Universidade da Beira Interior
Departamento de Engenharia Informática

João Carlos Raposo Neves, Nº23099
Licenciatura em Engenharia Informática

Orientador de Projecto: **Prof. Doutor Hugo Proença**

Covilhã, Junho de 2011

Agradecimentos

Agradeço a todos os que me ajudaram neste trabalho nomeadamente o meu Orientador, Professor Doutor Hugo Proença, pela sua disponibilidade e compreensão com algumas falhas minhas e inexperiência e a todos os que contribuíram para a construção do meu conjunto de testes com a sua assinatura. Quero também deixar uma nota de agradecimento a todos os meus amigos e colegas que me apoiaram durante este semestre pela sua compreensão e ajuda que me deram. Finalmente, agradecer aos meus pais e família pela oportunidade de frequentar este curso e apoio prestado.

Conteúdo

Lista de Figuras

Lista de Algoritmos

Acrónimos

NGL Núcleo Geniculado Lateral

MLP MultiLayer Perceptron

MSE Erro Médio Quadrado

FF *Feed-Forward*

ADALINE *Adaptive Linear Neuron*

RPROP *Resilient Backpropagation*

Capítulo 1

Introdução

Neste trabalho o meu objectivo é desenvolver um simulador do córtex visual primário que permita a configuração de vários parâmetros de modo o utilizador pode personalizar e construir complexas redes neuronais simulando o sistema visual do cérebro humano. Este trabalho enquadra-se no âmbito do projecto de investigação PTDC/EIA/103945/2008, *NECOVID: Covert Negative Biometric Recognition*, financiado pela FCT/FEDER.

1.1 Motivação e Abordagem

A Inteligência Artificial é uma área da Informática que tenta simular a inteligência humana tentando criar máquinas que consigam dar respostas correctas a problemas complexos.

Devido ao facto de esta área ter a sua base de inspiração na inteligência humana é natural que o seu campo de investigação se cruze com áreas como a Biologia, Psicologia, Filosofia, entre outras.

A expansão e estudo desta área surgiu com o aparecimento do computador dado que este é o meio físico com capacidade suficiente de processamento para testar as teorias desenvolvidas.

A sociedade actual demonstra um interesse cada vez maior na inteligência artificial e nos benefícios que esta poderá trazer para o futuro, além de que as suas aplicações são imensas.

A Visão Computacional e as Redes Neuronais são ramos de estudo da Inteligência Artificial e o reconhecimento de padrões revelou-se e ainda continua a revelar-se uma problema de bastante difícil resolução computacional que, contudo, o nosso cérebro consegue resolver com facilidade, como reconhecer caras e os mais diversos objectos, pelo que surgiu a ideia de utilizar as redes neuronais artificias, cuja inspiração advém das redes neuronais biológicas, para a resolução de problemas com reconhecimento de padrões.

Apesar de existir uma altura em que o interesse pelas redes neuronais decaiu, o seu estudo voltou a despertar interesse na comunidade científica continuando hoje a serem actuais e alvo de vários estudos. As redes neuronais mais comuns são as redes FF sobre as quais nos vamos focar para descrever melhor os principais componentes de uma rede neuronal, mas vamos também abordar as redes convolucionais sobre as quais este trabalho se foca, dado que o objectivo deste trabalho consiste na construção e implementação de uma rede neuronal convolucional que possa ser facilmente adaptada a diversos parâmetros de configuração. Um exemplo de software que permite realizar também a configuração e construção de redes neuronais é o *MatLab* [5]. Contudo para redes em que necessitamos de uma extensa quantidade de neurónios artificiais pode não ser viável esta escolha. Existe também software dedicado apenas a esta tarefa como por exemplo o programa *NeuroSolutions* [6] e o *Topographica* [7].

1.2 Estrutura do Relatório

O relatório está divido nos diferentes capítulos:

Capitulo 1 - Introdução

Enquadramento do trabalho e da descrição sumária do trabalho e objectivos pretendidos.

Capitulo 2 - Córtex Visual Primário

Descrição dos principais componentes do Córtex Visual Primário.

Capitulo 3 - Redes Neuronais Artificias

Principais marcos na história na evolução das redes neuronais. Descrição das redes neuronais FF e convolucionais.

Capitulo 4 - Método Proposto

Descrição dos método utilizado para a resolução dos vários problemas propostos.

Capitulo 5 - Resultados

Apresentação dos resultados obtidos com as redes FF e as redes neuronais convolucionais.

Capitulo 6 - Conclusão e Trabalho Futuro

Reflexão crítica sobre o trabalho desenvolvido e orientações para futuros melhoramentos.

Capítulo 2

Córtex Visual Primário

O cérebro e todo o conjunto de operações realizadas por este sempre fascinou a humanidade. O seu estudo, na tentativa de perceber a que se deve a nossa inteligência tem tido sempre muitos adeptos. Com algum do conhecimento já adquirido tem-se tentado simular o seu funcionamento de modo a criar uma máquina inteligente. O neurónio artificial é fortemente inspirado na estrutura e funcionamento dos neurónios biológicos, tal como as redes neuronais convolucionais cuja estrutura e arquitectura é muito semelhante à do córtex visual primário, pelo que apresentamos seguidamente um breve resumo sobre as principais estruturas do córtex visual primário e suas funcionalidades.

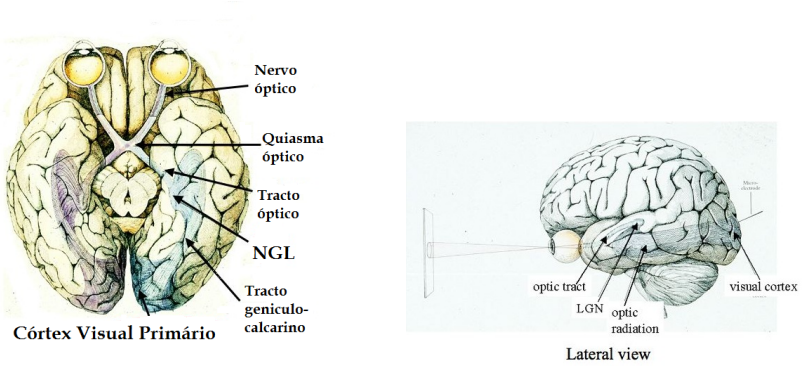


Figura 2.1: *Áreas referentes à visão do cérebro humano [1].*

2.1 Retina

O olho humano é o componente que faz a tradução de estímulos luminosos para estímulos eléctricos graças aos foto-receptores estando divididos em cones (mais sensíveis à cor) e bastonetes (mais sensíveis à luminosidade).

Como podemos observar pela imagem os foto-receptores são a última de três camadas que constituem a retina, contudo estes são os únicos capazes de processar a luz em impulsos luminosos sendo a resposta destes posteriormente enviada para a camada superior (Células Bipolares) e destas para as Células Ganglionares de modo a que a resposta seja encaminhada pelo nervo óptico para o cérebro.

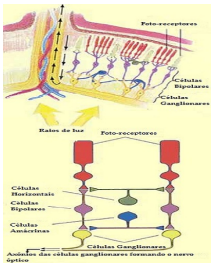
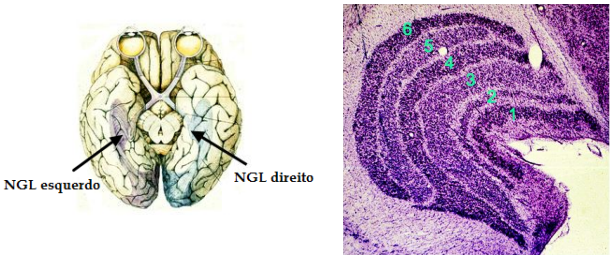


Figura 2.2: *Estrutura da Retina [2].*

2.2 Núcleo Geniculado Lateral

Após as respostas das Células Ganglionares enviarem a resposta para o nervo óptico estas chegam ao Núcleo Geniculado Lateral (NGL) onde os estímulos são processados de uma forma muito semelhante às células ganglionares. O NGL localiza-se no centro do cérebro existindo uma parte direita e esquerda como é ilustrado na figura 2.3(a). Ambos faz a comunicação entre as camadas retinais e o Córtex Visual Primário e dividem-se em 6 camadas identificadas na figura 2.3(b). Uma característica destas células, que se designa por retinotopia, é facto de cada área da retina ser mapeada numa área correspondente do NGL estando organizadas da mesma forma pelo que áreas adjacentes na retina correspondem áreas adjacentes no NGL.



(a) (b)

Figura 2.3: *Estrutura do Núcleo Geniculado Lateral [2].*

As primeiras quatro camadas são constituídas por células mais pequenas (Células Parvo) enquanto que as últimas duas são constituídas por células maiores (Células Magno).

2.3 Córtex Visual Primário

O Córtex Visual Primário situa-se na parte posterior do cérebro.

O Córtex Visual Primário é a parte principal do cérebro que possibilita o reconhecimento de padrões graças ao tipo de células neste existentes.

Estas células foram catalogadas em células da camada 4, células simples e células complexas, sendo as células da camada 4 as que recebem as respostas finais do NGL.

As células da camada 4 comportam-se de forma semelhante às ganglionares da retina, contundo o mesmo não acontece nas células simples, sendo estas que contribuem grandemente para o reconhecimento de padrões e formas no cérebro humano, já que estas são bastante sensíveis às orientações de linhas e bordas.

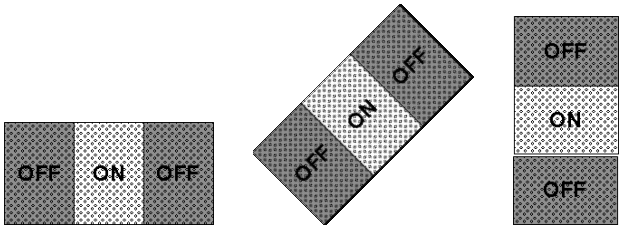


Figura 2.4: *Diagrama do campo receptivo de três células simples com receptividade a diferentes orientações.*

As zonas ON representadas indicam as zonas da célula onde deve existir um impulso energético para que a célula fique activa enquanto que na zona OFF deve existir um impulso com baixa energia de modo a que o impulso energético se concentre apenas na zona ON. Na imagem 2.4 vemos que o campo receptivo de cada célula depende neste caso da orientação para a qual a célula esta "programada"a detectar o que permite detectar a existência os limites de objectos na imagem (diferenças de intensidade na imagem).

O seguinte exemplo pretende evidenciar melhor como isto acontece.

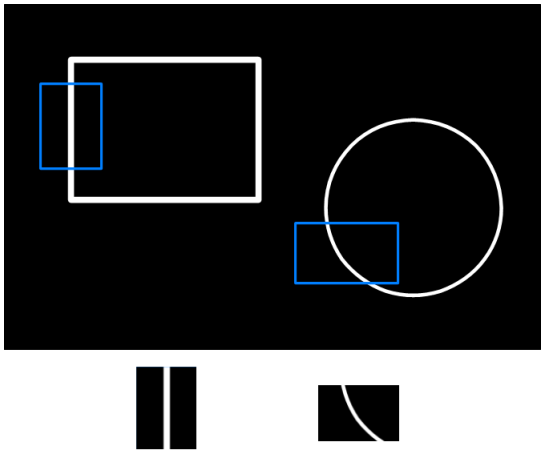


Figura 2.5: *Um exemplo particular de como as células simples podem ser sensíveis a certas porções de uma imagem.*

Na figura 2.5 foram retirados dois segmentos, o primeiro corresponde a

uma borda vertical em que existe maior energia no centro da imagem. É este padrão que faz accionar as células simples representadas na figura 2.4(a) mas não nenhumas das outras, dado que a parte energética da imagem não vai corresponder de forma satisfatória à zona ON da célula simples. Para o caso do segundo segmento este fará accionar a célula simples da figura 2.4(c) pelas mesmas razões explicadas acima.

Capítulo 3

Redes Neuronais Artificias

3.1 Introdução às Redes Neuronais Artificiais

Os mistérios do cérebro sempre intrigaram a espécie humana, mas apesar do estudo do cérebro ter começado alguns séculos antes de Cristo com os gregos não há respostas completas de como o cérebro pode processar os dados recebidos e retornar uma resposta inteligente [8].

Com a ascensão da electrónica uma nova ideia foi proposta por Warren McCulloch (neuro-fisiologista) e Walter Pitts (matemático) em 1943. A sua ideia era de modular o neurónio humano com circuitos lógicos e reuni-los para formar uma rede neural artificial, já que na sua opinião a capacidade de processamento do cérebro baseava-se no facto de ao juntar um extenso número de neurónios que embora apenas pudessem dar uma de duas respostas (activo ou inactivo) em conjunto poderiam dar respostas muito mais complexas [17].

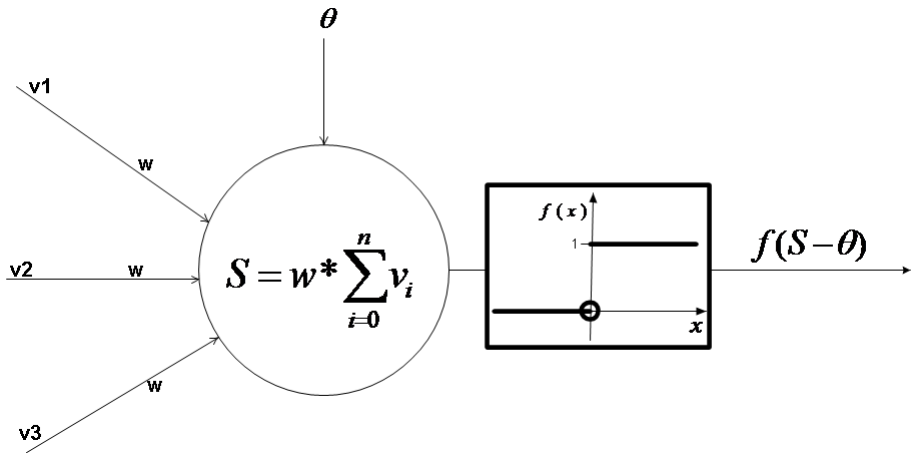


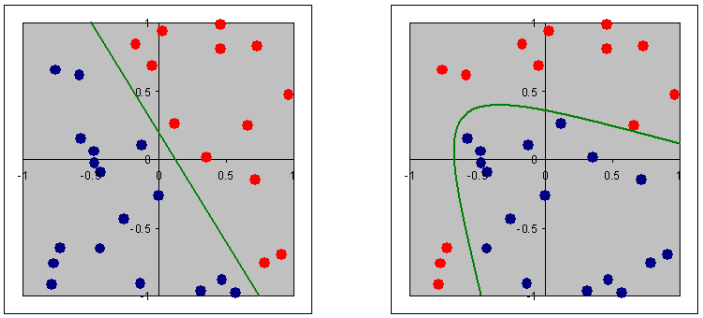
Figura 3.1: *O modelo do neurónio de McCulloch-Pitts.*

Na figura 3.1 podemos observar o modelo de funcionamento de um neurónio de McCulloch-Pitts em que cada entrada é pesada por um peso comum a todas as entradas sendo o somatório destas pesadas juntamente com θ passado para dentro da função de transferência, que neste caso é sempre a função *threshold* para que a saída do neurónio possa ser sempre 0 ou 1.

Em 1958, Frank Rosenblatt propôs o *perceptron* [21], um modelo bastaste semelhante ao do neurónio de McCulloch e Pitts, contudo com a diferença que cada entrada para o neurónio era pesada com o seu o próprio peso mas mais importante foi a proposta de uma regra de aprendizagem para resolução de problemas com o *perceptron*.

Esta regra é bastante semelhante à regra delta proposta por Widrow para o *Adaptive Linear Neuron* (ADALINE) [23], um modelo baseado no neurónio de McCulloch-Pitts. A regra consistia em ajustar cada peso consoante a diferença entre a resposta retornada pelo neurónio e a resposta esperada de tal modo que após algumas épocas de treino o *perceptron* converja os pesos para os valores que conseguem separar de forma correcta as classes do problema.

Cada *perceptron* consegue fazer separação entre 2 classes, dado que este tem um resposta binária, contudo em 1969 Marvin Minsky e Papert Seymour publicaram o livro *Perceptrons* [18] em que denotaram que apenas os problemas linearmente separáveis eram resolúveis pelo *perceptron* de tal modo que a partir daí o interesse pelas redes neuronais decaiu.



(a) Classes que podem ser separadas linearmente.

(b) Classes que não podem ser separadas linearmente.

Figura 3.2: *Exemplo de dois tipos de problemas [3].*

Na figura 3.2(b) um simples *perceptron* é incapaz de definir uma linha de separação entre as duas classes.

Após o livro *Perceptrons* a investigação em redes foi quase nula, voltando novamente a ressurgir na década de 80 iniciada com a publicação de Hopfield [15] que introduziu as redes de Hopfield que se baseavam em modelos de memória associativa. Nestas redes cada neurónio está ligado a todos os outros excepto ele mesmo estando um peso associado a cada conexão como vemos na figura 3.3.

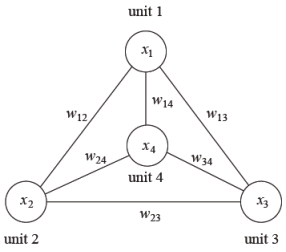


Figura 3.3: *Exemplo de uma rede de Hopfield com 4 neurónios[20].*

Além de Hopfield a contribuição de Rumelhart com o algoritmo de retropropagação do erro juntamente com o MultiLayer Perceptron (MLP) [22] foi decisiva para uma interesse crescente nesta área.

