Lab CNN

Paulo Roberto Farah
Universidade Federal do Paraná (UFPR)
Programa de Pós-Graduação em Informática (PPGInf)
INFO 7004 – Aprendizagem de Máquina

I. Introdução

Convolutional Neural Network (**CNN**) é uma categoria de métodos de aprendizado supervisionado que têm sido aplicada processar e aprender representações discriminantes em imagens. É uma classe de redes neurais artificiais *feedforward* onde as conexões seguem em um único sentido.

Nesse contexto, o objetivo desse trabalho é avaliar os efeitos das alterações dos parâmetros do CNN. Os parâmetros considerados foram neurônios, camadas escondidas, épocas, Primeiro, o impacto da quantidade de neurônios na camada escondida. Segundo, quais efeitos o aumento da quantidade de camadas escondidas produz nos resultados do classificador. Em seguida, avaliar diferentes valores de épocas de treinamento e, por fim, discutir qual arquitetura (simples ou complexa) entra em *overfitting* com mais facilidade.

II. Metodologia

A metodologia para a execução dessa atividade incluiu o planejamento dos experimentos, uma descrição da implementação e as instruções de instalação e execução do programa.

A. Planejamento

A avaliação foi planejada com os seguintes valores dos parâmetros: **neurônios:[10, 50, 100, 150, 200, 225, 250, 275, 300]**, **camadas: 1 e 2** e **épocas: [10, 50, 100, 125, 150, 175, 200]**. Executou-se o programa inicialmente com 200 épocas para encontrar a melhor quantidade de neurônios. Em seguida, executou-se a rede com o valor de neurônio com melhor acurácia para testar os valores das épocas.

Para a camada de convolução, foram testados os seguintes parâmetros: funções de ativação relu e sigmoid, poolings: MaxPooling2D e AveragePooling2D, tamanhos de filtros 3x3 e 5x5 e 2 e 3 camadas de convolução. Além disso, foi atribuído um valor fixo de *seed* para a rede com o objetivo de isolar o efeito de aleatoriedade do algoritmo.

Foram coletadas as métricas de desempenho de teste *acurácia*, *loss*, *matriz de confusão* e o *histórico* de acurácia de treino e teste.

Os experimentos foram executados em um computador com processador Intel i7 8^{th} geração, com 16GB de memória e sistema operacional Windows 10. Foi utilizado python 3.6.6rc1 (64 bits) e IDE PyCharm para o desenvolvimento.

B. Implementação

O programa cnnmes.py foi modificado para executar os parâmetros definidos anteriormente, gravar os resultados nos arquivos neuronios.csv, camadas.csv e epocas.csv. Foram acrescentados os métodos *criar_modelo*, *gerar_resultados*, *plot* e *salvar_arquivo*.

C. Instalação e Execução

A instalação dos pacotes necessários a execução do classificador pode ser feita com os seguintes comandos, descritos a seguir. As bases de treino e teste devem ser gravadas na pasta cnnmes com os nomes train.txt e test.txt.

```
unzip cnnmes.zip
cd cnnmes
virtualenv venv
source venv/bin/activate
pip3 install -r requirements.txt
```

Para executar o programa:

```
python3 cnnmes.py
```

III. Resultados

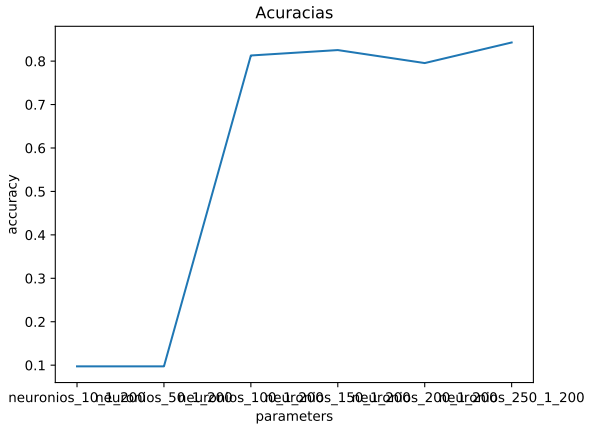
Os resultados obtidos são descritos a seguir. O critério principal para escolha dos resultados foi a acurácia.

