Relatório do Trabalho Prático Conhecimento e Raciocínio

Redes Neuronais

Docente
Viriato António Pereira Marinho Marques

Alunos
Paulo Henrique Figueira Pestana de Gouveia nº 2020121705 – LEI
Pedro Nogueira nº 2020136533 – LEI

Coimbra, 13 de Setembro de 2022

Índice

1 - Introdução

Este trabalho foi realizado no âmbito da Unidade Curricular de Conhecimento e Raciocínio, tem por objetivo o desenvolvimento de um projeto capaz de classificar corretamente 6 formas geométricas(círculo, papagaio, paralelepípedo, quadrado , trapézio , triângulo) utilizando diferentes arquiteturas de redes neuronais do tipo feedforward.

2 - Decisões Tomadas Para Implementação

2.1 - Pré processamento das imagens

Para uma otimização da aplicação foi realizado um redimensionamento das imagens para uma resolução de 28x28 com recurso a funções da toolbox image processing do Matlab.

Ao longo do processo de otimização foram feitos diversos passos, dos quais a transformação das imagens para binário, onde a cor preta é identificada pelo valor 1 e a cor branca identificada pelo valor 0, de seguida foi realizado o preenchimento da matriz à medida que iam sendo lidas as imagens.

2.2 - Targets

Para a utilização dos targets foi realizada uma matriz com as 6 formas geométricas, onde cada valor identifica a forma geométrica desejada.

3 - Testes

3.1 - Alínea A

Alínea A: Usando as funções de manipulação de imagem do Matlab converta as imagens fornecidas em matrizes binárias. Criar uma rede neuronal para treinar o reconhecimento dos caracteres da pasta "start".

Foram treinadas várias redes com diferentes topologias, funções de treino e ativação, podendo deste modo fazer uma comparação qual seria a melhor combinação de treinamento usando um parâmetro de "precisão" como avaliação da performance da rede.

Estes resultados podem ser consultados no ficheiro "a_testes" incluindo na pasta de submissão, em média foram efetuados 5 testes para cada cenário à exceção dos modos de treino que demoravam muito tempo.

Na seguinte tabela são os parâmetros de base para servir de comparação dos vários testes efetuados:

Rede Base				
Topologias	Função de Treino	Funções de Ativação	Epochs	Melhor Precisão
10	traingdx	tansig/purelin	1000	100%

Primeiramente alteramos apenas a função de ativação da camada escondida da rede base:

Diferentes funções de ativação				
Topologias	Função de Treino	Funções de Ativação	Epochs	Melhor Precisão
10	traingdx	compet/purelin	1000	50%
10	traingdx	satlins/purelin	1000	100%
10	traingdx	radbasn/purelin	1000	16,6%
10	traingdx	netinv/purelin	1000	86,6%

Confirmamos que a escolha da função de ativação para a camada escondida afeta drasticamente a precisão da classificação, os piores resultados provêm de funções que são tipicamente usadas em camada de saída, justificando assim a sua precisão baixa.

De seguida alteramos a função de ativação da camada de saída da rede base:

Diferentes funções de ativação				
Topologias	Função de Treino	Funções de Ativação	Epochs	Melhor Precisão
10	traingdx	tansig/netinv	1000	20%
10	traingdx	tansig/radbasn	1000	100%
10	traingdx	tansig/satlins	1000	100%
10	traingdx	tansig/compet	1000	16,6%

A conclusão da tabela anterior vem a provar os resultados desta tabela, as funções antes vistas a terem uma fraca precisão para as camadas escondidas, tornam-se muito mais eficientes nas camadas de saída (à exceção da satlins que obteve bons resultados em ambos). Isto vem a provar que existem realmente funções mais apropriadas para camadas escondidas e camadas de saída.

