

Técnica supervisionada: Multi Layer Perceptron (MLP)

Luiz Felipe de Souza Silva¹

¹Departamento de Engenharia da Computação (DCA) – Universidade Federal do Rio Grande do Norte (UFRN)


lf06092004@gmail.com

Resumo. Este relatório apresenta um pipeline desenvolvido com a técnica supervisionada MLP para um classificador de raças de gatos e cachorros. O relatório apresenta testes e escolhas das melhores condições do classificador como: função de ativação, solver, número de interação, quantidade de neurônios, quantidade de camadas escondidas para o melhor desempenho do classificador.

1. Informações Gerais.

O objetivo da atividade foi realizar a classificação de raças de gatos e cachorros a partir de imagens, utilizando o classificador supervisionado Multi Layer Perceptron (MLP). A ideia é realizar uma série de passos e respostas de algumas perguntas para montar uma boa rede neural artificial para a classificação das raças desse trabalho.

1. Utilize os valores default do MLP e varie as funções de ativação ('identity', 'logistic', 'tanh', 'relu'). Escolha a que deu o melhor desempenho;
2. Varie o valor do solver entre: 'sgd', 'adam'. Veja qual deles deu o melhor resultado e escolha este solver para os experimentos;
3. Varie a quantidade de neurônios da camada escondida (hidden_layer_sizes) de acordo com a metodologia mostrada na aula prática de MLP, que envolve o número de atributos e a quantidade de classes em seu dataset.
4. Escolha o número de neurônios com o melhor desempenho no item anterior. Varie a taxa de aprendizado (learning_rate_init) entre os seguintes valores: .001, 0.01, 0.1, ou qualquer outro conjunto de possíveis valores.
5. Escolha a taxa de aprendizado com o melhor desempenho no item anterior. Por último, varie o número de iterações (500, 1000, 1500, etc).

Os dados de acurácia do modelo treinado com *MLP* podem ser visto através desse direcionamento para tabela:  Tabela_Checkpoint3

Notebooks, extração de características e documentação podem ser visto no repositório do github: https://github.com/luiz-pytech/Classificador_Racas_gato_Cachorro

2. Função de ativação

O Scikit Learning tem em sua documentação 4 funções de ativação para os neurônios das camadas escondidas, são elas: relu, identity, logistic, tanh. O valor default para as interações da máxima da MLP é de 200 interações, na maioria dos casos esse valor de interação não foi o suficiente para a MLP convergir para uma solução ideal, ainda assim,

mesmo com esse número default a função “logistic” demonstrou-se como a melhor função de ativação para esse projeto. Para contornar esse problema de não convergência, foi padronizado um valor de 1800 interações para todas as funções, e novamente à função “logistic” se apresentou como a melhor função de ativação para o objetivo desse projeto, com 97,12% de acurácia média e segundo menor desvio padrão dentre as análises. Dessa forma, à função escolhida para a continuação desse checkpoint e construção do MLP será a função “logistic” também conhecida como a função sigmoide. Abaixo os dados dos testes realizados neste primeiro passo.

Figura 1: Acurácia das Bases Analisadas com diferentes funções de ativação

BASES	RELU	TANH	IDENTITY	LOGISTIC
CNN_VGG16_256_max ⁽⁵¹²⁾	97,49%	98,32%	96,65%	99,58%
CNN_VGG16_256_avg ⁽⁵¹²⁾	97,49%	98,74%	97,07%	99,16%
CNN_VGG19_128_max ⁽⁵¹²⁾	91,63%	91,63%	92,46%	94,56%
CNN_VGG19_256_max ⁽⁵¹²⁾	95,81%	97,90%	96,65%	98,74%
CNN_VGG19_128_avg ⁽⁵¹²⁾	92,46%	94,97%	89,54%	93,30%
CNN_VGG19_256_avg ⁽⁵¹²⁾	96,65%	98,74%	97,07%	99,58%
PCA_CNN_VGG16_256_max ⁽¹⁰⁾	97,49%	98,74%	99,16%	99,58%
PCA_CNN_VGG16_256_avg ⁽¹⁰⁾	99,16%	98,74%	99,16%	99,16%
PCA_CNN_VGG19_128_max ⁽¹⁰⁾	89,12%	92,46%	92,05%	92,46%
PCA_CNN_VGG19_128_avg ⁽¹⁰⁾	92,05%	94,56%	92,88%	95,81%
PCA_CNN_VGG19_256_max ⁽¹⁰⁾	97,07%	97,90%	97,90%	98,74%
PCA_CNN_VGG19_256_avg ⁽¹⁰⁾	94,49%	98,74%	97,90%	98,32%
MÉDIA ==>	94,81%	96,86%	95,71%	97,12%
DESVIO PADRÃO ==>	0,02985459	0,02196998	0,03244143	0,02546311

Fonte: Autor, 2025

3. Solver = Adam ou SGD?

O solver é um parâmetro do Scikit-Learn para as MLPs que denomina como a rede irá se comportar para minimizar o erro. No scikit as possibilidades de escolha são: adam, sgd e lbfgs. No caso de escolha e continuação desse trabalho o “adam” se apresentou como o melhor parâmetro para a construção da rede, sendo superior em quase todas as bases analisadas.