O MLP resulta na junção de vários *perceptrons* em várias camadas sendo o seu número escolhido em função do problema a resolver. Graças a esta reunião de *perceptrons* embora cada um deles apenas consiga distinguir entre classes linearmente separáveis em conjunto o seu poder de discriminação aumenta dado que aumenta também os graus de liberdade da rede. Deste modo o MLP consegue resolver problemas muito mais complexos que os linearmente separáveis desde que tenha um número suficiente de neurónios nas camadas intermédias.

Para que os MLP possam aprender foi proposto o algoritmo de retro-propagação do erro em que a grande diferença em relação aos algoritmos de aprendizagem do *perceptron* e do ADALINE baseia-se na sua simplicidade e flexibilidade de se ajustar a qualquer tipo de arquitectura da rede. Basicamente o objectivo do algoritmo é minimizar um função de erro de modo a optimizar cada peso da rede para que esta possa separar perfeitamente as diferentes classes do problema.

Depois disto as redes neuronais artificiais tiveram um grande desenvolvimento e interesse por parte da comunidade científica, dado que a partir do aparecimento do MLP juntamente com o algoritmo de retro-propagação do erro conseguiu-se resolver vários problemas recorrendo a este tipo de redes.

3.2 Redes Neuronais FF

As redes neuronais podem ser divididas em redes FF e redes recorrentes em que o *perceptron* e o MLP são exemplos de uma rede FF e as redes de Hopfield exemplo de uma rede recorrente. A principal diferença entre estas reside no facto de nas redes FF as saídas de cada neurónio são enviadas sempre para neurónios da camada seguinte enquanto que nas redes recorrentes a resposta pode ser enviada para camadas anteriores sendo a rede realimentada com o seu próprio *output*. Neste capitulo vamos focar a nossa atenção nas redes FF.

3.2.1 Arquitectura

Numa rede neuronal existem normalmente três tipos de camadas, a camada de entrada, as camadas escondidas, e a camada de saída.

Os neurónios da camada de entrada são responsáveis por introduzir na rede os dados a processar, dado que o seu potencial vai ser o valor dos dados de entrada. Após estes dados estarem carregados na camada de entrada os neurónios da primeira camada escondida utilizam estes valores para calcular o seu potencial da seguinte forma:

$$f(\sum_{i=1}^{N} w_{ik}.y_i + b), \tag{3.1}$$

sendo N o tamanho da camada anterior, w_{ik} o peso da ligação entre n_i e n_k, b o valor de *bias* de n_k, y_i a saída do neurónio n_i e f a função de activação de n_k.

Entre cada dois neurónios caso exista uma ligação esta é sempre pesada por um valor w_{ik} que é maleável ao conjunto de dados apresentado, pois após cada época cada peso é ajustado de modo a minimizar o erro da rede. É maleabilidade dos pesos que permite à rede ajustar-se ao um problema.

Para além dos pesos temos também o valor b que designamos por *bias* que é um valor de ajuste que cada neurónio possui para o cálculo do seu potencial e é modificado como um peso normal durante o treino da rede.

3.2.2 Tipo de aprendizagem

Para treinar a rede de modo a que esta optimize os seus pesos para o problema em questão temos duas abordagens diferentes que podemos utilizar que vamos descrever a seguir.

Aprendizagem Supervisionada

Neste tipo de aprendizagem a actualização dos pesos da rede é feita em função das respostas que está dá em comparação com as que deveria dar, ou seja, para cada exemplo do conjunto de treino é necessário saber a

verdadeira resposta de modo a que se possa saber quão distante está a rede da resposta desejada e orientar a actualização dos seus pesos nesse sentido.

Aprendizagem Não-Supervisionada

Por oposição ao tipo de aprendizagem anterior, neste caso a rede não tem forma de saber o quão bem ou mal vai a aprendizagem dado que não possui os valores correctos para cada exemplo do treino. A rede deve tentar reconhecer padrões nos exemplos que vai recebendo para posteriormente tentar reconhecer características nos exemplos de teste que a aprendeu nos exemplos de treino.

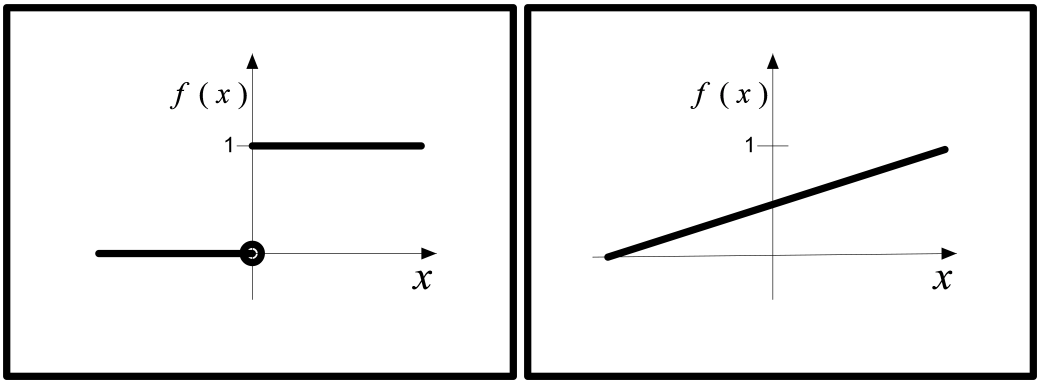
3.2.3 Funções de Activação

Como já foi referido anteriormente cada neurónio tem uma função que processa o produto interno entre os pesos e as entradas dos neurónios. Uma das já faladas é a função "escada"ou "threshold"presente no neurónio de McCulloch-Pitts.

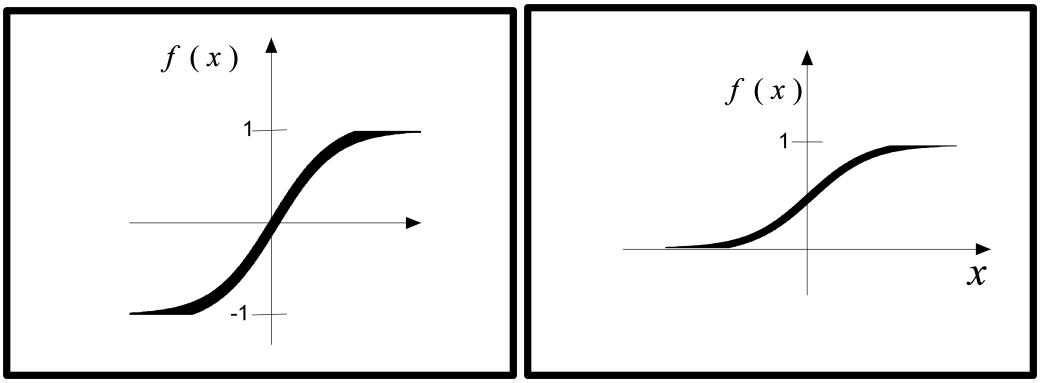
Além desta temos as funções Linear, Sigmóide e Tangente Hiperbólica representadas na figura 3.4 que são as mais utilizadas para o processamento dos dados na rede neuronal.

A escolha destas funções não foi fruto do acaso mas baseou-se no fundamento biológico do potencial de acção de cada neurónio biológico depender de um certo valor de estimulo a que nas funções corresponde ao x. Uma função que exemplifica bem este facto é a função "escada", figura 3.4(a) que fica activa, ou seja toma o valor 1 (ON), quando o estimulo é positivo . Contudo tal como vamos verificar seguidamente apenas funções diferenciáveis são passíveis de utilizar no algoritmo de aprendizagem, o que excluí a utilização desta função em redes multi-camadas sendo possível utiliza-la em redes de camada única.

No caso da função linear o problema da diferenciabilidade já não existe, contudo levanta o problema de não existir um potencial máximo para o neurónio dado que o contradomínio de uma função linear é $\mathbb{R}$ pelo que o ideal seria uma função com contradomínio limitado e diferenciável tal



(a) Função 'escada' (b) Função Linear



(c) Função Tangente Hiperbólica (d) Função Sigmóide

Figura 3.4: *Funções de activação utilizadas nos neurónios de uma rede.*

como a função sigmóide cujo contradomínio é [0,1] e a tangente hiperbólica com contradomínio [-1,1]. Além disso estas duas funções assemelham-se bastante à função 'escada' com a vantagem de serem diferenciáveis simulando assim a activação do neurónio biológico com o estimulo adequado.

Para além deste facto as derivadas das funções sigmóide e tangente hiperbólica podem ser expressas em função da função original como podemos ver a seguir, onde a f(x) é a equação da sigmóide e g(x) a da tangente hiperbólica.

$$f(x) = \frac{1}{(e^{-x}+1)} \tag{3.2}$$

$$\begin{aligned} f'(x) &= \frac{d}{dx}\frac{1}{(e^{-x}+1)} \\ &= \frac{(e^{-x}+1)\frac{d}{dx}1 - 1\frac{d}{dx}((e^{-x}+1)}{(e^{-x}+1)^2} \\ &= \frac{0-(-e^{-x}+0)}{(e^{-x}+1)^2} \\ &= \frac{e^{-x}+1-1}{(e^{-x}+1)^2} \\ &= \frac{e^{-x}+1}{(e^{-x}+1)^2} - \frac{1}{(e^{-x}+1)^2} \\ &= \frac{1}{(e^{-x}+1)} - \frac{1}{(e^{-x}+1)}\frac{1}{(e^{-x}+1)} \\ &= \frac{1}{(e^{-x}+1)}(1 - \frac{1}{(e^{-x}+1)}) \\ &= f(x)(1-f(x)) \end{aligned} \tag{3.3}$$

$$g(x) = \tanh x = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{3.4}$$

$$\begin{aligned} g'(x) &= \frac{d}{dx}\frac{(e^x-e^{-x})}{(e^x+e^{-x})} \\ &= \frac{(e^x+e^{-x})\frac{d}{dx}(e^x-e^{-x}) - (e^x-e^{-x})\frac{d}{dx}(e^x-e^{-x})}{(e^x+e^{-x})^2} \\ &= \frac{(e^x+e^{-x})(e^x+e^{-x}) - (e^x-e^{-x})(e^x-e^{-x})}{(e^x+e^{-x})^2} \\ &= \frac{(e^x+e^{-x})(e^x+e^{-x})}{(e^x+e^{-x})^2} - \frac{(e^x-e^{-x})(e^x-e^{-x})}{(e^x+e^{-x})^2} \\ &= 1 - \frac{(e^x-e^{-x})}{(e^x+e^{-x})}\frac{(e^x-e^{-x})}{(e^x+e^{-x})} \\ &= 1 - g(x).g(x) \end{aligned} \tag{3.5}$$

Como vemos cada uma das derivadas pode ser escrita em função da própria função o que facilita bastante o processamento de alguns passos importantes no algoritmo de aprendizagem como vamos ver seguidamente.

3.2.4 Algoritmo de Aprendizagem

Como já referimos acima, com o surgimento do MLP surgiu também o algoritmo de retro-propagação do erro de modo a que a rede pudesse optimizar os seus pesos em função do problema. Este tem como objectivo introduzir nos pesos da rede o ajustamento necessário de modo a que a

diferença entre a resposta obtida e a resposta esperada seja cada vez menor à mediada que a rede é treinada.

Após a fase de propagação é necessário verificar quão longe está a rede das respostas correctas para os vários casos de treino, sendo o erro das camadas finais calculado por essa diferença enquanto que o erro dos restantes neurónios é calculado a partir dos neurónios da sua camada seguinte, pelo que deste modo os neurónios da penúltima camada utilizam o erro dos da ultima e os do camada n os da camada $n + 1$ sendo o erro passado de camada em camada até chegar à primeira. Para o calculo deste erro é também necessária calcular $f'(x)$, a derivada da função de activação para o valor x que corresponde ao somatório dos pesos pelas entradas do anterior, mas como já referimos anteriormente as derivadas das nossas funções de activação utilizadas podem se expressas a partir da função original o que vai facilitar o calculo de $f'(x)$ já que por exemplo para a função sigmóide $f'(x) = p_i(1 - p_i)$, sendo p_i o potencial do neurónio para o qual estamos a calcular o erro.

Algoritmo 1 Algoritmo *BackPropagation*.

Inicializar os pesos da rede aleatoriamente.
$epoca = 1$

while $epoca > NrMaxEpocas$ **do**
for $z = 1 \rightarrow size(ConjuntoTreino)$ **do**
Atribuir os valores de $ConjuntoTreino_z$ aos neurónios da camada de entrada.

Para todas as camadas seguintes a saída de cada neurónio n_k é dada por:

$$f(\sum_{i=1}^{N} w_{ik}.y_i + b),$$

sendo N o tamanho da camada anterior,w_{ik} o peso da ligação entre n_i e n_k e f a função de activação de n_k.

Calcular erro de propagação ε da rede para cada neurónio n_k da camada final sendo $\varepsilon_k = d_k - y_k$,onde d_k é a saída desejada para n_k e y_k a sua saída.

Calcular o erro de cada neurónio da seguinte forma:

$$erro(n_k) = \begin{cases} \varepsilon_k \frac{df(y_k)}{dx} & \text{se } n_k \in \text{ultima camada} \\ \sum_{j=1}^{N} w_{kj}.erro(n_j)\frac{df(y_k)}{dx} & \text{se } n_k \notin \text{ultima camada} \end{cases}$$

end for
Actualizar cada peso segundo a fórmula:

$$w_{ik}(epoca + 1) = w_{ik}(epoca) + \eta.erro(n_k).y_k \ ,$$

onde η é a taxa de aprendizagem de n_k.
$epoca = epoca + 1$
end while

3.3 Redes Neuronais Convolucionais

Este tipo de rede surgiu devido à motivação biológica presente no trabalho de Hubel e Wiesel [16] em que ao fazerem testes com o Córtex Visual de gatos descobriram que as células simples e complexas são sensíveis a certos padrões e orientações, como já foi explicado no capitulo acima.

As redes neuronais convolucionais consistem num conjunto de camadas que extraí características das imagens de entrada fazendo sucessivas convoluções e redimensionamentos de modo que no final consigamos apenas ficar com a marca da classe a que a imagem de entrada pertence.

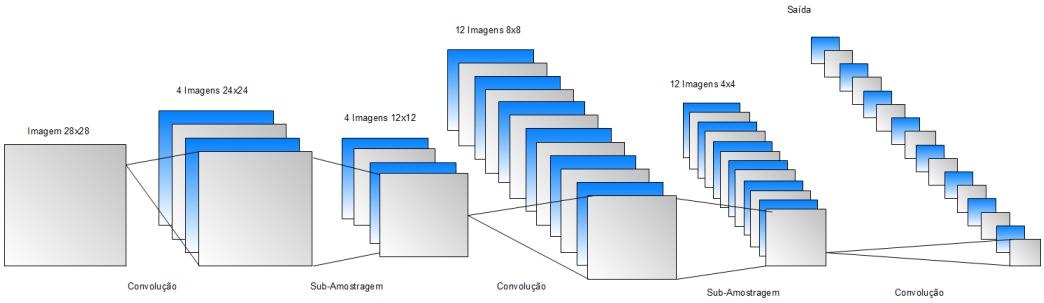


Figura 3.5: *Exemplo de uma rede convolucional[4].*

A figura 3.5 ilustra um exemplo de uma rede que recebe uma imagem normalizada de 28x28 que é alvo de uma convolução com um filtro de 5x5 que dá origem ao primeiro plano da primeira camada sendo os restantes três planos resultado da convolução com outros filtros de modo a extrair da mesma imagem diferentes características.

Após a fase da convolução segue-se a uma sub-amostragem de modo a diminuir a quantidade de informação e agregar as características recolhidas pela convolução. Seguidamente, repete-se o mesmo processo de convolução e sub-amostragem até obter uma camada final com o tamanho desejado, normalmente 1x1.

O *Neocognitron* proposto Kunihiko Fukushima [10] é um bom exemplo de uma rede neuronal convolucional utilizada para o reconhecimento de dígitos. Nesta rede Fukushima criou camadas S e C cujos neurónios têm funções bastantes semelhantes às células simples e complexas do córtex

visual primário respectivamente. Deste modo, o *Neocognitron* tenta simular o processamento de informação visual que o nosso cérebro realiza tentado que as camadas S retirem características da imagem como bordas, arestas e orientações das mesmas, enquanto que as camadas C tentam generalizar as informações enviadas pelas camadas S garantindo alguma invariância em relação à posição do padrão que as células S tentam transmitir para a próxima camada.

