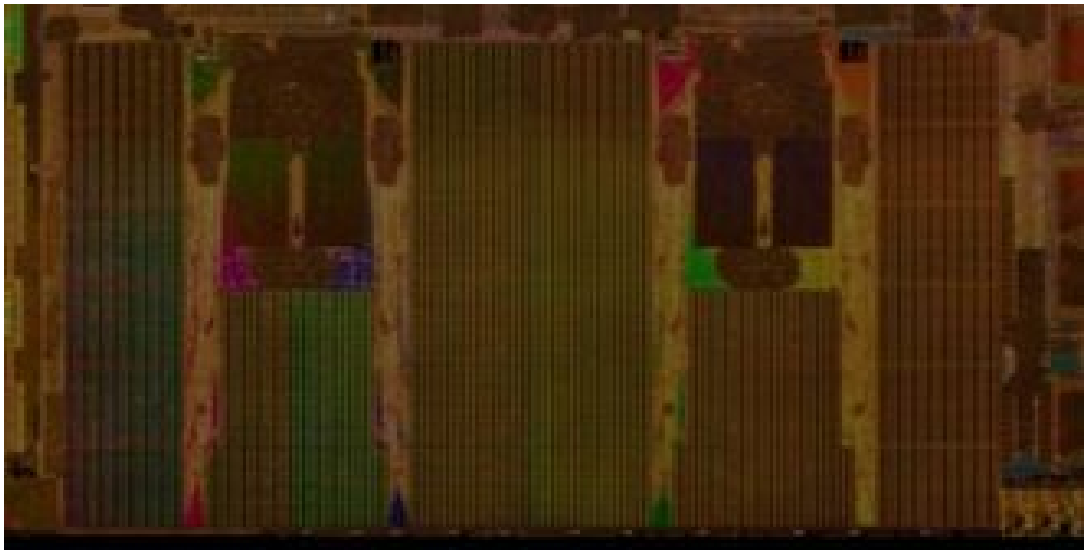
Instituto Superior de Engenharia de Coimbra

Licenciatura em Engenharia Informática
Conhecimento e Raciocínio
2018/2019
Trabalho Prático – Redes Neuronais



Nome: Pedro Miguel Franco Silvestre Número: 21240006
Nome: João Pedro Rodrigues Dias Número: 21260320

Indice

Redes neuronais **2**

Scripts **2**

Tratamento de imagem 3

Dataset generator 3

Estrutura do Sistema **3**

Treino **3**

Funções de ativação **4**

GUI **7**

Redes neuronais

Redes neuronais, são sistemas computacionais baseados numa aproximação à computação
baseada em ligações. Nós simples são interligados para formar uma rede de nós - daí o termo "rede neuronal". A inspiração original para esta técnica advém do exame das estruturas do cérebro, em particular do exame de neurónios.
O primeiro nível recebe a informação de entrada. Cada camada sucessiva recebe a saída da
camada que a precede. O último nível produz a saída do sistema. Cada nó de processamento tem sua própria pequena esfera de conhecimento, incluindo o que viu e as regras para as quais foi originalmente programado ou desenvolvido para si próprio.
As camadas são altamente interconectadas, o que significa que cada nó na camada input será conectado a muitos nós na camada hidden, que por ventura se encontram conectados na camada output. Pode haver um ou vários nós na camada de saída, a partir dos quais a resposta produzida pode ser lida.
As redes neuronais são notáveis por serem adaptativas, o que significa que elas se modificam à medida que aprendem com o treino inicial e as corridas subsequentes fornecem mais informações sobre o mundo.
Normalmente, uma rede neural é inicialmente treinada ou recebe grandes quantidades de dados. O treino consiste em fornecer entradas e dizer à rede qual deve ser a saída. Por exemplo, para construir uma rede para identificar as faces dos atores, o treinamento inicial pode ser uma série de imagens de atores, não-atores, máscaras, estátuas, rostos de animais e assim por diante. Cada entrada é acompanhada pela identificação correspondente, como nomes dos atores, informações "não-ator" ou "não-humano".