Efetuamos vários testes alterando apenas a função de treino da rede base:

Diferentes funções de treino				
Topologias	Função de Treino	Funções de Ativação	Epochs	Melhor Precisão
10	trainlm	tansig/purelin	1000	100%
10	trainbr	tansig/purelin	1000	100%
10	trainbfg	tansig/purelin	1000	100%
10	traincgf	tansig/purelin	1000	100%

Por final alteramos a quantidade de camadas e neurônios por camada da rede base:

Diferentes topologias				
Topologias	Função de Treino	Funções de Ativação	Epochs	Melhor Precisão
25	traingdx	tansig/purelin	1000	100%
50	traingdx	tansig/purelin	1000	100%
10 10	traingdx	tansig/purelin	1000	100%
25 25	traingdx	tansig/purelin	1000	100%
50 50	traingdx	tansig/purelin	1000	100%
10 10 10	traingdx	tansig/tansig/purelin	1000	100%
25 25 25	traingdx	tansig/tansig/purelin	1000	100%
50 50 50	traingdx	tansig/tansig/purelin	1000	100%

Os resultados de ambas as tabelas não se pode concluir muito devido a todos os testes terem obtido precisão máxima. Sendo poucas imagens os valores são bastante exagerados.

A maior diferença foi a demora da conclusão dos treinos, sendo que algumas funções de treino são bem mais intensas e mais precisas, e quanto mais camadas e neurônios mais tempo precisará (nos próximos testes a diferença do uso de diferentes funções de treino e topologias será bem mais visível)

3.2 - Alínea B

Alínea B: Teste várias parametrizações e topologias de forma a obter um bom desempenho para a classificação das imagens fornecidas na pasta "train".

Para tentar obter os melhores resultados na classificação das imagens fornecidas, foram testadas várias funções e parâmetros diferentes. Podendo assim comparar com a configuração inicial.

	Número de camadas escondidas	Número de neurónios	Funções de ativação	Função de treino	Divisão dos exemplos		Precisão Global	Precisão Teste
Configuração por defeito	1	10	tansig, purelin	trainlm	dividerand = {0.7, 0.15, 0.15}		92,6	66,6

O número e dimensão das camadas encondidas influencia o desempenho?								
Conf1	2	10,10	tansig, tansig, purelin	trainlm	dividerand = {0.7, 0.15, 0.15}		91,3	68,8
Conf2	2	25,25	tansig, tansig, purelin	trainlm	dividerand = {0.7, 0.15, 0.15}		90,6	71,1
Conf3	4	5,5,5,5	tansig, tansig, tansig, tansig, tansig, purelin	trainlm	dividerand = {0.7, 0.15, 0.15}		87,6	64,4
Conf4	4	10,10,10,10	tansig, tansig, tansig, tansig, tansig, purelin	trainlm	dividerand = {0.7, 0.15, 0.15}		89,6	68,8
Conf4	4	25,25,25,25	tansig, tansig, tansig, tansig, tansig, purelin	trainlm	dividerand = {0.7, 0.15, 0.15}		90,3	71,1

Com a realização dos primeiros testes, foi possível concluir que a alteração da quantidade de camadas e neurônios pouco afeta a precisão global e a de teste. Um parâmetro útil poderia ser o tempo decorrido para conclusão do teste, quanto mais camadas e neurônios, o tempo de conclusão aumentava consideravelmente .

A função de treino influencia o desempenho?							
Conf1	1	10	tansig, purelin	traingd	dividerand = {0.7, 0.15, 0.15}	63,3	51,1
Conf2	1	10	tansig, purelin	trainbfg	dividerand = {0.7, 0.15, 0.15}	51,3	28,8
Conf3	1	10	tansig, purelin	traingdx	dividerand = {0.7, 0.15, 0.15}	83,6	62,2
Conf4	1	10	tansig, purelin	trainbr	dividerand = {0.7, 0.15, 0.15}	96	73,3
Conf5	1	10	tansig, purelin	traincgf	dividerand = {0.7, 0.15, 0.15}	78,3	60

Após a testagem da rede com diversas funções de treino, podemos concluir que a função de treino "trainbr" utilizada na rede conf4 teve uma precisão de 96%, em contrapartida foram necessárias 7 horas para treinar a rede.