Numa rede convolucional como o *Neocognitron* podem existir vários conjuntos de camadas S e C como é representado na figura 3.6

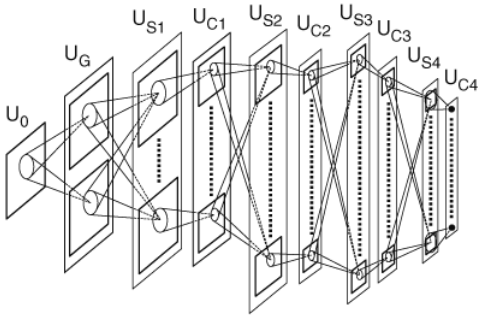


Figura 3.6: *Arquitectura de uma rede neocognitron[4].*

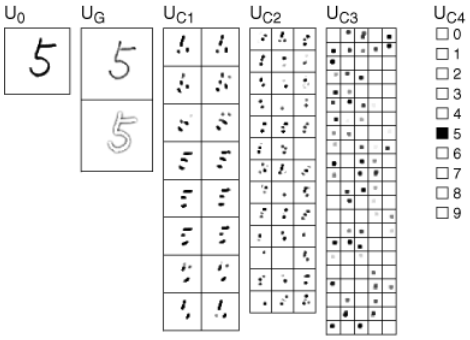


Figura 3.7: *Características recolhidas nas diferentes camadas de uma rede neocognitron[4].*

À medida que avançamos nas camadas da rede as características começam a ser cada vez mais globais em função do dígito inicial independentemente

da sua posição e escala como podemos ver pela figura 3.7. Na camada complexa 1 ainda conseguimos notar as arestas do dígito mas nas camadas seguintes as características extraídas são mais globais o que no final garante que possa existir reconhecimento. Com este tipo de rede Fukushima conseguiu obter resultados bastante satisfatórios no reconhecimento de dígitos manuscritos [11] sendo depois posteriormente melhorado pelo autor [12] [13].

Capítulo 4

Método Proposto

Para criar um simulador do córtex visual primário foi necessário uma investigação prévia sobre todos os tópicos até aqui já discutidos sobre os quais nos inspirámos para construir a nossa rede convolucional.

Seguidamente vamos descrever toda a arquitectura da nossa rede e o seu funcionamento.

4.1 Modelo da Rede

A tentativa de abordar um problema de reconhecimento de padrões utilizando redes FF seria falhada dado que este tipo de problemas envolvem sempre imagens que contêm demasiada informação pois cada *pixel* serviria de entrada para a rede levando a que o treino fosse muito mais lento e também a que houvesse um excesso de pesos na rede o que poderia influenciar o desempenho da rede. Desta forma, é necessário primeiro filtrara a imagem extraindo algumas características da imagem de modo a retirar apenas as marcas chave que possam garantir o reconhecimento da mesma.

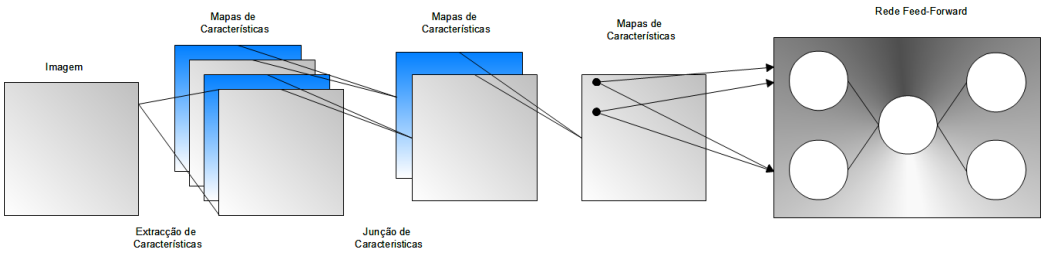


Figura 4.1: *Modelo da rede construída.*

4.2 Extracção de características

Para esta fase a propusemos a utilização de filtros que consigam retirar da imagem características importantes como bordas e diferenças de intensidade além de que sejam sensíveis à orientação dessas mesmas bordas. Deste modo encontramos nas propriedades do filtros de *Gabor* uma boa solução para o nosso problema pelas razões que vamos ver a seguir.

4.2.1 Filtros de *Gabor*

Este tipo de filtros foram propostos por Dennis Gabor [14] contudo para propósitos diferentes dos que posteriormente Daugman os utilizou quando descobriu que estes sinais podiam modelar a resposta das células simples do córtex visual primário [9].

As propriedades destes filtros permitem que sejam sensíveis à orientações de padrões e bordas na imagem tal como as células simples.

Os filtros de *Gabor* podem ter várias formas e configurações dado que podemos especificar para um filtro diversos parâmetros como por exemplo a sua frequência e a sua excentricidade.Em 4.1 e 4.2 apresentamos as equações das parte real e imaginária dos filtros de *Gabor* , onde f representa a frequência do sinal , θ a orientação do filtro , ψ representa a fase do sinal, γ a excentricidade do aspecto da curva e σ controla o achatamento das ondas.

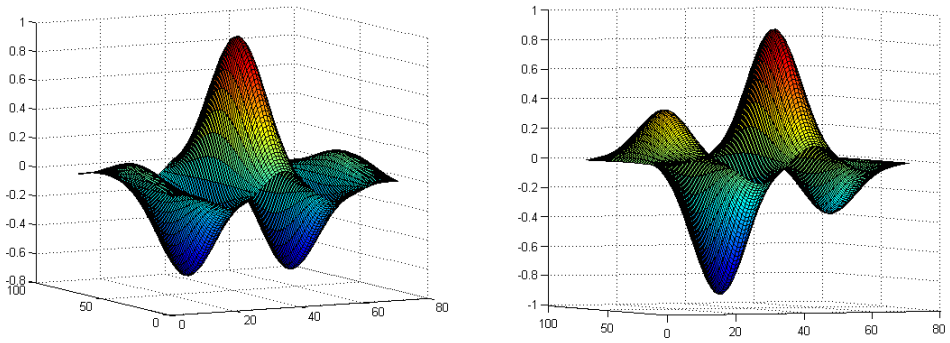
$$G_{f,\theta,\psi,\sigma,\gamma}(x,y) = e^{-\frac{x'^2+\gamma y'^2}{2\sigma^2}} \cos(2\pi x' f + \psi) \tag{4.1}$$

$$G_{f,\theta,\psi,\sigma,\gamma}(x,y) = e^{-\frac{x'^2+\gamma y'^2}{2\sigma^2}} \sin(2\pi f x' + \psi), \tag{4.2}$$

onde

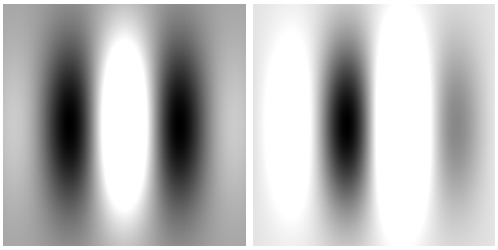
$$x' = x\cos\theta + y\sin\theta$$
$$y' = -x\sin\theta + y\cos\theta$$

Na figura 4.2(a) e 4.2(b) apresentamos a representação das funções 4.1 e 4.2 no plano xyz e na figura 4.3(a) e 4.3(b) apresentamos a sua representação matricial, sendo esta a forma mais comum de apresentação dos filtros de *Gabor*.



(a) Parte Real (b) Parte Imaginária

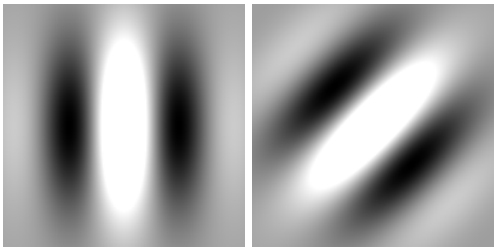
Figura 4.2: *Representação tridimensional das funções de Gabor.*



(a) Parte Real (b) Parte Imaginária

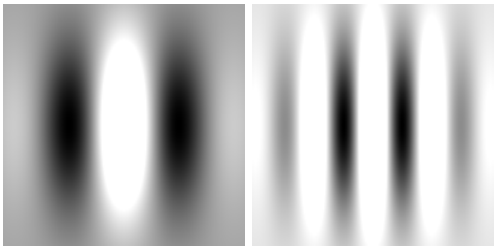
Figura 4.3: *Representação das funções de Gabor na forma matricial.*

Nas figuras 4.4(a) e 4.4(b) podemos observar a influência do parâmetro θ no aspecto do filtro alterando a direcção das linhas energéticas do filtro, enquanto que nas figuras 4.5(a) e 4.5(b) vemos que ao aumentar a frequência do sinal obtemos um filtro com mais oscilações entre extremos positivos e negativos.



(a) $\theta = 0$ (b) $\theta = \frac{\pi}{4}$

Figura 4.4: *Filtros de Gabor com diferentes orientações.*



(a) $f = 0.01$ (b) $f = 0.02$

Figura 4.5: *Filtros de Gabor com diferentes frequências.*

4.2.2 Fase de extracção de características

Tal como o exemplo apresentado em 2.5, os filtros de *Gabor* podem ser sensíveis a determinado contorno caso estejam orientados na direcção do contorno, pelo que na nossa rede utilizámos filtros com várias orientações.

Nesta primeira fase de extracção utilizámos unidade básicas de processamento que designámos de células simples artificias.

Cada uma destas unidades possui um filtro de *Gabor* com uma certa orientação, frequência e tamanho, sendo o valor do seu potencial o resultado da convolução com uma determinada região da imagem de entrada.

As células simples artificias estão organizadas em camadas de modo que cada camada possui células com filtros semelhantes variando apenas a região da imagem a que cada célula está afecta.

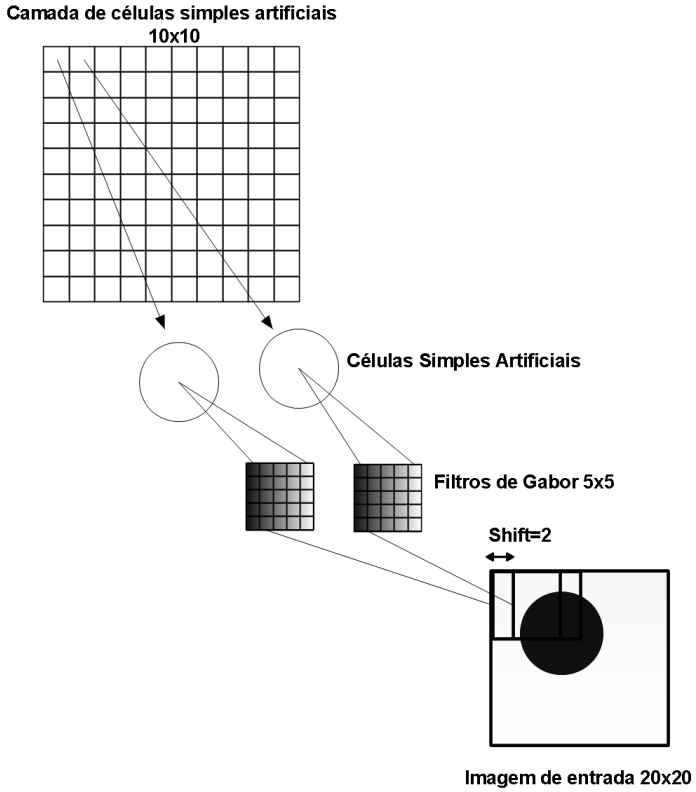


Figura 4.6: *Exemplo de uma camada da fase de extracção de características.*

A figura 4.6 exemplifica como se processa a extracção de características da imagem de entrada para uma camada de células simples. Os filtros são aplicados de x em x posições, sendo x um parâmetro da rede que denominamos *Shift*. O tamanho da camada de células é determinado em função do valor do *Shift* sendo no máximo igual ao tamanho da imagem quando $Shift = 1$.

Para a fase de extracção o número de camadas é variável dado que cada

camada procura na imagem por características diferentes em função dos parâmetros definidos para o filtro utilizado nessa camada.

Após o processamento das células simples ter terminado estas enviam a sua resposta para a próxima fase em que se processa a junção da características.

4.2.3 Fase de junção de características

Nesta fase é necessário reunir as características encontradas de modo que vamos agrupando-as por fases até que no final desta fase fiquemos apenas com uma camada de características.

A figura 4.7 tenta exemplificar como se processa esta junção em que primeiramente é feita uma junção das camadas que possuem filtros com orientações diferentes formando um conjunto de n camadas , sendo n o número de diferentes tamanhos utilizados na fase de extracção de características.No exemplo apresentado foram utilizados 2 diferentes tamanhos e 2 orientações diferentes pelo que a primeira fase de junção de características terá duas camadas correspondentes aos 2 tamanhos de filtros diferentes. Após ser feita a junção por orientação reunimos as camadas resultantes numa só fazendo a junção por tamanho.

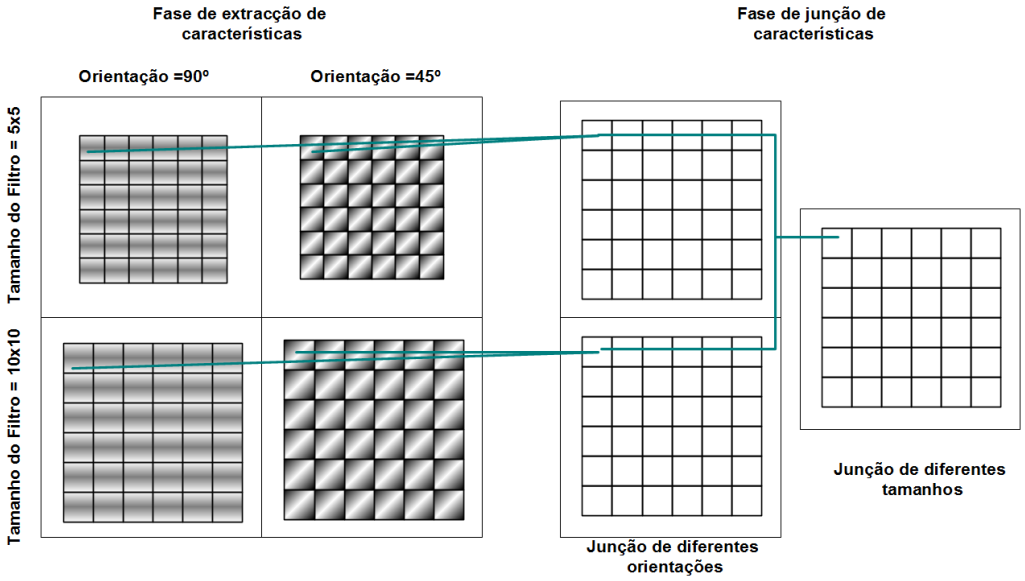


Figura 4.7: *Exemplo do processo de junção de camadas.*

4.3 Fase de análise de características

Para esta fase utilizámos uma rede FF que utiliza o mapa de características resultante da camada anterior como entradas da rede. Esta rede FF pode ter número variável de camadas escondidas sendo também variável o número de neurónios de cada uma, contudo a camada final deve conter um número de neurónios indeficiente para dar resposta ao problema que esta tenta resolver.

Capítulo 5

Resultados Obtidos

Após a implementação dos algoritmos necessários para a construção e aprendizagem não só das redes neuronais FF mas também das redes neuronais de convolução realizámos diversos testes explorando a sua capacidade de resolver problemas e a influência de vários parâmetros de configuração da rede.

5.1 Redes FF

Tal como explicado anteriormente as redes neuronais FF podem ter diferentes arquitecturas, taxas de aprendizagem, funções de activação, entre outros. Deste modo utilizámos a rede FF isoladamente das restantes fases convolucionais para testar a influência dos diversos parâmetros na aprendizagem de um problema comum.

O problema proposto foi determinar qual o maior número entre três números. Para este problema foram gerados aleatoriamente trios de números normalizados entre 0 e 1, existindo para cada trio a resposta correcta tanto para o conjunto de treino, dado que a aprendizagem é supervisionada como para o conjunto de testes, para avaliar as respostas da rede com casos novos. Dado o problema em mãos em que tivemos que analisar 3 números necessitámos de uma camada de entrada com 3 neurónios e uma camada final com o mesmo tamanho para que cada um dos neurónios

finais devolvesse 0 excepto aquele que correspondesse ao neurónio da entrada com o maior número devendo esse devolver 1. Para a entrada [0.8 0.63 0.1] a saída esperada seria [1 0 0].

Como vemos, na arquitectura da rede, podemos apenas alterar o número de camadas intermédias e o número de neurónios em cada camada.

Com os testes seguintes pretendemos avaliar a influência de cada parâmetro na evolução da taxa de erro efectivo e Erro Médio Quadrado (MSE) em que o primeiro traduz a quantidade de vezes que cada neurónio da camada final acerta na resposta esperada enquanto que o erro MSE é calculado por:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(d\_i - y\_i)^2, \tag{5.1}$$

sendo N o número de neurónios na camada final, d_i a saída esperada para o neurónio n_i e y_i a sua verdadeira saída.

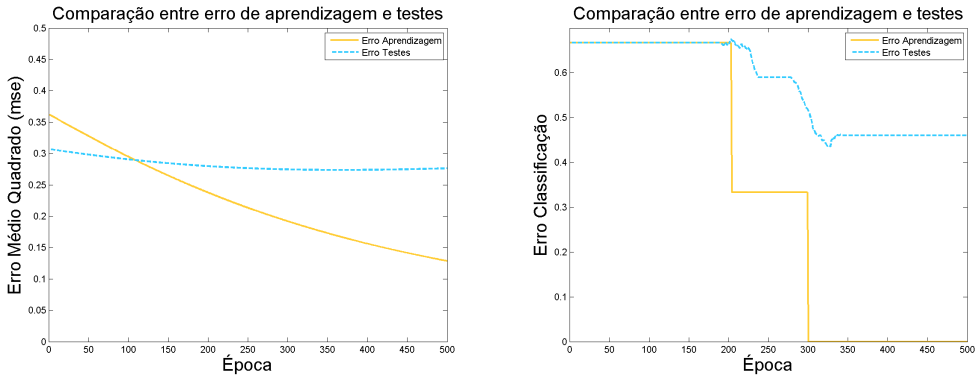
Para testar a influência do número de casos de treino no desempenho da rede realizámos testes com a rede construída variando apenas o número de casos de treino apresentados à rede para posteriormente a testar utilizando 200 casos diferentes dos de treino. Nestes testes utilizámos as configurações apresentadas a seguir.