4. Hidden Layer Sizes

Na tentativa de responder os questionamentos da pergunta número 3 foi variado o número de neurônios na camada escondida. Foi escolhido de ponto de partida 258 neurônios através de uma tese prática da aula de MLP, onde o número de neurônios é dado por:

$$N \text{ de neuronios} = \frac{2 \times N \text{ de atributos} + N \text{ de classes}}{2}$$

A partir desse ponto de partida foram escolhidos dois valores acima e dois valores abaixo a fim de verificar regiões de overfitting e underfitting. Os valores escolhidos foram: 1, 25, 258, 512 e 1000.

Com o aumento de neurônios foi possível notar em algumas bases à diminuição da acurácia, mas ainda sim, não o bastante para acusar um overfitting, mas é uma baixa na acurácia com os mesmo parâmetros de teste. Segue o exemplo da base CNN_VGG19_128_max e CNN_VGG19_128_avg, respectivamente:

Figura 2: Análise de Overfitting e Underfitting

Nº neurônios: 1		Acurácia Treino: 0.5144		Acurácia Teste: 0.4017
Nº neurônios: 25		Acurácia Treino: 0.9946		Acurácia Teste: 0.9498
Nº neurônios: 258		Acurácia Treino: 1.0000		Acurácia Teste: 0.9665
Nº neurônios: 512		Acurácia Treino: 1.0000		Acurácia Teste: 0.9623
Nº neurônios: 1000		Acurácia Treino: 1.0000		Acurácia Teste: 0.9540

Nº neurônios: 1		Acurácia Treino: 0.5144		Acurácia Teste: 0.4100
Nº neurônios: 25		Acurácia Treino: 1.0000		Acurácia Teste: 0.9498
Nº neurônios: 258		Acurácia Treino: 1.0000		Acurácia Teste: 0.9498
Nº neurônios: 512		Acurácia Treino: 1.0000		Acurácia Teste: 0.9498
Nº neurônios: 1000		Acurácia Treino: 1.0000		Acurácia Teste: 0.9456

Com a quantidade de neurônios mais baixa é possível notar faixa de underfitting, onde o modelo não aprende nada e ainda está mal ajustado, neste caso de 1 a 25 neurônios foi onde não aprendeu o suficiente para uma acurácia otimista na fase de teste. Dessa forma, para os números escolhidos à faixa de underfitting está entre 1-25 neurônios e a faixa de overfitting está acima de 512 neurônios.

5. Rate Learning

De acordo com os resultados obtidos, foi necessário escolher a melhor quantidade de neurônios e, em seguida, variar a taxa de aprendizagem da MLP. A quantidade de neurônios que apresentou o melhor desempenho foi justamente o ponto inicial de partida, alcançando uma média de 97,93% de acurácia em todas as bases, tanto nos testes com K-Fold quanto com Holdout.

As taxas de aprendizagem foram testadas nos seguintes valores: 0.001, 0.01 e 0.1, com o objetivo de entender como esse hiperparâmetro influencia o processo de ajuste da rede neural. A taxa de aprendizagem tem relação direta com o tamanho dos passos que o modelo dá durante a atualização dos pesos a cada iteração. Dessa forma, taxas muito altas podem fazer o modelo não convergir para uma solução ideal, ou até mesmo divergir. Por outro lado, taxas muito baixas podem resultar em um aprendizado lento, impedindo que o modelo alcance uma boa capacidade de generalização.

Ao analisar os resultados, foi possível observar os seguinte comportamentos:

- A taxa de 0.1 apresentou, em alguns casos, problemas de não convergência, com acurácias bem abaixo do esperado. Isso caracterizou situações de underfitting,

onde tanto a acurácia de treino quanto a de teste foram baixas.

- Já as taxas de 0.001 e 0.01 mostraram-se mais estáveis, resultando em bons níveis de acurácia e sendo recomendadas como faixas iniciais adequadas para testes em MLPs. Mas ainda assim a taxa 0.001 se mostrou como mais estável para o aprendizado da MLP.

Por fim, com base nos experimentos, foi possível concluir que a faixa de underfitting tende a ocorrer em valores mais altos da taxa de aprendizagem (acima de 0.1). Já a ocorrência de overfitting esteve mais associada ao número de iterações e à complexidade da rede (como o número de neurônios) do que propriamente à taxa de aprendizagem.