A divisão de exemplos pelos conjuntos influencia o desempenho?							
Conf1	1	10	tansig, purelin	trainlm	dividerand = {0.33, 0.33, 0.33}	75,3	58
Conf2	1	10	tansig, purelin	trainlm	dividerand = {0.9, 0.05, 0.05}	98	86,6
Conf3	1	10	tansig, purelin	trainlm	dividerand = {0.5, 0.25, 0.25}	86,6	69,3
Conf4	1	10	tansig, purelin	trainlm	dividerand = {0.60, 0.20, 0.20}	86,3	70
Conf5	1	10	tansig, purelin	trainlm	dividerand = {0.40, 0.30, 0.30}	76	54,4
Conf6	1	10	tansig, purelin	trainlm	dividerand = {0.80, 0.10, 0.10}	94	66,6

Com a realização de vários testes foi possível concluir que a rede Conf2, com uma segmentação do dataset 80% , 0.05% , 0.05% teve a melhor precisão global(98%) e de teste(86.6%) entre todas as redes testadas.

3.3 - Alínea C

Alínea C: Teste várias parametrizações e topologias de forma a obter um bom desempenho para a classificação das imagens fornecidas na pasta “test”, utilizando a melhor rede obtida em b).

3.3.1 - Tarefa 1

Tarefa 1: Sem re-treinar a rede verifique se a classificação dada é correta.

Sem re-treinar a melhor rede neuronal da alínea b, foi possível verificar uma precisão de 75%, ao identificar as imagens da pasta “test”.

Sem re-treinar a melhor rede neuronal da alínea b, foi possível verificar uma precisão de 80%, ao identificar as imagens da pasta “start”.

Sem re-treinar a melhor rede neuronal da alínea b, foi possível verificar uma precisão de 98%,ao identificar as imagens da pasta “train”.

3.3.2 - Tarefa 2

Tarefa 2: Volte a treinar a rede só com os exemplos da pasta “test”, e teste a rede com as imagens da pasta “start” , “train” e “test”.

Foi treinada a melhor rede neuronal da alínea b com imagens da pasta “test”, onde foi possível verificar uma precisão de 100%, ao identificar as imagens da pasta “test”.

Foi treinada a melhor rede neuronal da alínea b com imagens da pasta “test”, onde foi possível verificar uma precisão de 66.6%, ao identificar as imagens da pasta “start”.

Foi treinada a melhor rede neuronal da alínea b com imagens da pasta “test”, onde foi possível verificar uma precisão de 89.6%, ao identificar as imagens da pasta “train”.

3.3.3 - Tarefa 3

Tarefa 3: Volte a treinar a rede só com os exemplos da pasta (“test” + “start” + “train”), e teste a rede com as imagens da pasta “start” , “train” e “test” em separado.

Foi treinada a melhor rede neuronal da alínea b com imagens das pastas “test”+”start”+”train”, onde foi possível verificar uma precisão de 98.3%, ao identificar as imagens da pasta “test”.

Foi treinada a melhor rede neuronal da alínea b com imagens das pastas “test”+”start”+”train”, onde foi possível verificar uma precisão de 96.6%, ao identificar as imagens da pasta “start”.

Foi treinada a melhor rede neuronal da alínea b com imagens das pastas “test”+”start”+”train”, onde foi possível verificar uma precisão de 98%, ao identificar as imagens da pasta “train”.

3.4 - Alínea D

Alínea D: Desenhe manualmente algumas formas geométricas que apresentem semelhanças com os exemplos usados no treino da rede. Transcreva os desenhos para matrizes binárias. Desenvolva um pequeno programa para ler um ficheiro correspondente a uma destas imagens e aplicá-lo à melhor rede neuronal obtida em c).

Na diretoria "Imagens\extra" encontram-se o total de seis pastas com o nome da figura a que correspondem. Cada pasta contém duas figuras desenhadas manualmente para serem avaliadas pela melhor rede obtida em c).