5.1.1 Resultados para diferentes quantidades de exemplos de treino

Tabela 5.1: *Configuração da rede*

Nr. de Neurónios na Camada Intermédia	2
Taxa de aprendizagem	0.01
Função de activação	Sigmóide
Nr. de casos de treino	Variável
Nr. de épocas	Max(10*Nr de casos de treino,500)
Nr. de casos de teste	200

Dado que o número de casos de treino varia foi necessário treinar a rede com um número de épocas ajustável à quantidade de informação que esta tem de processar pelo que o acordámos que o número de épocas seria 10 vezes maior que o número de casos fornecidos à rede, contudo definimos um mínimo de 500 épocas para os casos em que o número de casos de treino é muito baixo.

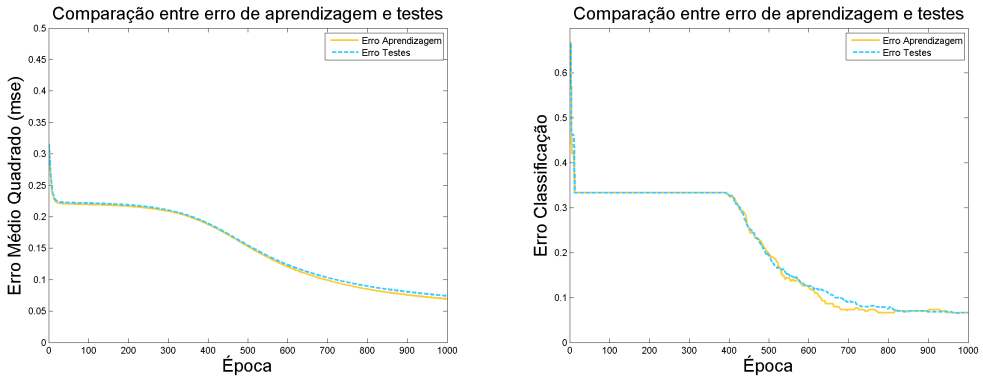


(a) (b)

Figura 5.1: *Resultados para 1 caso de treino.*

Como podemos observar pelo gráfico 5.1(a) que ilustra o valor do MSE a rede adapta-se cada vez mais ao único caso de treino, levando a que à medida que isso acontece perde a capacidade de generalizar em relação ao conjunto dos 200 testes.

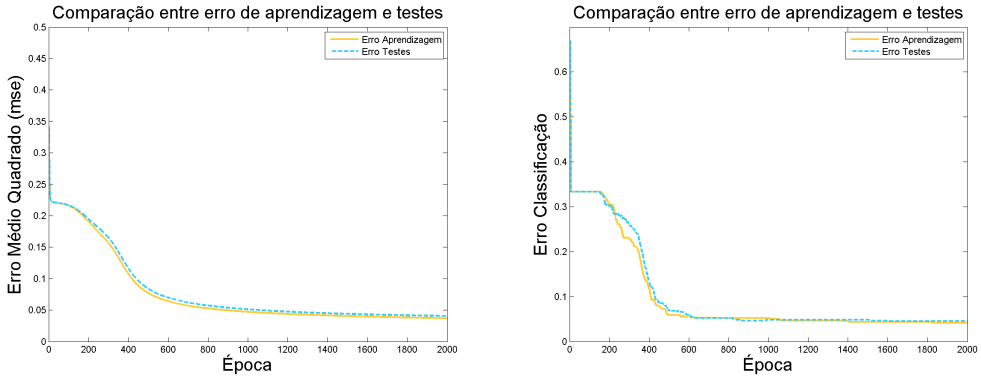
No gráfico 5.1(b) vemos que o erro de aprendizagem passa de 0.66 (66%) para 0.33 (33%) até chegar a 0. Isto deve-se ao facto de o erro efectivo ser calculado não pelo resposta global da rede mas sim por cada resposta correcta dos neurónios da camada final, ou seja, caso a saída esperada fosse {0 0 1} e a resposta fosse 0 1 0 apenas a primeira a saída corresponde à esperada sendo o erro de 2/3 = 0.66 que passaria a 0.33 se duas das saídas estivessem correctas e caso acertasse todas o erro seria 0.



(a) (b)

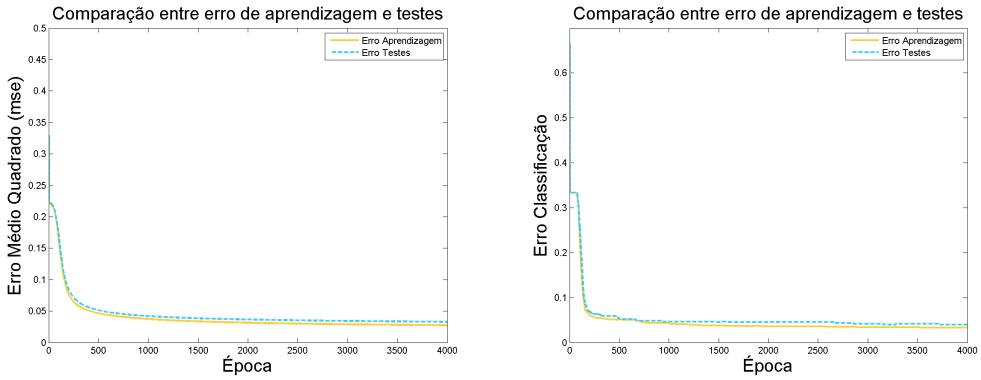
Figura 5.2: *Resultados para 100 casos de treino.*

Em 5.2(a) e 5.2(b) vemos que ao aumentarmos o número de exemplos de treino permitimos à rede aprender o problema desejado optimizando a cada época que passa os seus pesos de tal modo que cada vez que é testada com os 200 casos que nunca tinha visto esta consegue acertar cada vez mais levando o erro de testes a baixar à medida que o erro de aprendizagem baixa também. Observamos também que o erro de teste é sempre um pouco superior ao erro de aprendizagem como seria esperado, dado que a rede tenta optimizar os seus pesos em função do erro obtido pelos casos de treino, o que leva a que esta esteja melhor adaptada aos casos que treinou do que aos que nunca viu.



(a) (b)

Figura 5.3: *Resultados para 200 casos de treino.*



(a) (b)

Figura 5.4: *Resultados para 400 casos de treino.*

Apesar de aumentarmos os exemplos de treino para 200 e 400 verificamos que a rede consegue adaptar-se mais rápido ao problema ,como podemos ver pelo momento em que o erro efectivo "caí"dos 0.33, época 400 quando utilizámos 100 casos de treino, gráfico 5.2(b), e época 200 para 200 e 400 casos, gráficos 5.3(b) e 5.4(b). Isto pode ser explicado pelo facto de 100 casos não serem suficientes para "esclarecer"totalmente a rede do problema em questão levando a que esta leve mais tempo para aprender o mesmo que aquela que é treinada com mais casos.

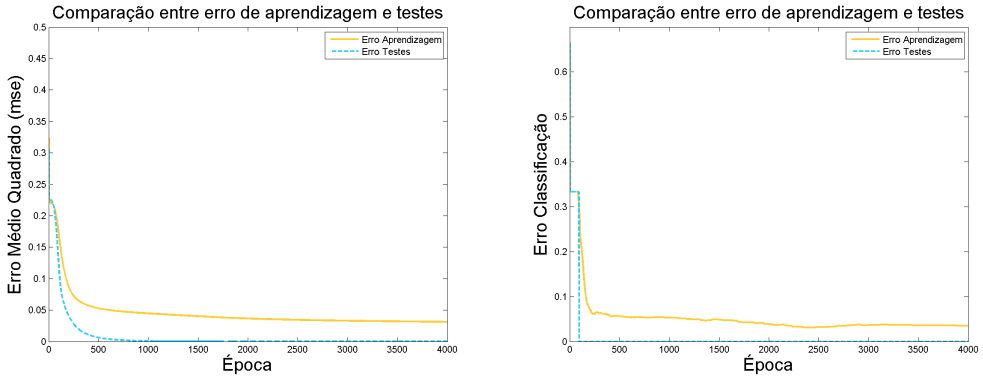
Ao aumentar o número de casos de treino para 1000, 2000, 5000 e 10000

não notámos mais alterações importantes pelo que não colocámos aqui os seus resultados.

5.1.2 Resultados para diferentes quantidades de exemplos de teste

Tabela 5.2: *Configuração da rede*

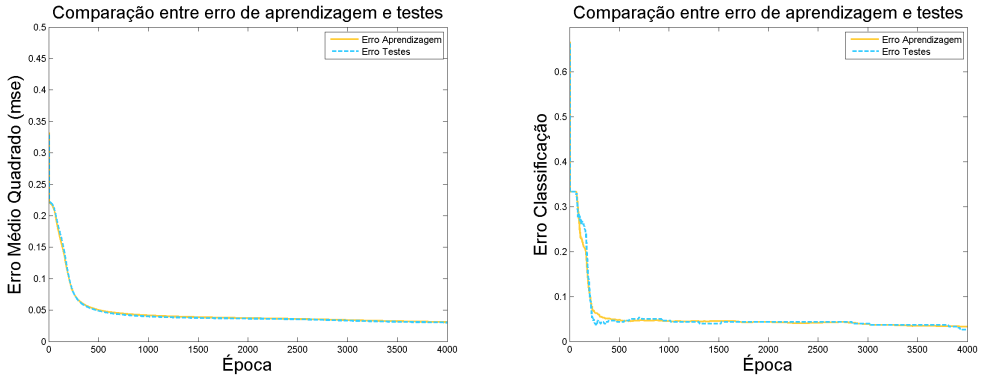
Nr. de Neurónios na Camada Intermédia	2
Taxa de aprendizagem	0.01
Função de activação	Sigmóide
Nr. de casos de treino	400
Nr. de épocas	4000
Nr. de casos de teste	Variável



(a) (b)

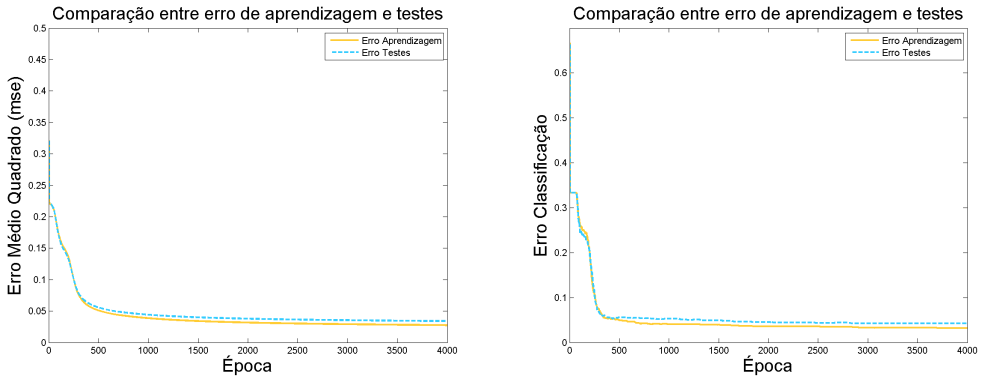
Figura 5.5: *Resultados para 1 caso de teste.*

Por oposição à situação expressa pela gráficos 5.1(a) e 5.1(b) em que a rede tinha apenas um exemplo para treinar e não conseguia encontrar os pesos ideais para generalizar o problema, neste caso vemos que com vários casos de treino a rede consegue facilmente acertar no único caso de teste que lhe é posto sem erros de classificação, gráfico 5.5(a), e com um erro MSE insignificante, gráfico 5.5(b).



(a) (b)

Figura 5.6: *Resultados para 100 caso de teste.*



(a) (b)

Figura 5.7: *Resultados para 500 caso de teste.*

Pela comparação dos gráficos 5.6(a) com 5.7(a) e 5.6(b) com 5.7(b) verificamos que o erro de teste aumenta com o aumento do número de casos de teste pois vemos que em 5.6(a) e 5.6(b) as linhas de treino e teste quase se confundem enquanto que em 5.7(a) e 5.7(b) o erro de teste é sempre superior ao erro de treino. No caso em que utilizamos 100 casos de teste estamos a utilizar $\frac{1}{4}$ do número de casos de treino enquanto que com 500 casos de teste utilizamos mais do que o número de casos de treino o que leva a rede ter uma maior dificuldade em acertar tão bem como no caso em que temos 100 casos pois existe uma maior probabilidade de existirem ca-

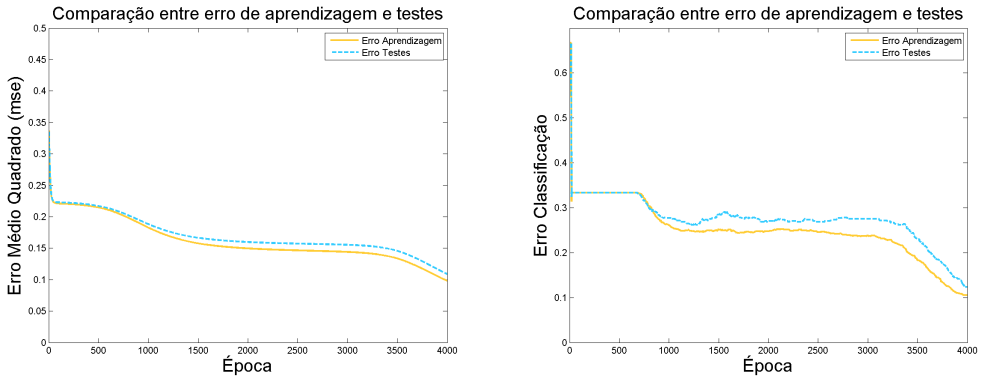
sos que fujam à normalidade da maioria dos casos apresentados no treino. Para conseguir bons resultados com um conjunto de teste com um número elevado de casos é necessário fornecer à rede um número apropriado de casos de treino.

5.1.3 Resultados para diferentes taxas de aprendizagem

Tabela 5.3: *Configuração da rede*

Nr. de Neurónios na Camada Intermédia	2
Taxa de aprendizagem	Variável
Função de activação	Sigmóide
Nr. de casos de treino	400
Nr. de épocas	4000
Nr. de casos de teste	200

Outro factor importante no desempenho no treino e consequente desempenho da rede é a taxa de aprendizagem que deve variar entre 0 e 1, cuja influência vamos também testar. Como já foi explicado anteriormente, a taxa de aprendizagem influência a velocidade com que a rede actualiza os seus pesos, ou seja, o número de épocas que são necessárias para que a rede atinja valores de erro aceitáveis para que se possa dizer que o problema foi aprendido.

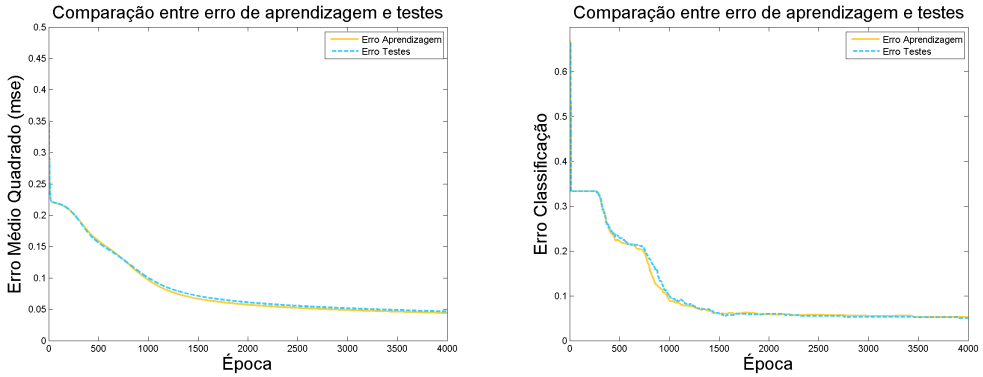


(a) (b)

Figura 5.8: *Resultados para uma taxa de aprendizagem de 0.001 .*

Comparando o resultados da evolução do erro MSE em 5.8(a) e o erro de classificação em 5.8(b) com os gráficos 5.4(a) e 5.4(b) em que foram utilizados os mesmos parâmetros de configuração da rede, excepto a taxa de aprendizagem que era 10 vezes maior, podemos verificar que a rede

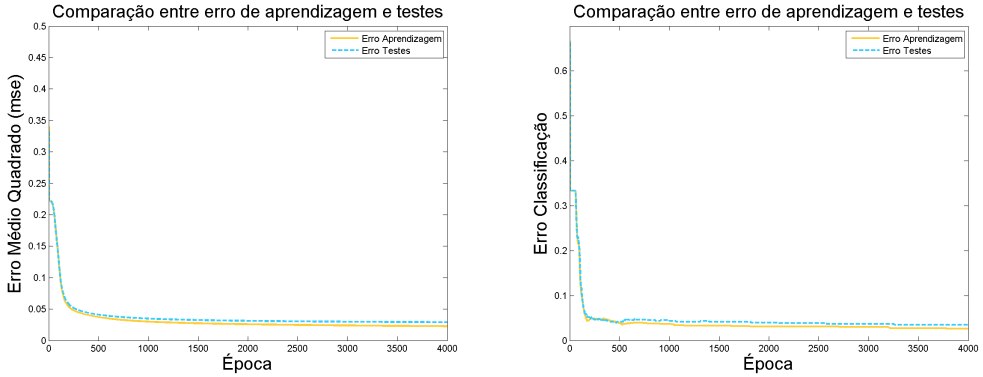
que usa a taxa de aprendizagem de 0.001 demora mais épocas para atingir os mesmos valores de erro que a rede com a taxa de aprendizagem 0.01.