A. Quantidade de neurônios na camada escondida

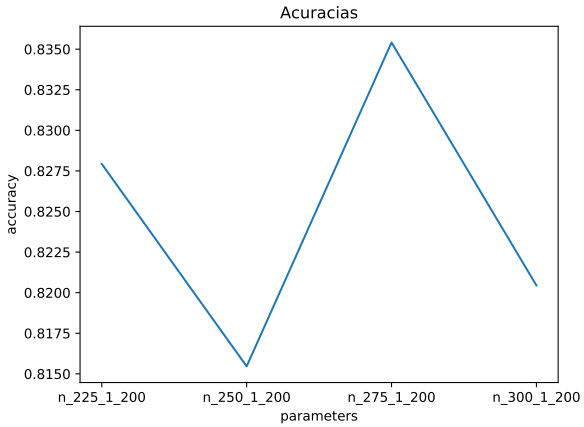
A Figura 1 mostra os valores das acurácias obtidos na primeira análise. Em seguida, foi realizada uma análise com valores próximos ao do melhor resultado na primeira avaliação, como um ajuste fino da análise. Essas análises foram feitas com 1 camada escondida e 200 épocas. Identificou-se que a maior acurácia foi obtida com 275 neurônios e 175 épocas.

Nesta seção estão descritos os resultados da quantidade de neurônios na camada escondida. A Figura 2 mostra os valores de acurácia e perda (*loss*) para 275 neurônios com 1 camada escondida e 175 épocas. Apesar da acurácia, pode-se observar que a perda para esse conjunto de parâmetros é alta. Esse resultado indica que provavelmente não seja a melhor escolha de parâmetro. Pela demora de treinamento não foi possível testar novamente a rede com diferentes parâmetros.

A Figura 3 apresenta os valores da acurácia de treino e de teste, indicando que as escolhas não foram as melhores e que a rede acabou entrando em overfitting. Além disso, mostra a matriz de confusão dessa configuração.



(a) Primeira avaliação da acurácia.



(b) Segunda avaliação da acurácia.

Figura 1: Escolha da melhor acurácia.

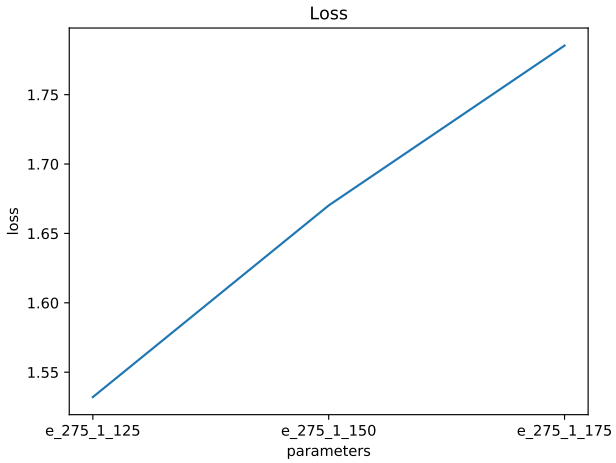


Figura 2: Perda para 275 neurônios, 1 camada escondida e 175 épocas.

Também foram testadas duas camadas densas para a rede neural, mostrada na Figura 4.

Os resultados anteriores utilizaram a função de ativação *relu*. Na Figura 5 é mostrado o resultado da análise para a função de ativação *sigmoid*. Observa-se pelos resultados que, à medida que as épocas aumentam, a acurácia do treino aumenta e do teste reduz. Esse comportamento indica que, para esse conjunto de dados, o modelo apresenta overfitting para os parâmetros utilizados nesse experimento.

