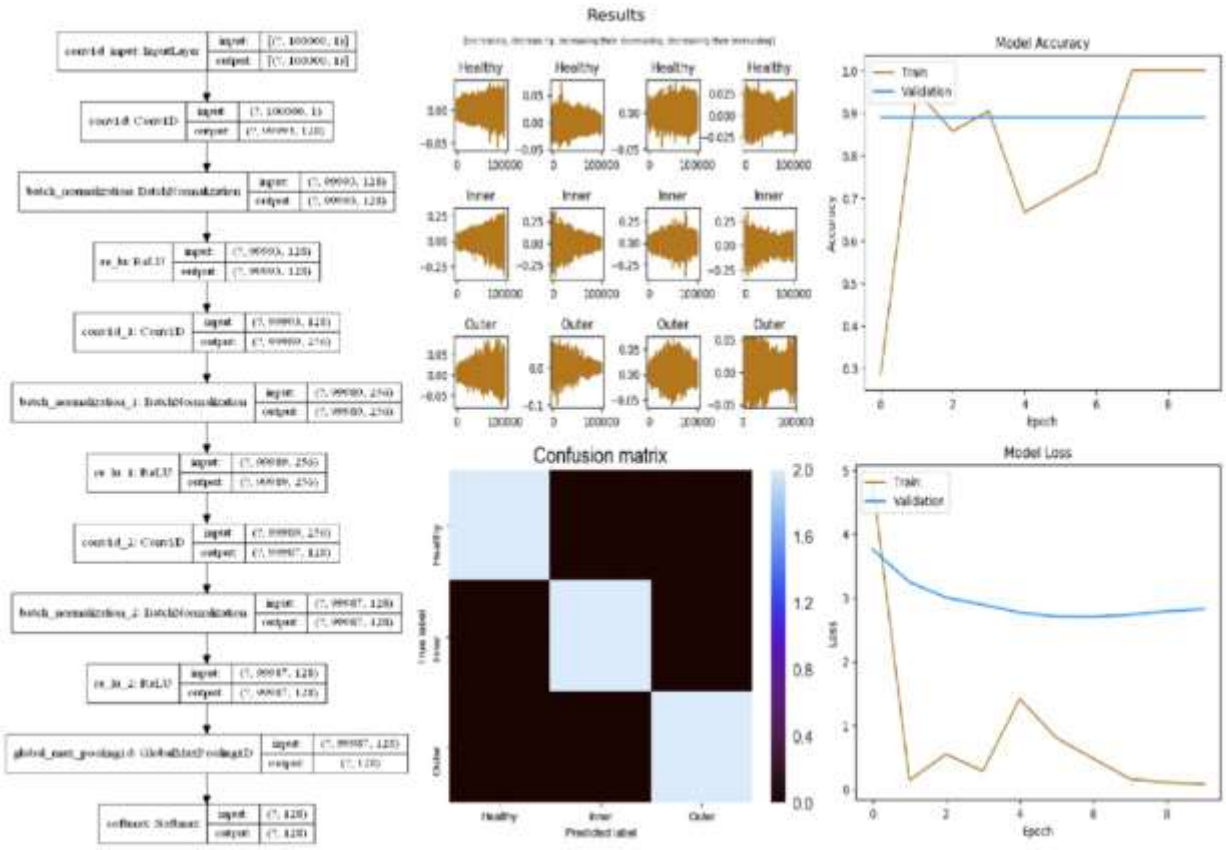

Figure 6: Exemplo de Resultado

Essa tela contém um desenho esquemático do modelo, um *plot* para um *trial* aleatório de cada subcondição do experimento, depois de aplicado o *downsample*, um gráfico para *Model Accuracy* e um gráfico para *Model Loss*.

Artigos Citados:

[1] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, Pierre-Alain Muller. Deep learning for time series classification: a review https://arxiv.org/abs/1809.04356

[2] Zhiguang Wang, Weizhong Yan, Tim Oates. Time Series Classification from Scratch with Deep Neural Networks: A Strong Baseline https://arxiv.org/abs/1611.06455