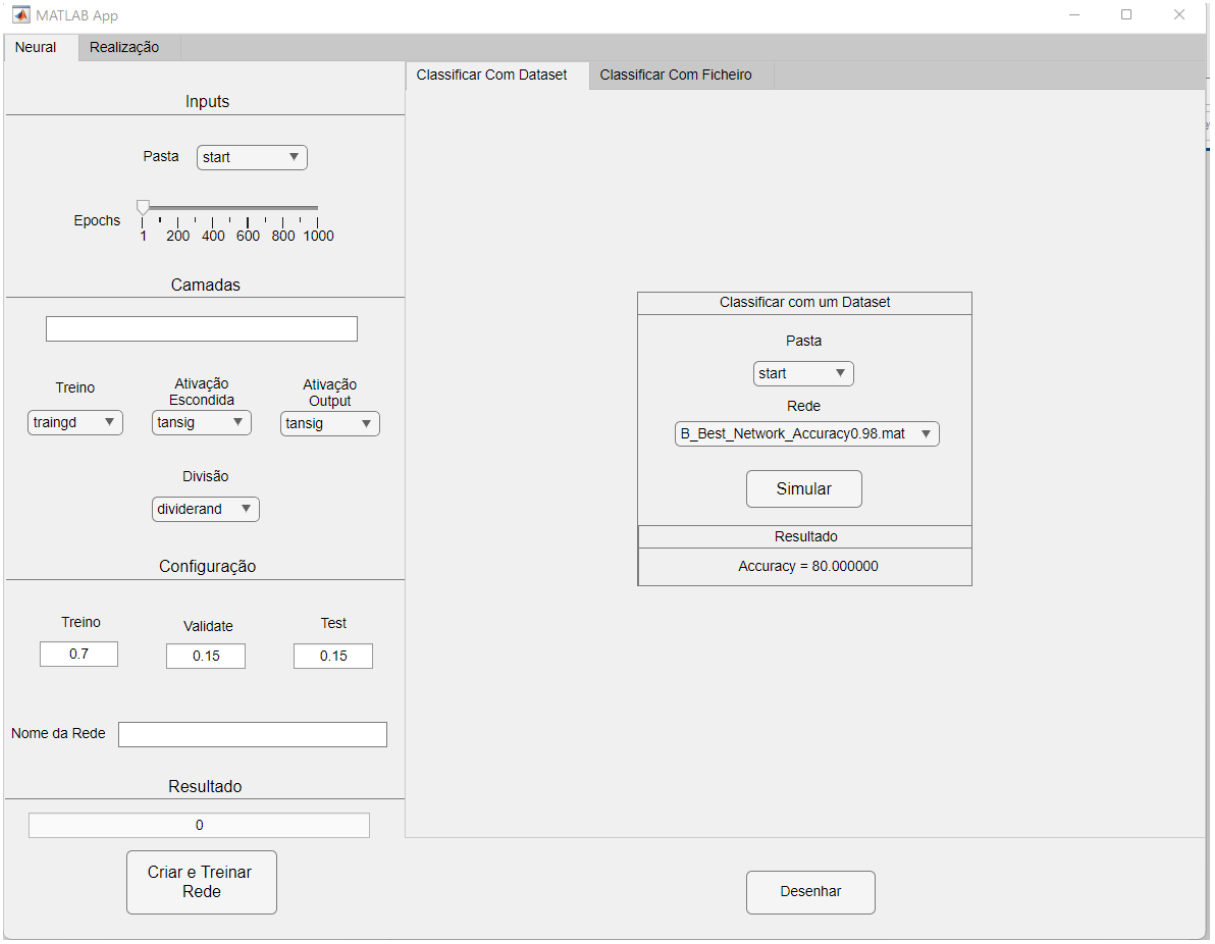
Figura da Pasta	Figura Obtida
Círculo	Kite
Círculo	Círculo
Kite	Triângulo
Kite	Círculo
Paralelogramo	Quadrado
Paralelogramo	Círculo
Quadrado	Quadrado
Quadrado	Círculo
Trapézio	Trapézio
Trapézio	Paralelogramo
Triângulo	Círculo
Triângulo	Círculo

Como podemos concluir o seu desempenho foi muito pobre, conseguindo apenas acertar em 3/12 corretamente.

Um dos fatores a referir para justificar a baixa precisão é o facto de que por norma são usados milhares de dados para treinar uma rede, enquanto neste trabalho foram usadas apenas centenas de imagens, o que não é suficiente para testes precisos.