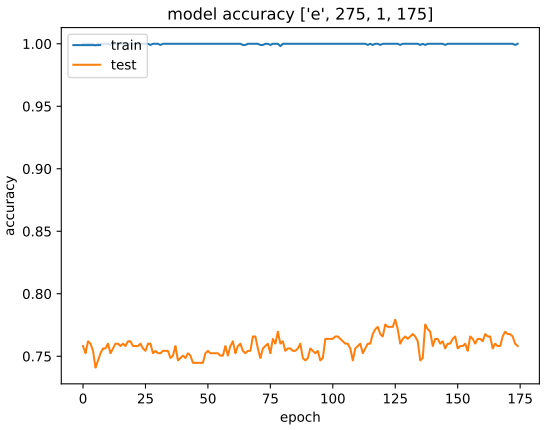
Os resultados anteriores utilizaram a função de pooling *MaxPooling2D*. Na Figura 6 é mostrado o resultado da alteração para a função de pooling *AveragePooling2D*. Observa-se pelos resultados que, à medida que as épocas aumentam, a acurácia do treino aumenta e do teste reduz. Esse comportamento indica que, para esse conjunto de dados, o modelo apresenta overfitting para os parâmetros utilizados nesse experimento.

Os resultados anteriores utilizaram tamanho de filtro *3x3*. Na Figura 7 é mostrado o resultado da alteração para tamanho de filtro de *5x5*. Observa-se pelos resultados que, à medida que as épocas aumentam, a acurácia do treino aumenta e do teste reduz. Esse comportamento indica que, para esse conjunto de dados, o modelo apresenta overfitting para os parâmetros utilizados nesse experimento.

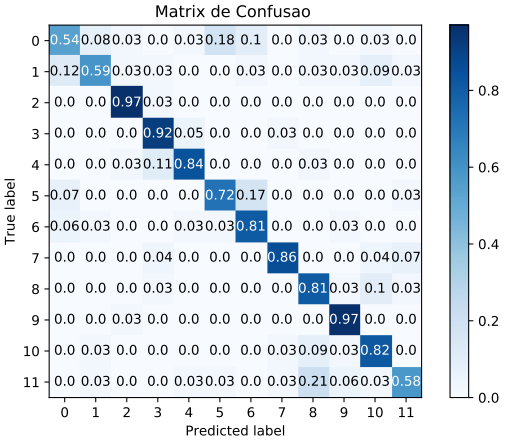
Os resultados anteriores utilizaram 3 camadas de convolução com tamanho de filtro 3x3, MaxPooling2D e função de ativação relu. Na Figura 8 é mostrado o resultado da alteração para tamanho de filtro de *5x5*. Observa-se pelos resultados que, à medida que as épocas aumentam, a acurácia do treino aumenta e do teste reduz. Esse comportamento indica que, para esse conjunto de dados, o modelo apresenta overfitting para os parâmetros utilizados nesse experimento.

IV. Conclusão

O presente trabalho teve como objetivo principal a avaliação do desempenho dos parâmetros da rede neural CNN. Foi avaliado o desempenho desses parâmetros com uma base de meses. Infelizmente, os resultados não ficaram conforme desejado, pois a rede acabou entrando em overfitting e não houve tempo hábil para realizar mais testes com outros valores. Foi adotada uma estratégia errada para definir valores muito grandes de épocas, o que ocasionou um tempo muito maior para treinamento da rede CNN e impossibilitou o teste de outros valores de parâmetros que pudessem apresentar um resultado melhor. Foi adotada a estratégia de definir valores altos e distribuídos de forma que pudessem pegar um grande intervalo para análise. Contudo, em um próximo experimento, pode ser mais interessante usar valores menores devido ao tempo que os valores altos levam para treinamento e execução.

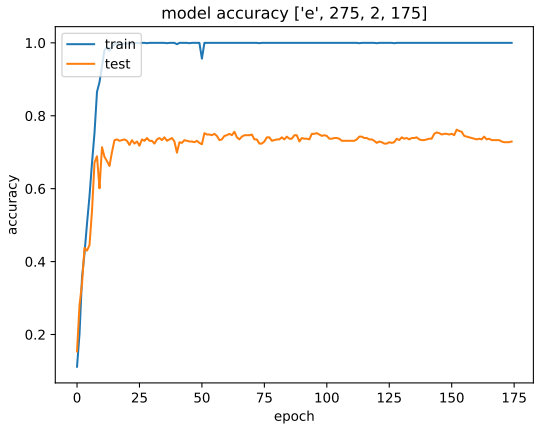


(a) Acurácias de treino e teste.