5. Número de interações

A fim de verificar como o número de interações influencia na construção da rede, foram testados diferentes configurações desse hiperparâmetro. Os valores testados foram: 200, 500, 1200, 2000, 5000, 8000 e 10000. Nesse tipo de parâmetro, o comportamento esperado é o seguinte: para valores baixos a rede deve apresentar underfitting, ou seja, à quantidade de interação não foi o suficiente para o aprendizado da MLP e para valores altos de interação a MLP se ajusta demais aos meus dados apresentando overfitting. O número 200 é a quantidade de interação padrão da documentação do Scikit-Learn. Para essa quantidade em alguns casos a rede não tinha convergido diretamente, o que sugere que um intervalo de 1-200 de underfitting. Para interações de 500, 1200, 2000 a rede mostrou-se tá bem ajustada e generalizando muito bem. No entanto para valores acima de 5000 a acurácia de treino continua bem alta, mas a acurácia de teste começa a ter decaída demonstrando problema de overfitting.

Figura 3: Análise de Overfitting e Underfitting para número de interações

```
----- PCA_cnn_VGG19_256_avg -----
C:\Users\lf060\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\normalization\_multilayer_perceptron.py:691: ConvergenceWarning: Stochastic Optimizer: Maximum iterations (100) reached and the optimization hasn't converged yet.
  warnings.warn(
Nº neurônios: 258 | Acurácia Treino: 1.0000 | Acurácia Teste: 0.9833
Nº neurônios: 258 | Acurácia Treino: 1.0000 | Acurácia Teste: 0.9833
Nº neurônios: 258 | Acurácia Treino: 1.0000 | Acurácia Teste: 0.9833
```

6. GridSearchCV

Por questão de comparação, foi utilizado a função Grid Search para verificar o quão distante os valores encontrados manualmente ficaram distantes dos oferecidos pela função utilizada. A busca foi realizada em um espaço de parâmetros contendo diferentes quantidades de neurônios, funções de ativação, taxas de aprendizagem com um número fixo de interações, 500 interações. É possível notar os resultados na tabela 1.

A função “*logistic*” ou função sigmoideal se apresentou como melhor função de ativação na maioria das bases, o que conferiu com uma análise manual da MLP. Em outras poucas bases a melhor função de ativação se apresentou como a Relu e Tangente

Hiperbólica. No entanto, é possível concluir que a função sigmoide se apresenta como melhor função de ativação para as bases utilizadas nesse projeto.

Tabela 1: Resultados da função Grid Search

Base	Função de Ativação	Nº de Neurônios	Learning Rate	Acurácia Média CV	Acurácia Holdout
cnn_VGG16_256_max	logistic	258	0.001	0.9928	0.9916
cnn_VGG16_256_avg	logistic	100	0.001	0.9946	0.9916
cnn_VGG19_128_max	tanh	400	0.001	0.9712	0.9163
cnn_VGG19_256_max	logistic	500	0.001	0.9928	0.9916
cnn_VGG19_128_avg	logistic	100	0.01	0.9748	0.9540
cnn_VGG19_256_avg	logistic	100	0.001	0.9946	0.9958
PCA_cnn_VGG16_256_max	relu	500	0.001	0.9964	0.9916
PCA_cnn_VGG16_256_avg	tanh	400	0.001	0.9946	0.9916
PCA_cnn_VGG19_128_max	logistic	200	0.001	0.9712	0.9247
PCA_cnn_VGG19_256_max	logistic	100	0.01	0.9928	0.9874
PCA_cnn_VGG19_128_avg	logistic	100	0.01	0.9676	0.9623
PCA_cnn_VGG19_256_avg	logistic	258	0.01	0.9964	0.9833

Na taxa de aprendizagem demonstrou as expectativas analisadas manualmente, onde 0.1 era uma taxa muito alta e a rede não convergia para uma solução adequada. E o 0.001 e a 0.01 demonstraram resultados mais estáveis e seria um bom pontapé inicial para construção da MLP. Com o Grid Search demonstrou-se claramente essas duas taxas como as melhores para as MLP treinadas nas bases desse projeto.

A diferença de acurácia foi pequena, entre 1 a 2 pontos percentuais, mas a função Grid Search conseguiu encontrar melhores configurações para redes que inicialmente tiveram

um resultado abaixo na busca manual, como as bases padronizadas com imagens de 128x128, demonstrando a importância de funções automatizadas para construção de redes neurais.

A quantidade de neurônios apresentou muita variação com relação ao que foi considerado melhor na análise manual, 258 neurônios. No entanto, com análise da função outras quantidades demonstraram melhores resultados: 100, 400, 500, demonstrando uma diferença de 158, 142 e 252 neurônios, respectivamente. Mas ainda sim, em algumas bases a quantidade de partida demonstrou como a melhor quantidade de neurônios, mostrando que é um bom ponto de partida.