4 - Aplicação

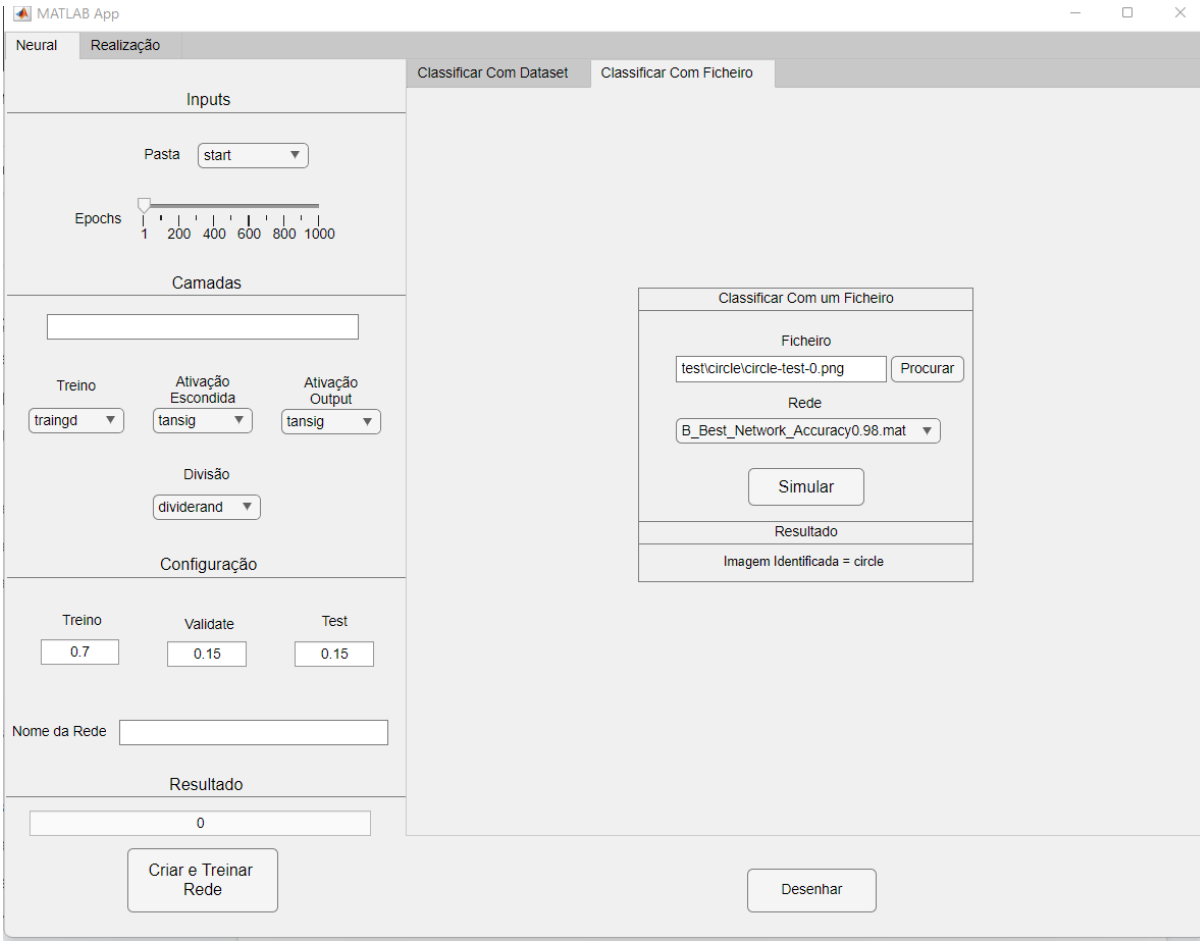
4.1 - Início



Optámos por uma interface simples e fácil de navegar, no lado esquerdo é onde é feita a criação de uma nova rede com configurações escolhidas pelo utilizador. Em conjunto com a criação da rede é feito o treinamento da mesma, no final é apresentado o resultado da sua "accuracy" da rede que será guardada na pasta com diretoria "Out/Redes/App" que depois podem ser usadas com resto das funcionalidades da App.