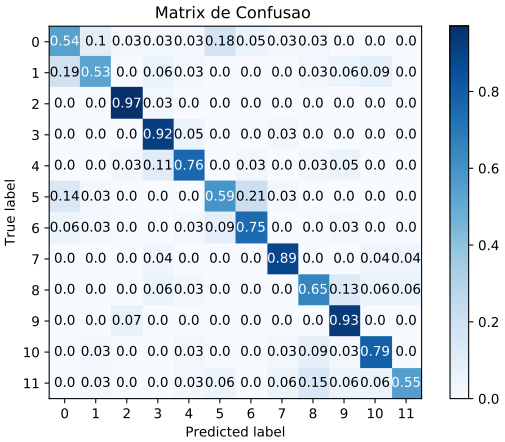


(b) Matriz de confusão.

Figura 3: Parâmetros: 1 camada escondida com 275 neurônios e 175 épocas em overfitting.

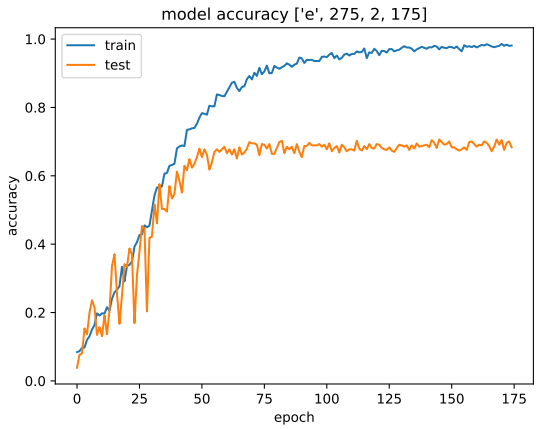


(a) Acurácia.

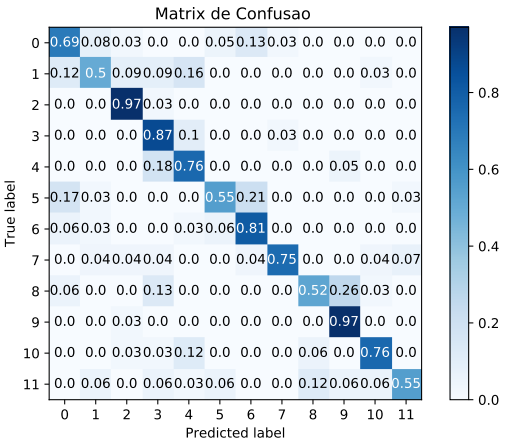


(b) Matriz de confusão.

Figura 4: Parâmetros: 2 camadas escondidas e 275 neurônios e 175 épocas.

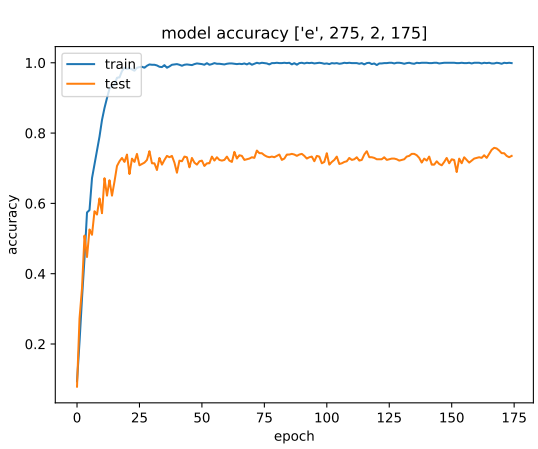


(a) acurácia.

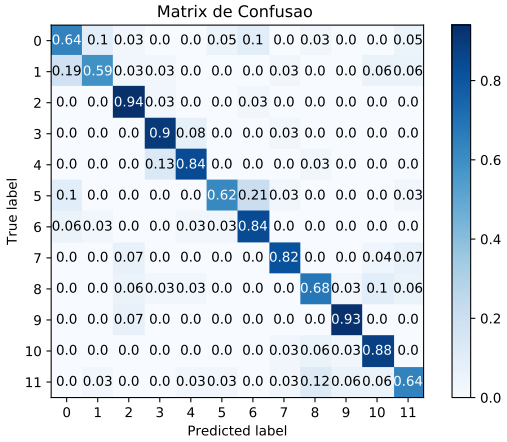


(b) Matriz de confusão.