Scripts

Tratamento de imagem

Para o tratamento de cada imagem que usamos o script imageProcesser que recebe uma imagem e faz o denoise, corta a imagem pelas bordas do objeto e redimensiona para um tamanho mais pequeno.
O objetivo do denoise é limpar o lixo das imagens isto é limpar os pixels soltos à volta da imagem.
O corte é apagar os espaços a volta da forma visto que estes não são importantes para a sua identificação,
O resize é para reduzir o tamanho da imagem para reduzir o dataset, tentamos reduzir ao máximo o dataset e acelerar o processamento da rede neuronal.

Dataset generator

No dataset generator é gerado o dataset para usar na rede. Este recebe o diretório o numero de rotações, o tamanho final da imagem, uma flag para das boundaries, uma flag das hogFetatures, exportDataset e o nome do ficheiro a exportar o dataset.
A diretoria é o local na sistema de ficheiro em que está as pastar das imagens que vão ser carregadas recursivamente.
O número de rotações é o numero de rotações que uma imagem vai sofrer para ser adicionada, isto tem como objetivo aumentar o dataset e assim a rede neuronal ser mais precisa.
O tamanho é o numero da lagura e altura que a imagem vai ficar.
A flag hogFeatures é para indicar se o dataset é de hogFeatures ou se é imagem, as hogFeatures permitem reduzir o tamanho do dataset sem perder a informação das formas das ima
gens e assim reduzir o numero de inputs da rede neuronal.

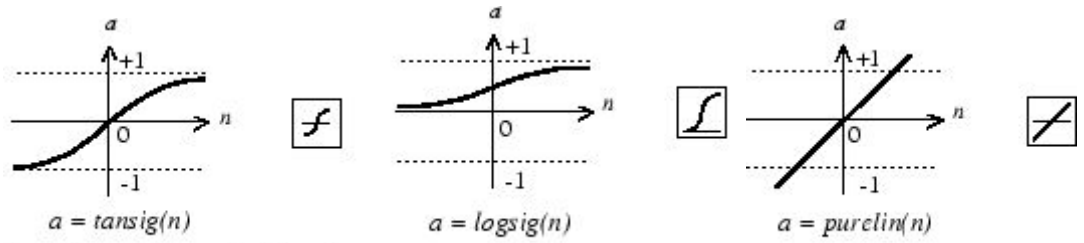
Estrutura do Sistema

Primeiro é necessário escolher um diretório com as figuras separadas por pastas e configurar os parâmetros para criar o dataset. Depois faz-se o treino da rede neuronal, e passa a ser possível testar figuras.

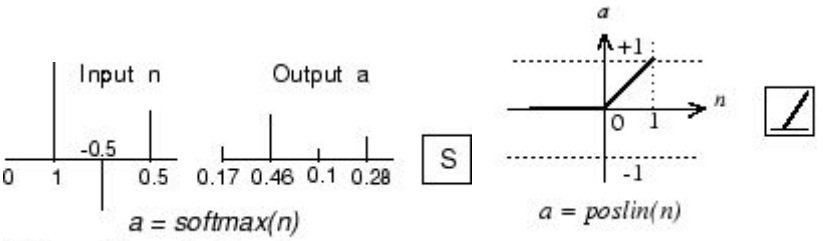
Treino

Para escolher a melhor configuração foram iteradas vários tipos de rede e configurações, após isso foram tiradas conclusões de qual será a melhor rede.

Funções de ativação



Tan-Sigmoid Transfer Function Log-Sigmoid Transfer Function Linear Transfer Function



Softmax Transfer Function Positive Linear Transfer Function

a) Comece por uma rede neuronal de uma camada com 10 neurónios. Use a rede para treinar as figuras geométricas que se encontram na pasta Formas_1. Nesta pasta encontra-se uma imagem de cada forma. Use todos os exemplos no treino. Teste outras topologias, funções de activação e de treino, registe e compare os resultados obtidos.