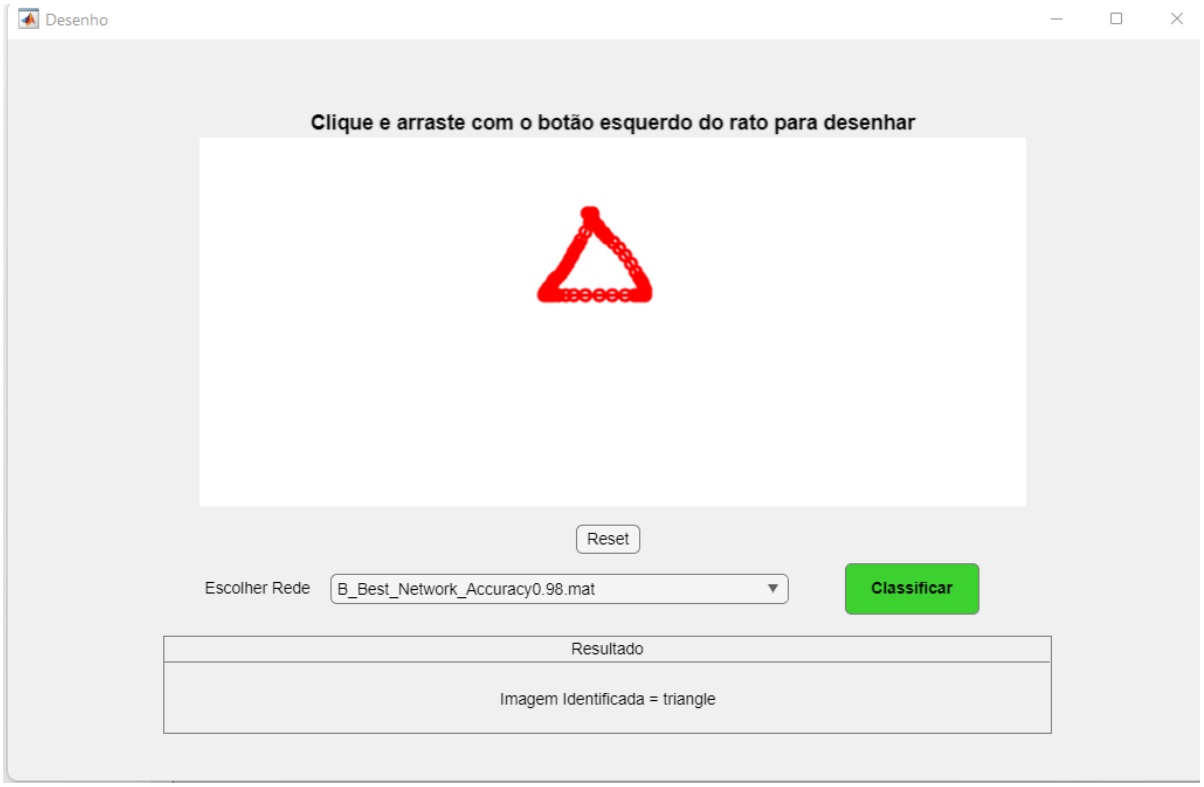
Na Tab "Classificar Com Dataset" podemos usar uma rede criada e treinada para percorrer um "Dataset" escolhido pelo utilizador, que de seguida devolve o valor da sua "accuracy" perante os dados percorridos.

4.2 - Classificação da rede com um ficheiro



O nossa outra Tab na App serve para a identificação de figuras geométricas escolhidas pelo utilizador através de redes previamente criadas e treinadas.

4.3 - Desenhar



O botão de desenho funciona semelhante à Tab antes explicada, com a diferença sendo que em vez de uma imagem escolhida pelo utilizador, é uma imagem desenhada na app que depois a classifica por uma rede antes criada e treinada.

5 - Conclusões

Com a realização do trabalho prático foi possível aprender mais sobre a utilização e capacidades das redes neuronais do tipo “feedforward”, conseguindo entender a sua importância em várias situações, sendo bastante útil por exemplo: na detecção de fraudes em cartões de crédito, diagnósticos médicos, controlo de qualidade.

Observamos que a parametrização e o tratamento prévio das imagens são características bastante importantes para um bom desempenho da rede.

6 - Bibliografia

Powerpoints informativos da Teórica
Moodle Isec

Informação sobre funções do Matlab
Mathworks