(a) (b)

Figura 5.9: *Resultados para uma taxa de aprendizagem de 0.002 .*

Ao aumentar a taxa de aprendizagem para o dobro em relação à situação apresentada em 5.8 verificamos que após as mesmas épocas a rede consegue alcançar um valor de MSE inferior a 0.05, gráfico 5.9(a), enquanto que com a taxa de 0.001 apenas atinge um MSE de 0.1, gráfico 5.8(a), contudo não ultrapassa o desempenho da rede com taxa de 0.01, gráfico 5.4(a).

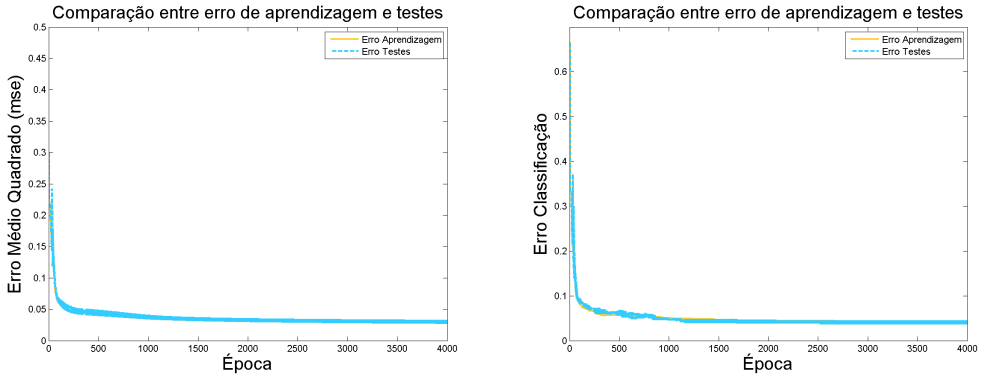


(a) (b)

Figura 5.10: *Resultados para uma taxa de aprendizagem de 0.02 .*

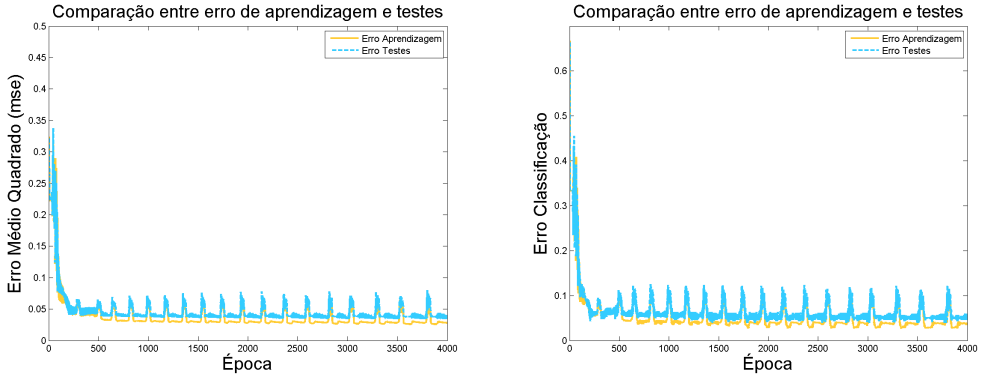
Comparando os resultados obtidos em 5.10(a) e 5.10(b) com os gráficos

5.4(a) e 5.4(b), cuja taxa de aprendizagem é metade, vemos que já não conseguimos um melhor desempenho ao fim de 4000 épocas.



(a) (b)

Figura 5.11: *Resultados para uma taxa de aprendizagem de 0.05 .*



(a) (b)

Figura 5.12: *Resultados para uma taxa de aprendizagem de 0.1 .*

Com taxas de 0.05 e 0.1 começamos a notar alguma instabilidade no erro como podemos verificar pelo gráficos 5.11(a) e 5.12(a) em que a alta variação do erro entre cada época faz com que a curva de erro tenha mais altos e baixos do que as redes com taxas mais baixas.

Isto pode ser explicado devido ao facto de a actualização de cada peso ser directamente proporcional à taxa de aprendizagem, de modo que a cada

época os pesos variam mais do que com taxas mais pequenas, levando a que cada peso se afaste mais do valor anterior originando erros com maiores diferenças em relação aos anteriores.

5.1.4 Resultados com diferente número de neurónios na camada escondida

Tabela 5.4: *Configuração da rede*

Nr. de Neurónios na Camada Intermédia	Variável
Taxa de aprendizagem	0.01
Função de activação	Sigmóide
Nr. de casos de treino	400
Nr. de épocas	4000
Nr. de casos de teste	200

O número de neurónios que possui a camada escondida é bastante importante para o desempenho da rede, dado que são estes que possibilitam a aproximação à função óptima de resolução do problema e caso existam em número insuficiente a rede pode não conseguir aprender.

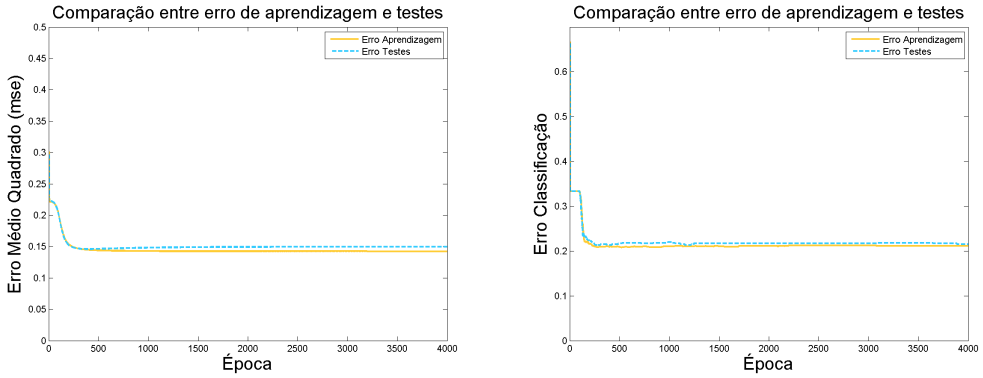
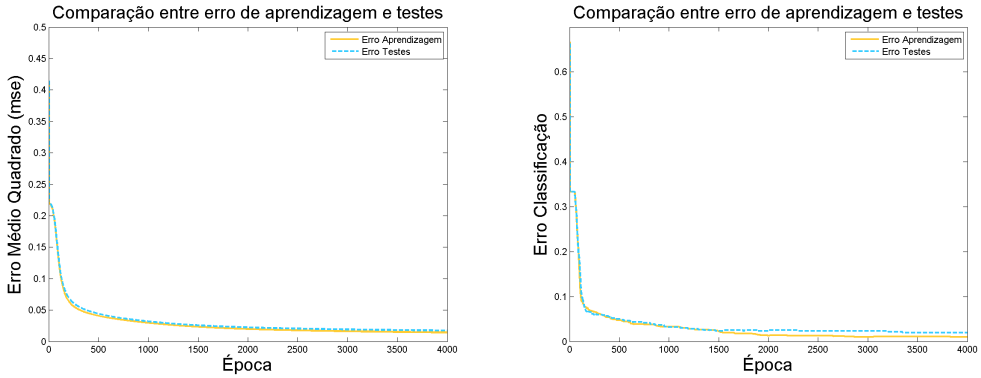


Figura 5.13: *Resultados com 1 neurónio na camada escondida .*

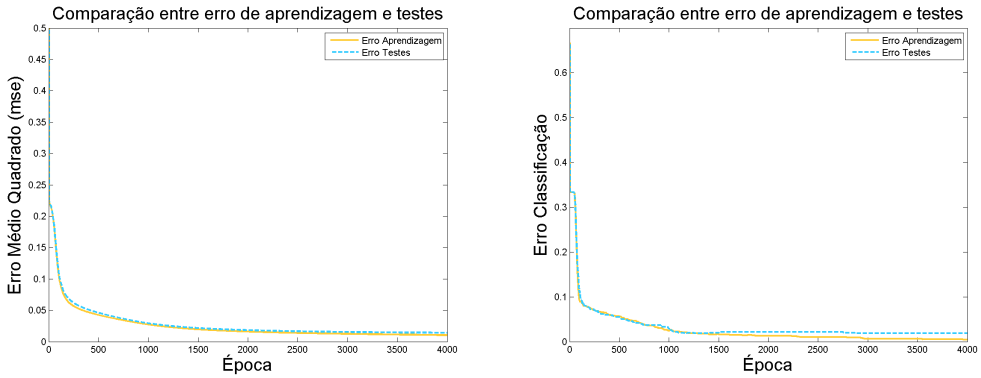
O desempenho da rede com 1 neurónio na camada intermédia, 5.13, em comparação com a rede com 2 neurónios ilustrada na figura 5.4 é inferior, dado que ao fim de as mesmas épocas o erro é superior.



(a) (b)

Figura 5.14: *Resultados com 4 neurónio na camada escondida .*

À medida que aumentamos o número de neurónios na camada escondida conseguimos alcançar um erro menor com vemos pela evolução da taxa de erro utilizando 1, 2 e 4 neurónios como podemos verificar pelo gráficos do erro MSE e erro efectivo 5.13, 5.4, 5.14 respectivamente.

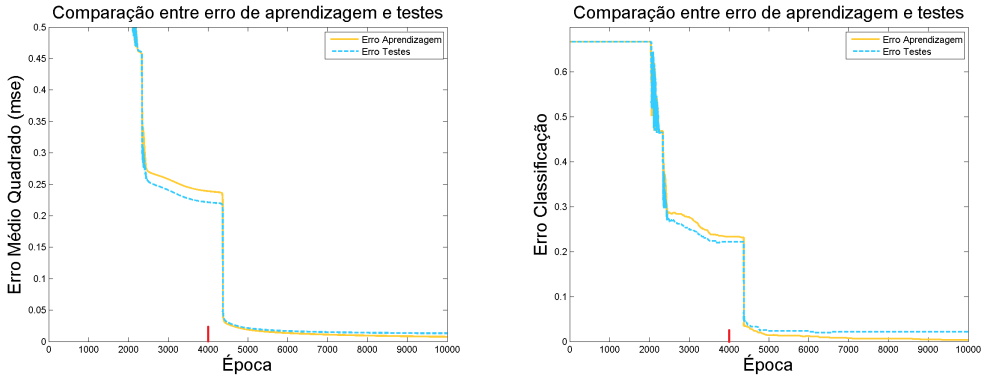


(a) (b)

Figura 5.15: *Resultados com 10 neurónio na camada escondida .*

Podemos verificar que o aumento de 4 para 10 já não melhora o desempenho da rede como vemos pelos gráficos 5.14(a) e 5.15(a) em que o valores do erro MSE não diferem muito ao fim de 4000 épocas.

O aumento excessivo da quantidade de neurónios pode levar a aprendi-



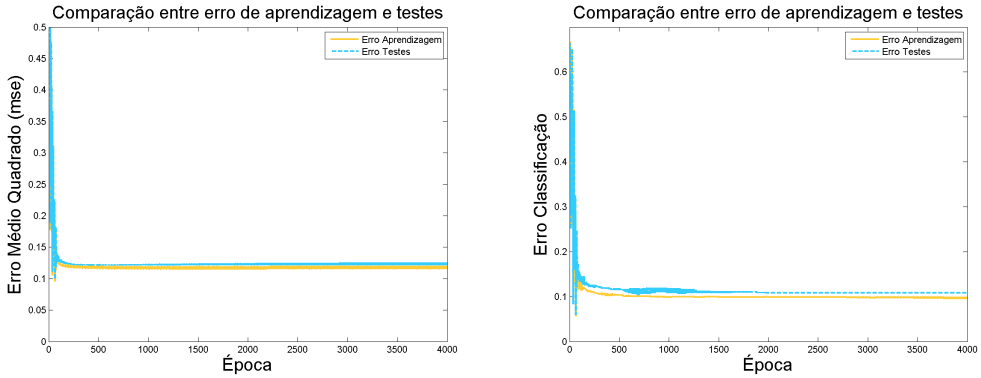
(a) (b)

Figura 5.16: *Resultados com 35 neurónio na camada escondida .*

zagem a demorar muito mais para atingir taxas de erro aceitáveis. Devido a este facto 4000 épocas não seriam suficientes para que o erro MSE chegasse a valores abaixo dos 0.05 pelo que alargámos o número de épocas excepcionalmente para 10000 para poder verificar que realmente o erro após 4000 épocas permanece demasiado alto em relação a redes com menos neurónios contudo este acaba por atingir valores aceitáveis algumas épocas depois, como vemos pelos gráficos 5.16(a) e 5.16(b).

5.1.5 Resultados para diferentes funções de activação

Pela comparação entre os gráficos 5.17(a) e 5.17(b) que usam a tangente hiperbólica como função de activação e os gráficos 5.4(a) e 5.4(b) que usam as mesmas configurações excepto o facto de utilizarem a função sigmóide podemos dizer que a função sigmóide adapta-se melhor a este tipo de problema e parâmetros.



(a) (b)

Figura 5.17: *Resultados para a tangente hiperbólica como função de activação.*

5.2 Redes Convolucionais

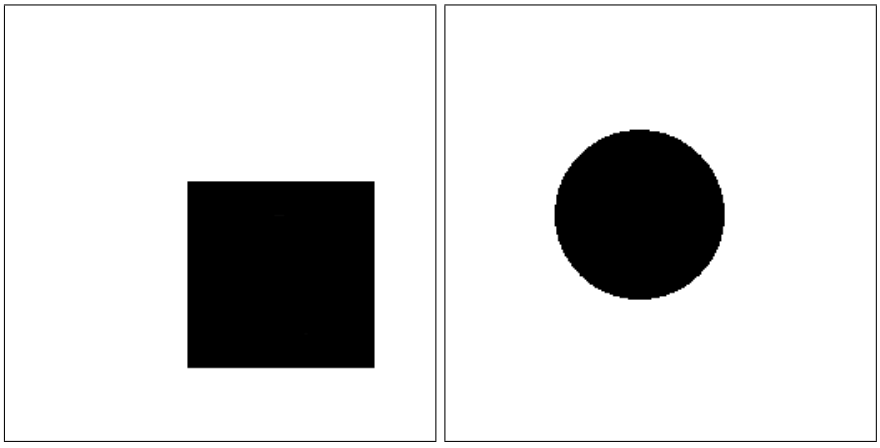
5.2.1 Problema 1

Neste novo problema o nosso objectivo é tentar que a rede consiga diferenciar entre duas imagens que contêm ou não o mesmo objecto, mais concretamente usámos um quadrado e um círculo tal como na figura 5.18(a) e 5.18(b).

As figuras usadas tinham uma dimensão de 20 x 20 pixels. Estas são introduzidas na rede para que a rede devolva [1 0] no caso de as imagens serem do mesmo objecto e [0 1] caso contrário. Cada imagem terá um quadrado ou círculo sendo o tamanho desse objecto aleatório mas nunca excedendo os 20 pixels que corresponde ao tamanho da imagem.

Para este problema uma simples rede FF não é suficiente como já foi explicado, pelo que é com este tipo de problemas que faz sentido utilizar uma rede neuronal convolucional que retira algumas características das imagens de entrada para introduzi-las como entradas de uma rede FF.

Neste tipo de rede temos mais parâmetros de configuração do que os já explicitados nos testes anteriores 5.1 pelo que vamos agora incidir os nossos testes no estudo da influência de novos parâmetros desta rede já que os restantes parâmetros já foram testados anteriormente.



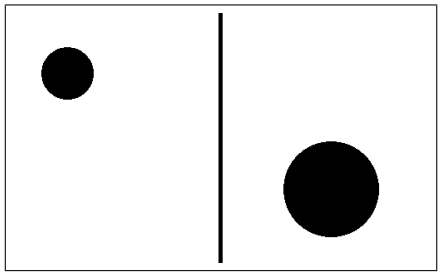
(a) (b)

Figura 5.18: *Imagens usadas para o treino e teste da rede.*

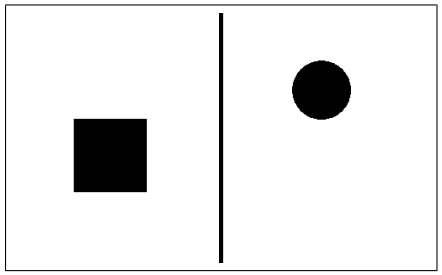
Na tabela 5.5 apresentamos as configurações utilizadas em todos os testes deste problema. Estes parâmetros são comuns aos diversos testes realizando, exceptuando claro o parâmetro que cada teste varia de modo a averiguar a sua influência no desempenho da rede.