Figura 5: Acurácia e matriz de confusão com alteração da função de ativação de relu para sigmoid.

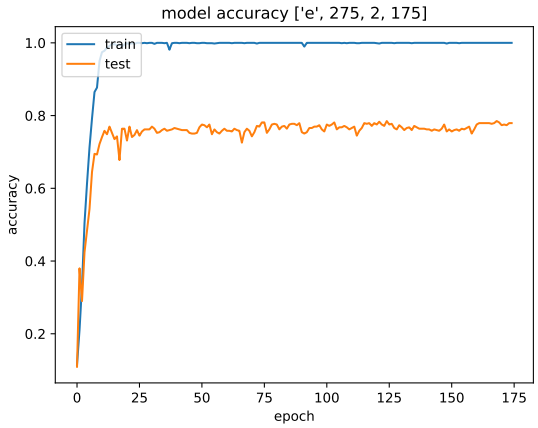


(a) acurácia.

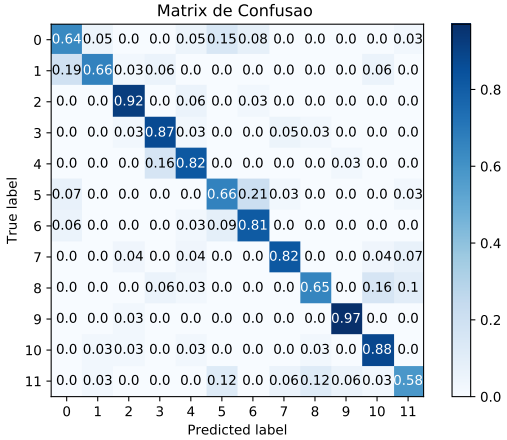


(b) Matriz de confusão.

Figura 6: Acurácia e matriz de confusão com alteração da função de pooling de MaxPooling2D para AveragePooling2D.

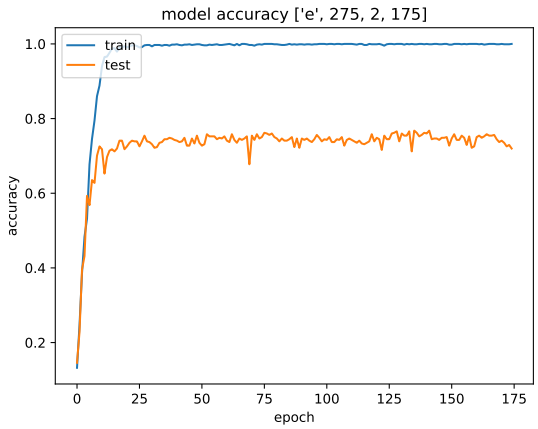


(a) acurácia.

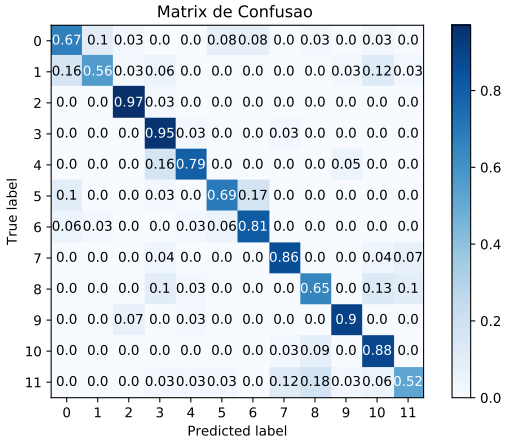


(b) Matriz de confusão.

Figura 7: Acurácia e matriz de confusão com tamanho de filtro de 5x5.



(a) acurácia.



(b) Matriz de confusão.

Figura 8: Acurácia e matriz de confusão com 3 camadas de convolução.