Formas_1	Número de camadas escondidas	Número de neurónios	Funções de ativação	Tamanho de imagem	HogFeatures ativadas	Tempo de treino	performance	gradient	epoch	precisão treino
	1	10	poslin	20 x 20	1	0	2.46	5.5	3	97.9604
	2	10, 10	poslin, poslin	20 x 20	1	0	4.26	11.6	5	99.6646
	2	10, 20	poslin, poslin	20 x 20	1	1	8.12	21.9	5	98.8239
	3	10, 10, 10	poslin, poslin, poslin	20 x 20	1	0	4.14	14.6	8	99.4416
	1	10	tansig	20 x 20	1	0	.0357	.086	5	99.4668
	2	10, 10	tansig ,	20 x 20	1	0	.805	1.26	4	99.7581

			tansig							
	2	10, 20	tansig , tansig	20 x 20	1	0	2.11	3.75	5	99.6595
	3	10, 10, 10	tansig , tansig,tansig	20 x 20	1	0	0.352	1.12	5	99.5235

b) [20%] Implemente e treine a rede neuronal para reconhecer o conjunto total de imagens da pasta Formas_2. Utilize uma segmentação do dataset de 70%, 15%, 15% para treino, validação e teste. Observe a matriz de confusão, erros de treino e teste. Explore e compare várias configurações da rede. Obtenha a melhor, registe os resultados. Grave a rede neuronal com melhor desempenho.

formas_2	**Número de camadas escondidas**	**Número de neurónios**	**Funções de ativação**	**Tamanho de imagem**	**HogFeatures ativadas**	**Tempo de treino**	**performance**	**gradient**	**epoch**	**precisão treino**
	1	10	tansig	20 x 20	1	0	0.756	1.22	20	100
	2	10, 10	tansig , tansig	20 x 20	1	1	0.918	1.48	22	100
	2	10, 20	tansig , tansig	20 x 20	1	1	1.13	2.25	16	99.9997
	3	10, 10, 10	tansig , tansig,tansig	20 x 20	1	1	0.854	1.39	5	99.9963

c) [15%] Utilize agora as imagens da pasta Formas_3 que não foram usadas no treino anterior. Sem treinar a rede verifique se a classificação dada pela RN é correta. Apresente os resultados obtidos. Posteriormente, volte a treinar a rede com estes novos exemplos, compare e registe os resultados obtidos.

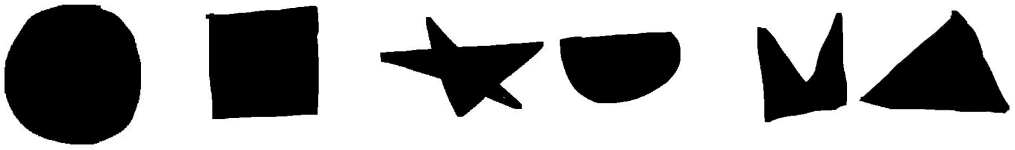
formas_3 Na melhor rede formas_2	**Número de camadas escondidas**	**Número de neurónios**	**Funções de ativação**	**Tamanho de imagem**	**HogFeatures ativadas**	**Tempo de treino**	**performance**	**gradient**	**epoch**	**precisão treino**
	10, 10	tansig , tansig	20 x 20	1	1	0.608	01.09	16	100	

O resultado é 100% de match.

Os tempos de treino são curtos devido ao tratamentos que foram feitos as imagens passamos de 4000 inputs para cerca de 40 com as hogfeatures o scaling e o corte.