Tabela 5.5: *Configuração da rede*

Nr. de Escalas	1
Escalas	$\{5x5\}$
Nr. de Orientações	1
Orientações	$\{0\pi\}$
Shift	2
Nr. de Neurónios nas Camadas Intermédias	$\{8, 4\}$
Taxa de aprendizagem	0.005
Função de activação	Sigmóide
Nr. de casos de treino	5000
Nr. de casos de teste	5000
Nr. de épocas	3000

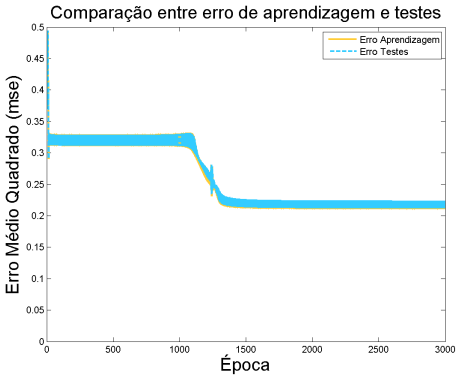


(a) Exemplo de um par do conjunto de treino com o mesmo objecto.

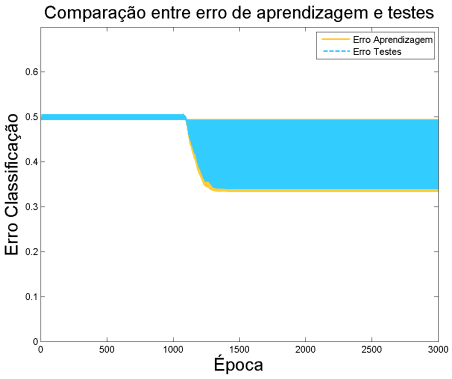


(b) Exemplo de um par do conjunto de treino com diferentes objectos.

Figura 5.19: *Imagens utilizadas no conjunto de treino e teste.*



(a)

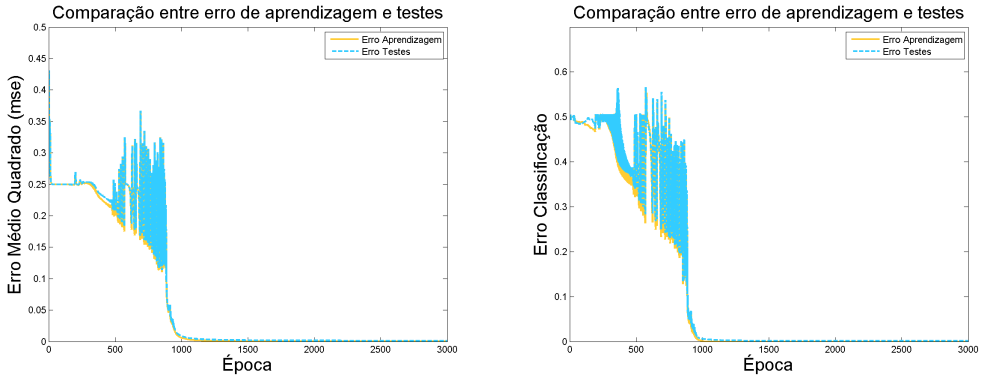


(b)

Figura 5.20: *Resultados para* $\eta = 0.01$.

5.2.2 Resultados com taxas de aprendizagem diferentes

Podemos verificar que para este problema conseguimos atingir um melhor desempenho utilizando uma taxa de aprendizagem mais baixa 5.21 do que a comum 0.01 5.20. Este resultado contrariou as nossas expectativas dado que como vimos em 5.1.3 em que uma taxa maior levaria a um melhor desempenho, contudo se observarmos também os resultados de 5.11(a) e 5.12(a) vemos que taxas demasiado elevadas podem ser prejudiciais para o treino da rede, pelo que podemos concluir que neste tipo de problema e devido também ao tamanho das imagens ser reduzido è necessário actualizar os pesos da rede de forma bastante suave, sendo $\eta = 0.01$ demasiado

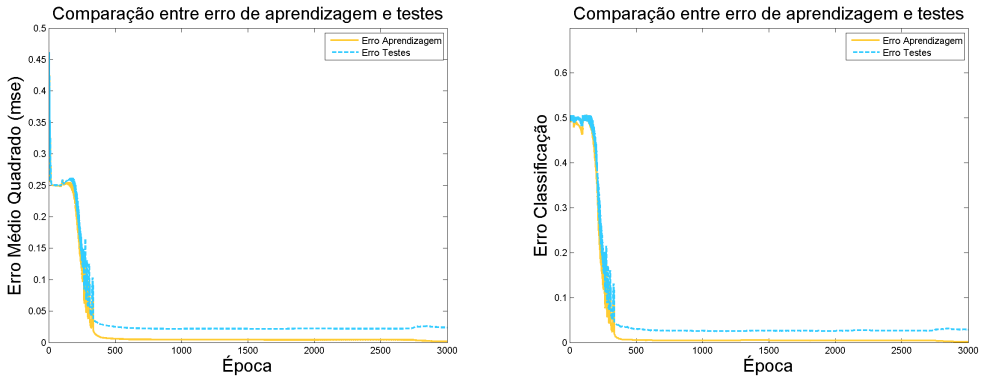


(a) (b)

Figura 5.21: *Resultados para* $\eta = 0.005$.

elevada para este problema.

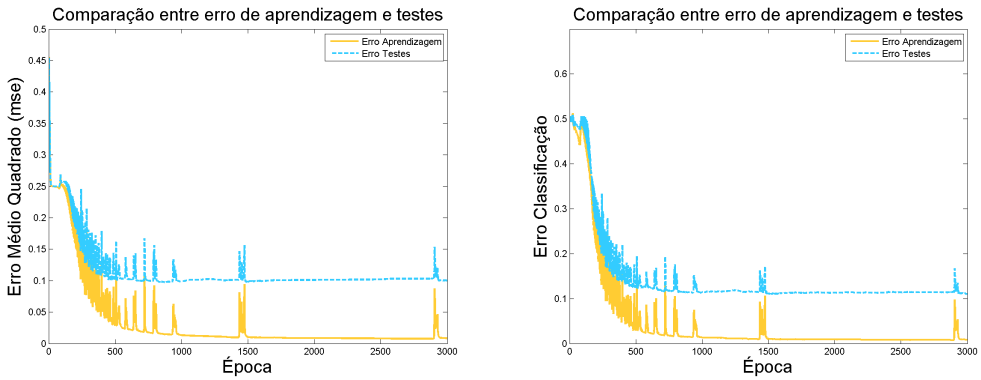
5.2.3 Resultados para diferentes números de escalas e orientações



(a) (b)

Figura 5.22: *Resultados para Nr. Escalas=2 e Nr. Orientações=2.*

No primeiro teste 5.21 os filtros utilizados tinham dimensão 5x5 e orientação 0π, para o segundo teste 5.22 adicionámos a dimensão 10x10 e a



(a) (b)

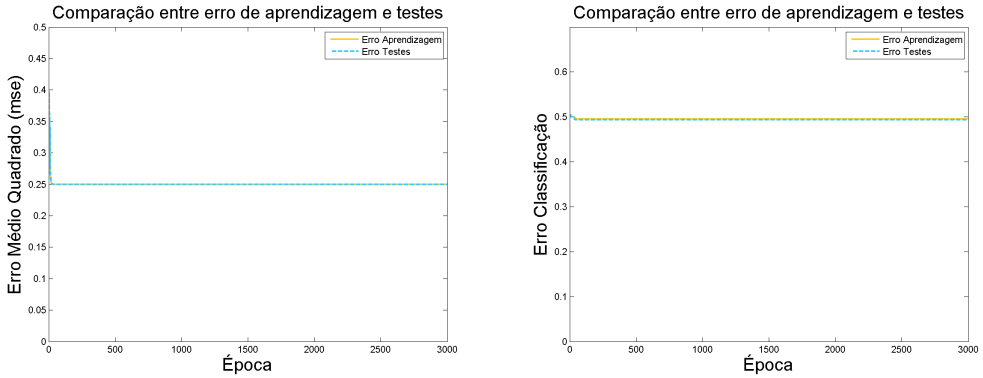
Figura 5.23: *Resultados para Nr. Escalas=3 e Nr. Orientações=3.*

orientação $\frac{\pi}{2}$, e no terceiro teste 5.23 juntámos mais a dimensão $15x15$ e a orientação $\frac{\pi}{4}$. Como podemos verificar pelos gráficos 5.21(a),5.22(a),5.23(a) o valor do erro MSE aumenta à medida que introduzimos na rede mais camadas com diferentes orientações. Este resultado contrariou as nossas expectativas, dado que esperávamos que ao aumentar o número de características retiradas da imagem conseguíssemos que a rede conseguisse fazer uma melhor distinção entre diferentes objectos, contudo vemos que acontece o inverso. Por este resultado concluímos que para este problema o uso de vários tipos de filtros com diferentes tamanhos e orientações é prejudicial para o desempenho da rede.

5.2.4 Resultados com filtros de tamanho diferente

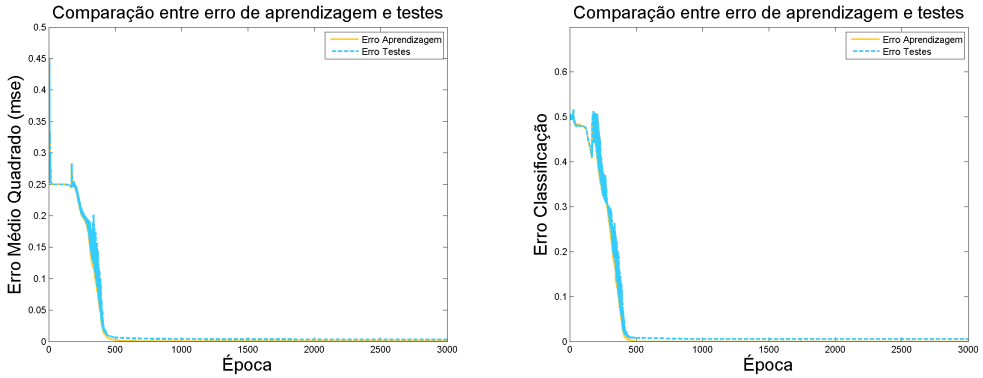
Verificamos que com filtros de $3x3$ a rede não tem nenhuma capacidade de aprendizagem 5.24(a) e 5.24(b), dado que este tamanho não permite que o filtro contenha as suas propriedades características e não seja possível extrair características convenientes.

Pelo contrário vemos que um filtro de $8x8$ é suficiente para que a rede possa aprender o problema 5.25(a), tal como um filtro de $5x5$ 5.21 sendo a única diferença entre estes a instabilidade inicial que existe na evolução do valor do MSE 5.21(a) e do erro efectivo 5.21(b).



(a) (b)

Figura 5.24: *Resultados para filtros com tamanho 3x3.*

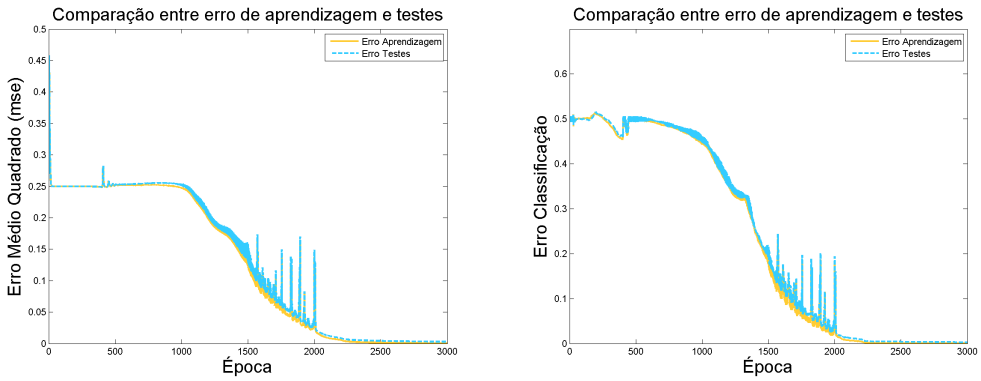


(a) (b)

Figura 5.25: *Resultados para filtros com tamanho 8x8.*

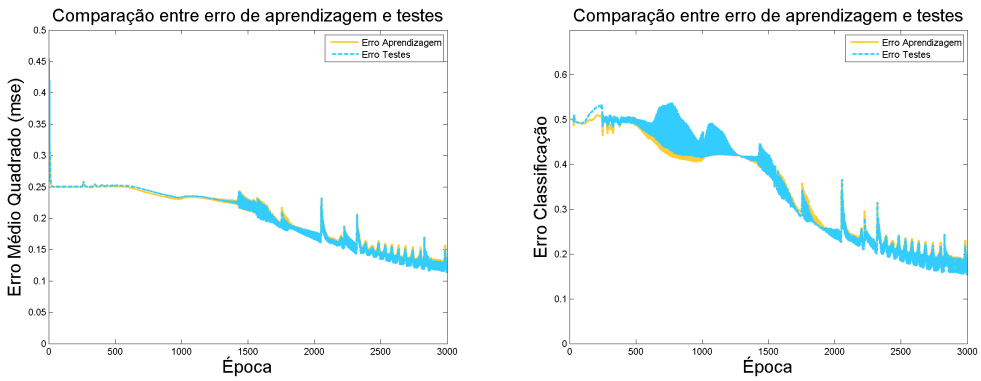
5.2.5 Resultados com valores de *Shift* diferentes

Ao aumentar o valor do *Shift* de 2 para 3 não notamos diferenças significativas contudo o desempenho da rede piora para *Shift* = 5 como podemos observar tanto para o valor de MSE 5.21(a) 5.27(a) como o erro efectivo 5.21(b) 5.27(b) piora. Este resultado deve-se ao facto do aumento do valor do *Shift* implicar uma menor quantidade de características extraídas.



(a) (b)

Figura 5.26: *Resultados para Shift = 3.*



(a) (b)

Figura 5.27: *Resultados para Shift = 5.*

5.2.6 Intervalo de confiança para os resultados da rede

De modo a garantir que os resultados anteriores não divergem muito da média procedemos à construção de um diagrama de extremos e quartis. Optámos por fazer este diagrama apenas para o resultado para uma das experiências deste problema sendo a situação 5.21 a escolhida, dado que foi aquela cujas configurações permitiu um melhor desempenho à rede. Desta forma realizámos a experiência 5.21 10 vezes e em 5.28 podemos observar que o nosso intervalo de confiança situa-se entre 2% e 2.25% para a aprendizagem e cerca de 3% para os testes.

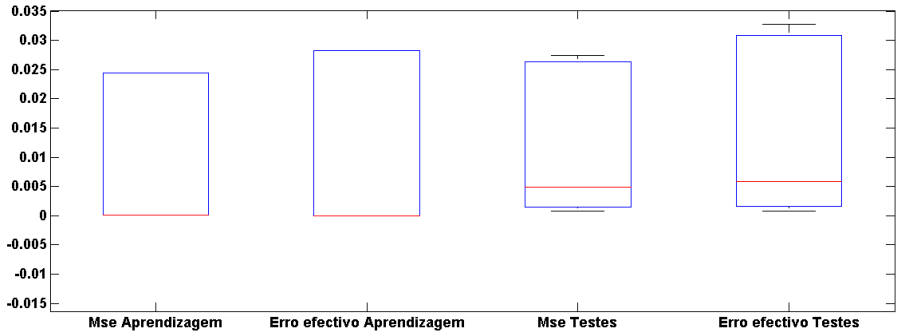


Figura 5.28: *Intervalos de confiança para os erros do treino e teste.*

5.2.7 Problema 2

Para este segundo problema propusemos uma tarefa mais árdua à rede, queremos que esta seja capaz de reconhecer se determinada imagem contêm ou não um padrão num conjunto de vários padrões misturados. Para a realização do conjunto de treino e teste escolhemos 9 imagens de personagens de desenhos animados conhecidas ilustradas na figura 5.29.

Figura 5.29: *Imagens do conjunto de treino e teste.*

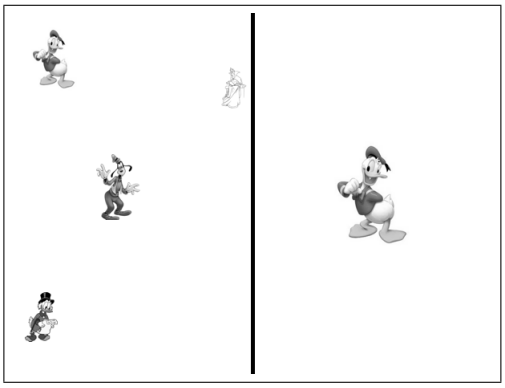
Com estas 9 imagens vamos construir pares de imagens da seguinte forma:

1. Para construir a primeira imagem
   (a) Escolher n imagens aleatoriamente do conjunto de imagens, com $1 \leq n \leq 9$
   (b) Redimensionar as n imagens com uma escala p escolhida aleatoriamente, com $\frac{1}{4} \leq p \leq 1$
   (c) Colocar as n imagens reunidas numa só imagem de modo que não se sobreponham
2. Para construir a segunda imagem

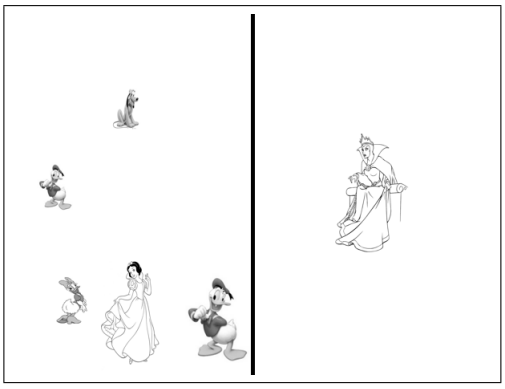
(a) Escolher 1 imagens aleatoriamente do conjunto de imagens

(b) Colocar a imagem escolhida centrada numa nova imagem com o mesmo tamanho que a primeira imagem construída

Após a construção do par de imagens verificamos se a imagem escolhida na segunda imagem está contida nas figuras da primeira imagem, de tal modo que se estiver a saída esperada para a rede deve ser [1 0] caso contrário [0 1]. Desta forma, a rede tem como objectivo responder se o padrão que encontra na segunda imagem se encontra na primeira com a dificuldade acrescida de na primeira imagem estas estarem a uma escala diferente.



(a)



(b)

Figura 5.30: *Imagens do conjunto de treino e teste.*

A figura 5.30(a) pretende exemplificar o caso em que o padrão da segunda imagem se encontra na primeira enquanto que a figura 5.30(b) exemplifica

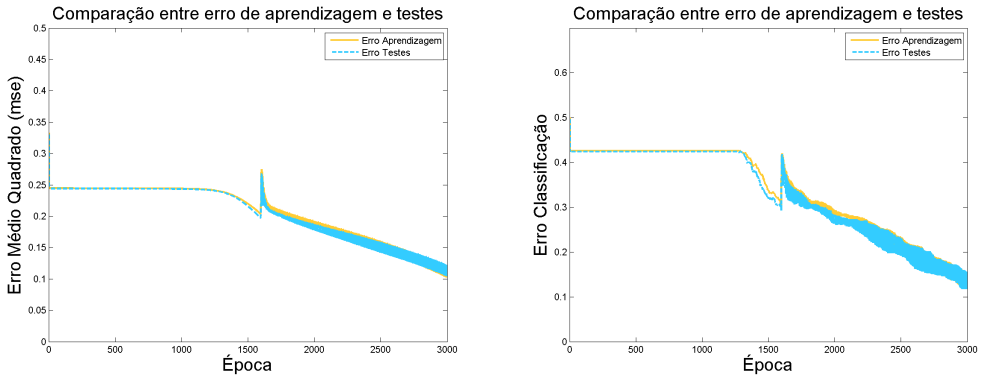
o oposto. As imagens que foram utilizadas têm uma dimensão $200x150$.

Na tabela 5.6 apresentamos as configurações utilizadas em todos os testes deste segundo problema. Tal como no problema 1 apenas varia o parâmetro que cada experiência pretende testar.

Tabela 5.6: *Configuração da rede*

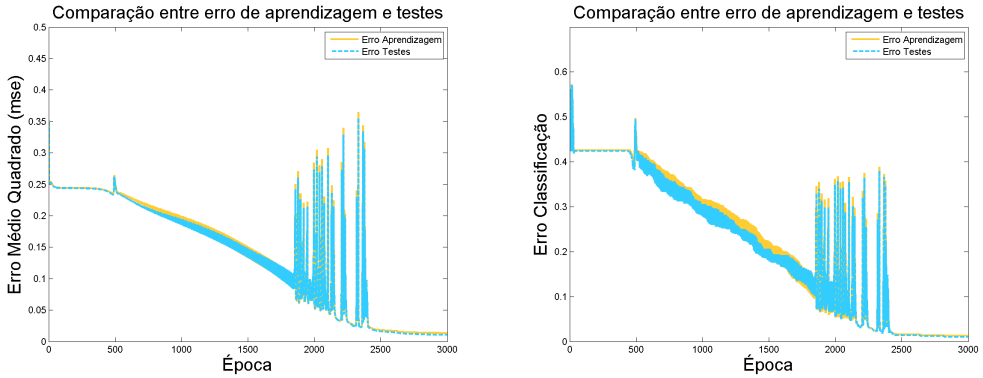
Nr. de Escalas	1
Escalas	$\{5x5\}$
Nr. de Orientações	1
Orientações	$\{0\pi\}$
Shift	5
Nr. de Neurónios nas Camadas Intermédias	$\{8, 4\}$
Taxa de aprendizagem	0.01
Função de activação	Sigmóide
Nr. de casos de treino	1000
Nr. de casos de teste	500
Nr. de épocas	3000

5.2.8 Resultados com taxas de aprendizagem diferentes



(a) (b)

Figura 5.31: *Resultados para* $\eta = 0.005$.



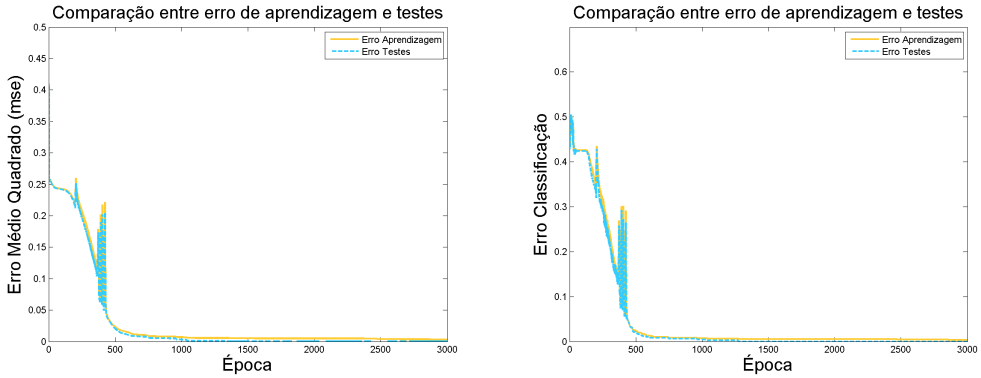
(a) (b)

Figura 5.32: *Resultados para* $\eta = 0.01$.

Após termos realizado os testes do problema anterior com uma taxa de 0.005 5.21 devido a esta melhorar o desempenho da rede encontramos neste segundo problema a situação oposta em que uma taxa de 0.01, gráficos 5.32(a) e 5.32(b),consegue superar o desempenho da rede treinada com $\eta = 0.005$, gráficos 5.31(a) e 5.31(b).

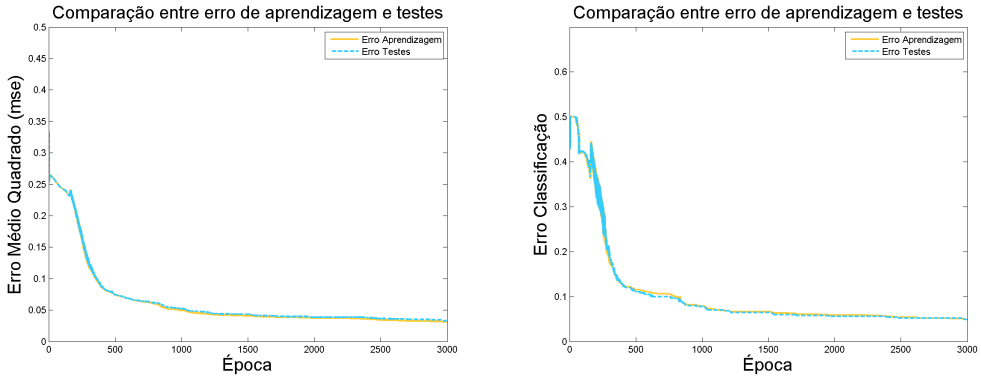
5.2.9 Resultados para diferentes números de escalas e orientações

Pela comparação do valor do MSE em 5.32(a), 5.33(a), 5.34(a), 5.35(a) e também pelo erro efectivo em 5.32(b) e 5.33(b), 5.34(b), 5.35(b) podemos concluir que para este novo problema a rede adapta-se melhor utilizando 4 tipo de filtros, ou seja ,com orientação de 0π e $\frac{\pi}{2}$ e escalas $5x5$ e $10x10$, dado que é para a situação 5.33 que conseguimos atingir valores menores de erro. Isto pode dever-se ao facto de estas imagens terem uma dimensão superior e existir uma maior entropia nestas,porque enquanto que no problema anterior as imagens apenas possuíam dois valores de intensidade, estas têm 255 valores possíveis para cada pixel.Deste modo, existe uma maior exigência na diferenciação de padrões neste problema levando que a que seja necessário retirar mais características. Contudo, parece existir um limite de características que ao ser ultrapassado apenas leva à "confusão"da rede, como podemos observar pelos resultados 5.34 e 5.35.



(a) (b)

Figura 5.33: *Resultados para Nr. Escalas=2 e Nr. Orientações=2.*

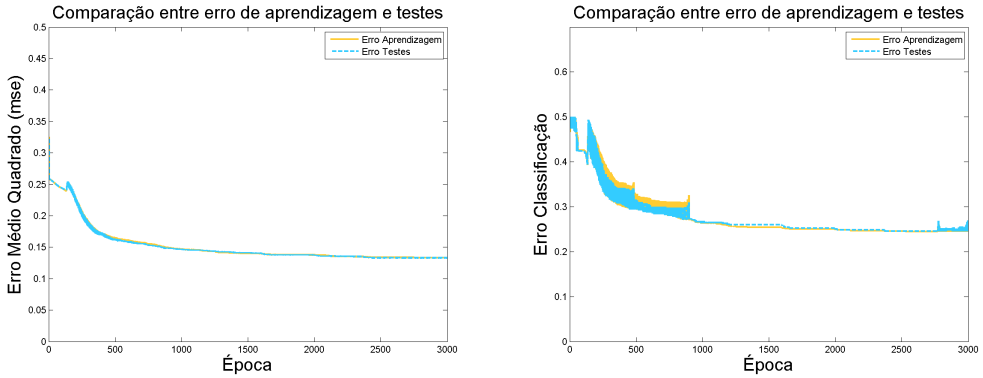


(a) (b)

Figura 5.34: *Resultados para Nr. Escalas=3 e Nr. Orientações=3.*

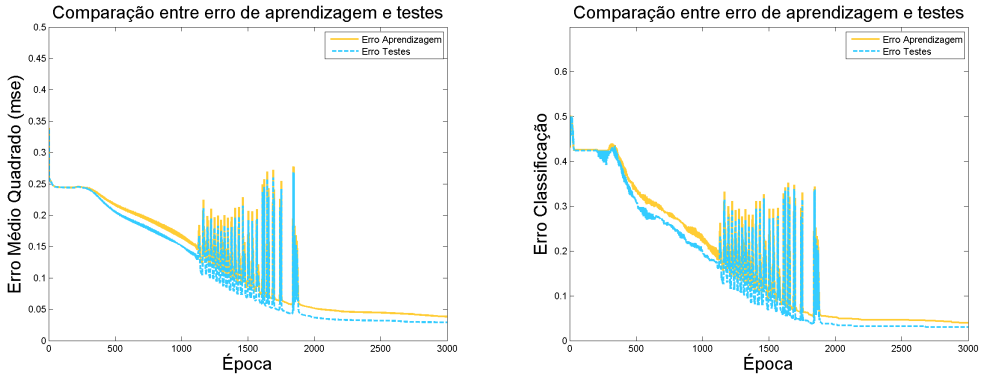
5.2.10 Resultados com valores de *Shift* diferentes

Podemos observar que tal como no problema anterior o aumento do valor de *Shift* piora o desempenho da rede como vemos pela evolução do valor MSE e erro efectivo quando utilizamos um valor de *Shift* de 5 5.32(a), 5.32(b), 10 5.36(a) 5.36(b) e 20 5.37(a). Contudo verificamos que neste problema temos um bom desempenho para *Shift* = 5 enquanto que no problema anterior o valor 2 era ideal 5.21 e tínhamos um mau desempenho para o valor 5 5.27. Estes resultados devem-se ao facto de as imagens dos



(a) (b)

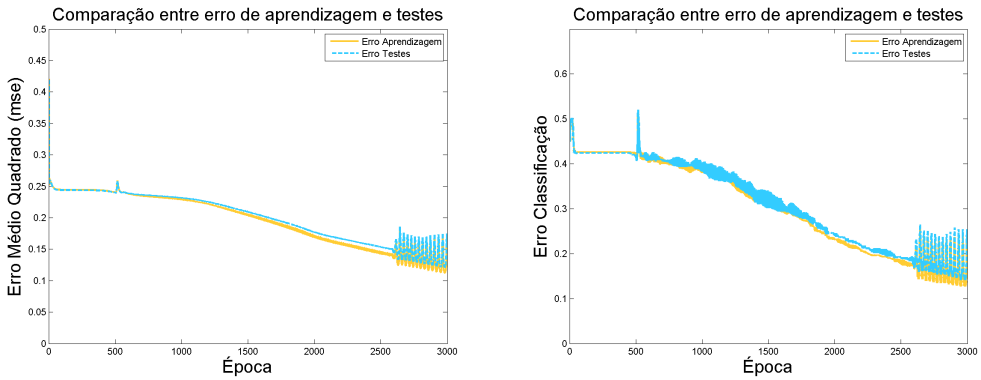
Figura 5.35: *Resultados para Nr. Escalas=4 e Nr. Orientações=4.*



(a) (b)

Figura 5.36: *Resultados para Shift = 10.*

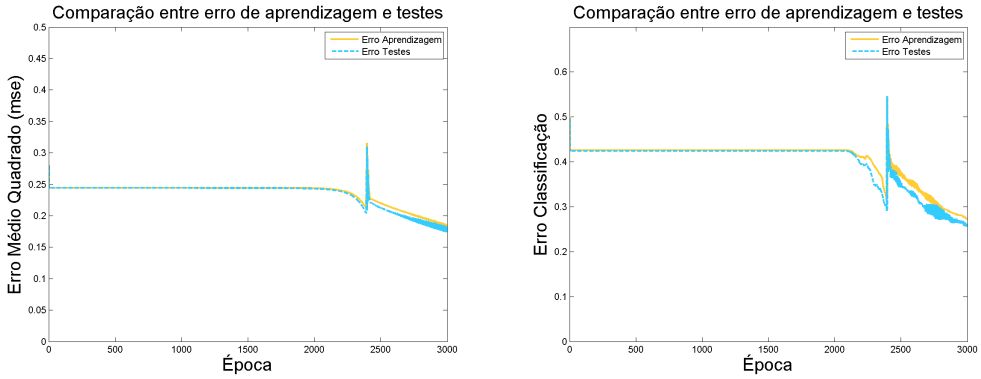
dois problemas terem dimensões diferentes o que leva a que nas imagens do problema 1 seja mais importante que o *Shift* seja mais pequeno pois estas têm apenas uma dimensão de 20x20. Ao utilizar um *Shift* de 5 nestas imagens apenas retiramos destas $\frac{20}{5}.\frac{20}{5} = 16$ características por camada enquanto que nas imagens do segundo problema,com dimensão de 200x150 conseguimos retirar $\frac{200}{5}.\frac{150}{5} = 1200$.



(a) (b)

Figura 5.37: *Resultados para Shift = 20.*

5.2.11 Resultados com diferente número de neurónios nas camadas escondidas



(a) (b)

Figura 5.38: *Resultados para duas camadas escondidas com arquitectura* $\{4, 2\}$.

Neste problema decidimos variar a configuração das camadas escondidas, tal como já tínhamos feito em 5.1.4 e tal como esperado a rede tem um pior desempenho ao diminuir o número de neurónios nestas camadas.

5.2.12 Intervalo de confiança para os resultados da rede

Tal como no problema 1 apresentamos um diagrama de extremos e quartis dos resultados da experiência com melhor desempenho 5.33 para o qual recolhemos 10 amostras. Em 5.39 podemos observar que o nosso intervalo de confiança é cerca de 0.3% para a aprendizagem e 0.4% para os testes.

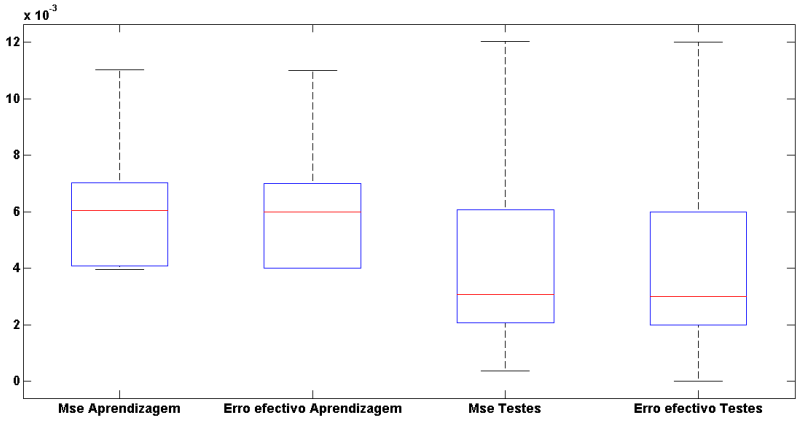
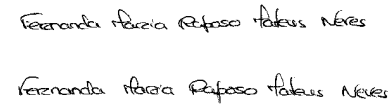


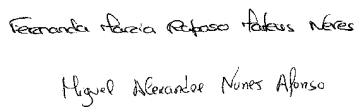
Figura 5.39: *Intervalos de confiança para os erros do treino e teste.*

5.2.13 Problema 3

Após a resolução de estes dois problemas pensámos em utilizar a nossa rede para um problema com utilidade que pudesse ser aplicado numa situação real do dia a dia. Deste modo pensamos em utilizar a nossa rede como um reconhecedor de assinaturas, sendo que desafiámos a rede a verificar entre duas assinaturas se estas pertencem à mesma pessoa ou não. Para a construção do conjunto de treino e teste recolhemos 20 assinaturas por pessoa que posteriormente foram digitalizadas com uma resolução de 1200*dpi* para seguidamente serem combinadas em pares de modo a gerar os dois tipos de resposta da rede. O caso 5.40(a) ilustra um par de assinaturas da mesma pessoa enquanto que o segundo caso 5.40(b) ilustra um par de assinaturas de pessoas diferentes.