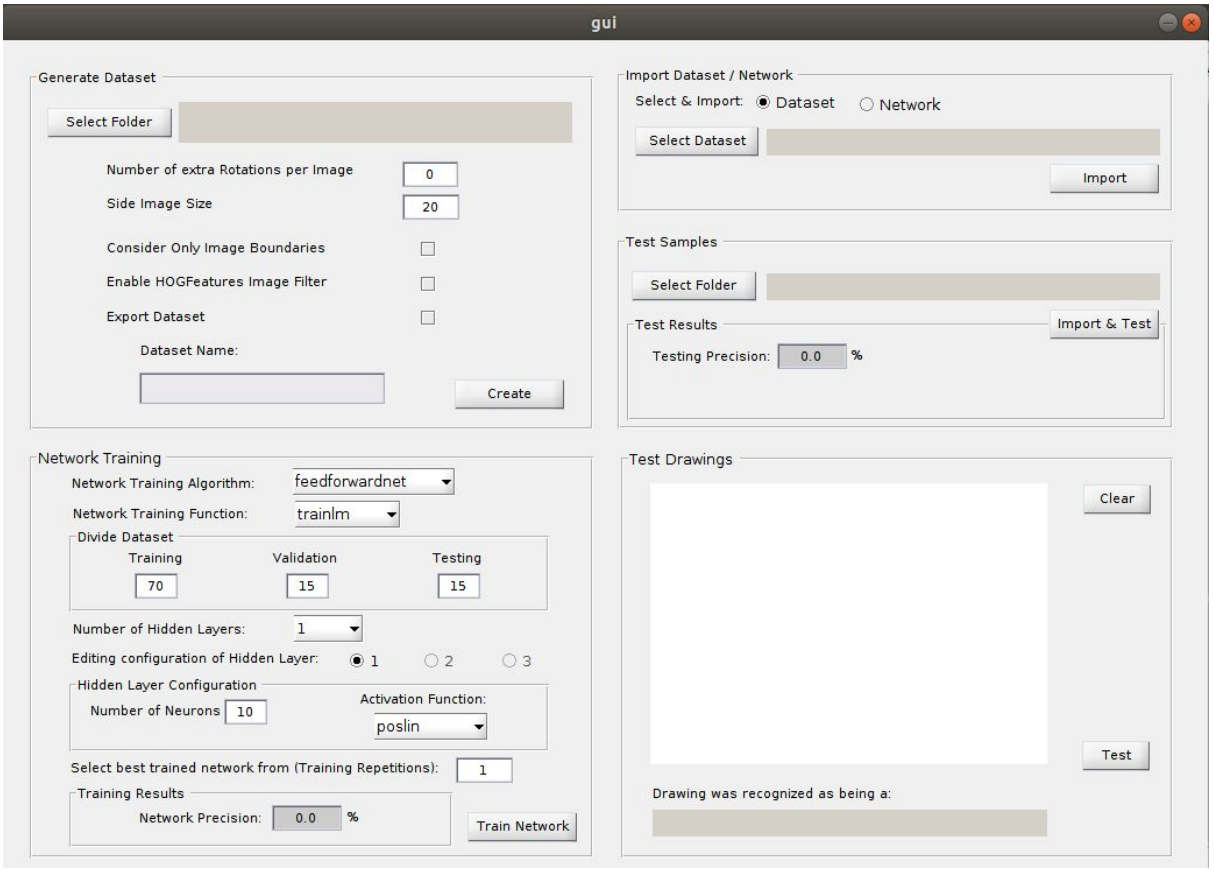
d) [15%] Desenhe manualmente algumas formas que apresentem semelhanças com os exemplos usados no treino da rede. Transcreva os desenhos para matrizes binárias. Desenvolva um pequeno programa para ler um ficheiro correspondente a uma destas imagens e aplicá-lo à rede obtida em c). Quais os resultados?

	1	2	3	4	5	6
circle	0.9069	0.5000	0.5000	0.5000	0.5000	0.5000
sqare	0.5273	1.0871	0.5000	0.8639	0.5000	0.5000
star	0.5000	0.5000	0.7395	0.5000	0.5564	0.5000
triangle	0.5000	0.5000	0.5000	0.5000	1.1962	1.1406



Aqui podemos verificar que as formas parecidas disparam os neurónios cernos e as estranhas á rede a rede assume que são outras formas pois nao conhece estas novas.

GUI



Separador generate dataset permite gerar um dataset com imagens de uma pasta.
Separador NetworkTraining Permite configurar a topologia da rede assim como as funções de ativação e os algoritmos de treino.
Separador de import dataset/network permite importar uma rede treinar ou um dataset já gerado.
Separador Test Samples permite testar a precisão da rede no reconhecimento de uma pasta de imagens é criado um novo dataset neste processo que respeite as configuraçoes do que foi usado para treinar a rede.
Test drawings permite fazer um desenho e fazer o teste na rede para identificar a forma