(a) Exemplo de um par do conjunto de treino com o mesmo objecto.



(b) Exemplo de um par do conjunto de treino com diferentes objectos.

Figura 5.40: *Imagens utilizadas no conjunto de treino e teste.*

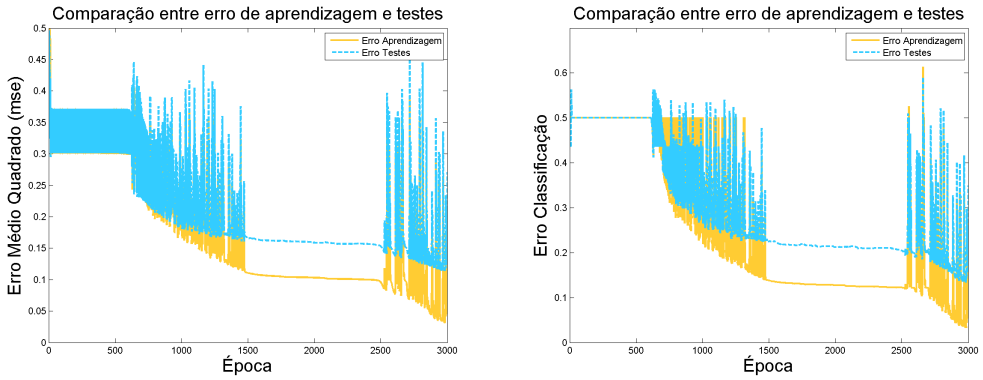
Tabela 5.7: *Configuração da rede*

Nr. de Escalas	1
Escalas	$\{5x5\}$
Nr. de Orientações	1
Orientações	$\{0\pi\}$
Shift	5
Nr. de Neurónios nas Camadas Intermédias	$\{8, 4\}$
Taxa de aprendizagem	0.01
Função de activação	Sigmóide
Nr. de casos de treino	3000
Nr. de casos de teste	1000
Nr. de épocas	3000

Na tabela 5.7 apresentamos as configurações utilizadas em todos os testes

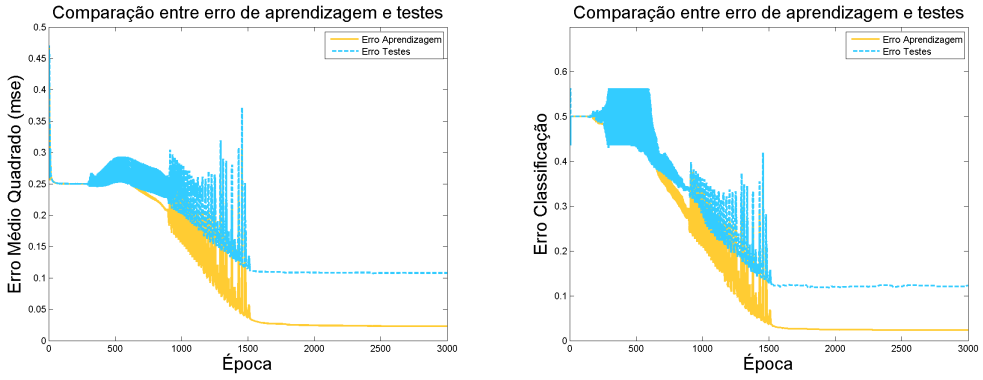
deste segundo problema. Tal como no problema 1 apenas varia o parâmetro que cada experiência pretende testar.

5.2.14 Resultados com taxas de aprendizagem diferentes



(a) (b)

Figura 5.41: *Resultados para* $\eta = 0.02$.



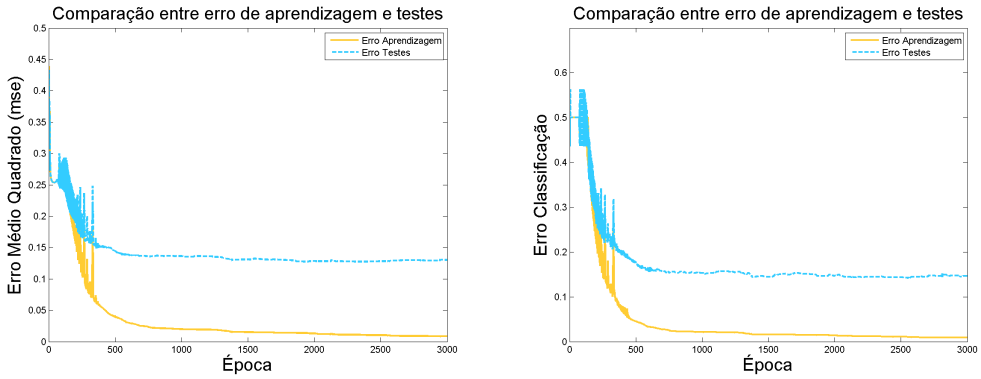
(a) (b)

Figura 5.42: *Resultados para* $\eta = 0.01$.

Neste problema decidimos testar a rede para $\eta = 0.02$ e $\eta = 0.01$. Como vemos pela comparação dos gráficos 5.41(a) e 5.42(a) e também 5.41(b)

e 5.42(b), conseguimos melhores resultados para $\eta = 0.02$ apesar de a evolução do erro ser bastante instável.

5.2.15 Resultados para diferentes números de escalas e orientações



(a) (b)

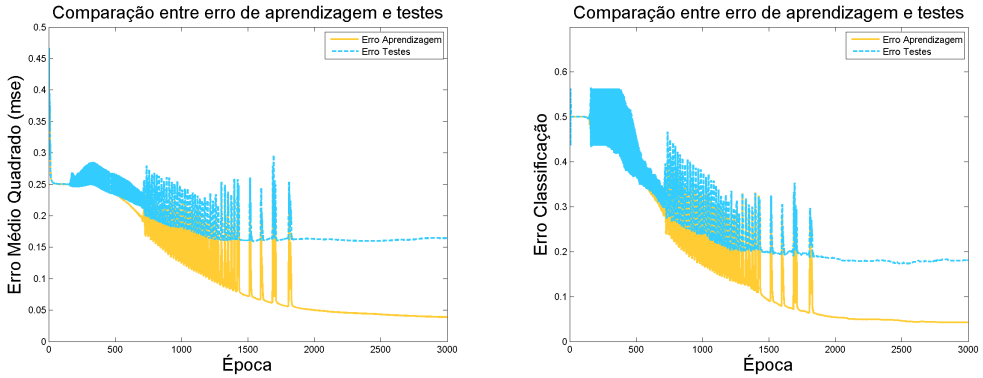
Figura 5.43: *Resultados para Nr. Escalas=2 e Nr. Orientações=2.*

5.2.16 Resultados com valores de *Shift* diferentes

Com este exemplo voltamos a verificar a importância de utilizar um valor de *Shift* que neste caso deve por volta de 3 como vemos pela comparação dos erros entre os gráficos 5.45(a), 5.44(a), 5.42(a). Neste problema embora sejam imagens com uma dimensão considerável a verdadeira informação concentra-se apenas na assinatura e é necessário que a rede seja minuciosa na sua análise pelo que um valor de 5 para o *Shift* não é suficiente como no problema anterior.

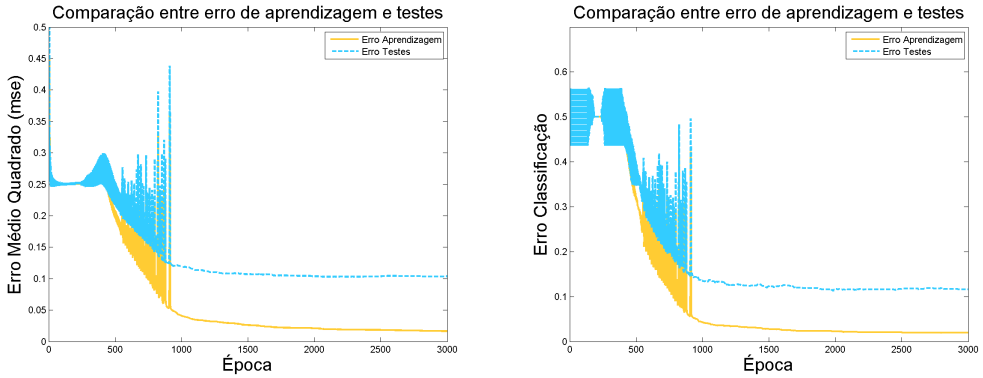
5.2.17 Intervalo de confiança para os resultados da rede

Apresentamos um diagrama de extremos e quartis dos resultados da experiência 5.2.14 para a qual recolhemos 10 amostras. Em 5.46(a) podemos



(a) (b)

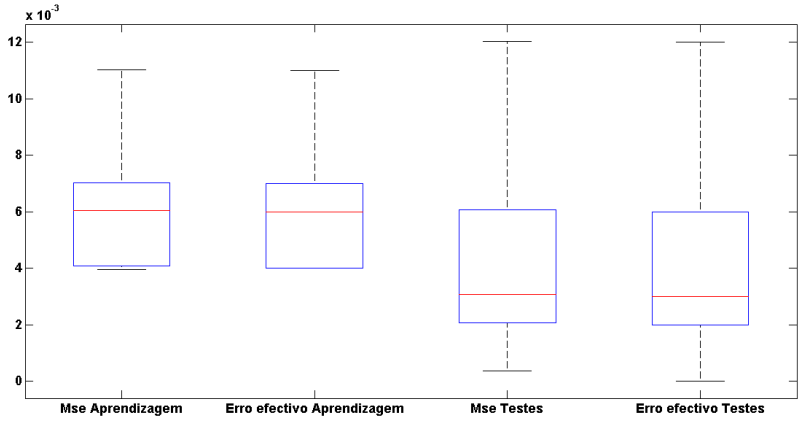
Figura 5.44: *Resultados para Shift = 10.*



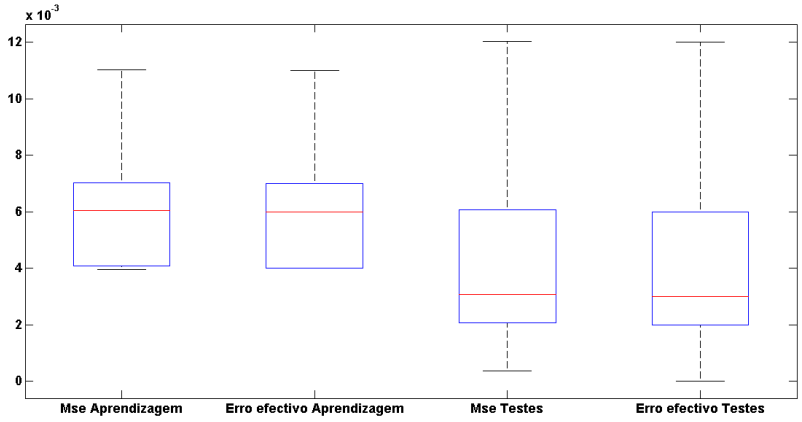
(a) (b)

Figura 5.45: *Resultados para Shift = 3.*

observar que o nosso intervalo de confiança é cerca de 1% para a aprendizagem e em 5.46(b) vemos que é cerca de 2% para os testes.



(a)



(b)

Figura 5.46: *Intervalos de confiança para os erros do treino e teste.*

Capítulo 6

Conclusão e Trabalho futuro

6.1 Conclusão

Os testes efectuados em 5.1 permitiram-nos verificar experimentalmente a influência dos vários factores de configuração de um rede FF. Por outro lado os testes realizados para a rede convoluiocnal permitiram-nos verificar que o desempenho da nossa é bastante sensível aos parâmetros especificados para o treino da mesma. Uma das conclusões que pudemos retirar foi que o valor do *Shift* é bastante importante ser adequado à dimensão das imagens e à complexidade do problema dado que se isto não acontecer podemos não ter resultados satisfatórios. Outro parâmetro importante são as características dos filtros utilizados com a orientação e a escala que pode influenciar de forma determinante o desempenho da rede, uma vez que em função do problema a tratar encontramos valores limite para o número de tipos de filtros a utilizar. Factores como a taxa de aprendizagem e a arquitectura das camadas escondidas influenciam da mesma forma a rede convolucional construída e a rede FF como vimos pelos testes realizados em ambas, contudo é sempre necessário adequar a taxa de aprendizagem ao problema em questão como pudemos ver pelos resultados obtidos no problema 1.

Em suma, podemos concluir que os objectivos do trabalho foram satisfatoriamente atingidos dado que a construção da rede neuronal FF e da convolucional foi bem sucedida e foi possível resolver com um erro satisfatório os problemas propostas aplicando as configurações ideias para o

problema em questão. Apesar de estes serem problemas introdutórios em relação aos problemas de reconhecimento em tempo real e sem cooperação como por exemplo utilizando câmaras de vigilância, são um passo importante para a iniciação de um novo tipo de abordagem em que se utiliza as redes neuronais convolucionais. Descobrir novos métodos que consigam resolver este tipo de problemas com erros inferiores aos actuais é cada vez mais importante no mundo actual, dado que a segurança é um tópico cada vez mais importante hoje em dia.

6.2 Trabalho futuro

Após uma reflexão sobre todo o trabalho desenvolvido durante este projecto pensamos este projecto pode ser melhorado em vários aspectos deixando muitas oportunidades de novos trabalhos. Consideramos que futuramente poderão ser realizados os seguintes melhoramentos:

- Construção de uma interface mais interactiva para a rede neuronal.
- Construção de um banco de assinaturas com assinaturas verdadeiras e falsificações.
- Implementação de mais funcionalidades e possibilidades de configuração da rede.

6.2.1 Construção de uma interface mais interactiva para a rede neuronal

Uma possibilidade de um projecto futuro na área da Computação Gráfica seria desenvolver uma interface intuitiva e de fácil utilização para o utilizador poder configurar a rede através desta. A comunicação entre a interface e a rede construída deveria ser transparente de tal modo que o utilizador não se apercebesse da existência das duas aplicações.

6.2.2 Construção de um banco de assinaturas com assinaturas verdadeiras e falsificações

Após os testes realizados com as assinaturas em 5.2.13 seria bastante interessante tentar abordar o problema numa outra perspectiva recolhendo assinaturas verdadeiras e também falsificações feitas por outras pessoas. O objectivo seria novamente juntar pares de assinaturas, em que uma delas era sempre verdadeira, para que a rede tentasse descobrir se a segunda imagem era uma falsificação ou não da primeira. Para a realização destes testes pensamos que seria necessário construir um banco de assinaturas mais robusto, ou seja, com bastantes exemplos e bastantes variações para que na fase de testes a rede se conseguisse adaptar bem aos novos exemplos.

6.2.3 Implementação de mais funcionalidades e possibilidades de configuração da rede

Este projecto foi apenas uma fase inicial da rede convolucional construída, pois existem muitos melhoramentos que podem ser experimentados na fase de extracção de características. A fase de junção de características pode ser abordada de outras formas e estudada com mais profundidade em função dos resultados obtidos em 5.2. Quanto à fase FF da rede convolucional pode ser melhorada com a junção de mais algoritmos de aprendizagem, como por exemplo o *Resilient Backpropagation* (RPROP) [19].

Bibliografia

[1] http://www.cns.nyu.edu/~david/courses/perception/lecturenotes/V1/lgn-V1.html.

[2] http://thebrain.mcgill.ca/flash/d/d_02/d_02_cl/d_02_cl_vis/d_02_cl_vis.html.

[3] http://dynamicnotions.blogspot.com/2008/09/single-layer-perceptron.html.

[4] http://www.scholarpedia.org/article/Neocognitron.

[5] http://www.mathworks.com/.

[6] http://www.neurosolutions.com/.

[7] http://topographica.org/.

[8] http://webvision.med.utah.edu/VisualCortex.html.

[9] J. Daugman. Two-dimensional spectral analysis of cortical receptive field profiles. *Vision Research*, 20(10):847–856, 1980.

[10] K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202, 1980.

[11] K. Fukushima. Neocognitron: A hierarchical neural network capable of visual pattern recognition. *Neural Networks*, 1(2):119–130, 1988.

[12] Kunihiko Fukushima. Neocognitron for handwritten digit recognition. *Neurocomputing*, 51:161 – 180, 2003.

[13] Kunihiko Fukushima. Restoring partly occluded patterns: a neural network model. *Neural Netw.*, 18:33–43, January 2005.

[14] Dennis Gabor. Theory of communication. *J. Inst. Elect. Eng.*, 93:429–457, 1946.

[15] J J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558, 1982.

[16] D. H. Hubel and T. N. Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 160:106–154, January 1962.

[17] Warren McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biology*, 52:99–115, 1990. 10.1007/BF02459570.

[18] Papert Minsky, Marvin; Seymour. *Perceptrons: An Introduction to Computational Geometry*. 1969.

[19] Martin Riedmiller and I. Rprop. Rprop - description and implementation details, 1994.

[20] Raul Rojas. *Neural Networks - A Systematic Introduction*. 1996.

[21] Frank ROSENBLATT. The perceptron: A perceiving and recognizing automaton. Technical report, Cornell Aeronautical Laboratory, 1957.

[22] D E Rumelhart, G E Hinton, and R J Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

[23] B. Widrow and M. E. Hoff. Adaptive switching circuits. In *Institute of Radio Engineers, Western Electronic Show and Convention, Convention Record, Part 4*, pages 96–104, 1960.