Aprendizagem Computacional

Relatório do Trabalho 3

“Prevention and Detection of Epileptic Seizures”

André Gouveia – 2014216306
Joel Pires - 2014195242

Universidade de Coimbra
Departamento de Engenharia Informática
Mestrado em Engenharia Informática

Índice

Introdução

Pretende-se com este trabalho identificar crises de epilepsia baseando-nos para isso na informação de frequências providenciadas por Eletroencefalogramas.

Através destes encefalogramas extrai-se dados relativos à atividade cerebral sendo que, a partir de um conjunto de características conseguimos identificar momentos de crises epiléticas (pontos ictais), momentos que antecedem e sucedem as crises (pré-ictais e pós-ictais respetivamente) e os restantes momentos em que não se regista atividade epilética (pontos inter-ictais).

Qual é o principal desafio então deste trabalho? Criar uma aplicação em Matlab que permita analisar esses dados recolhidos das EEG's de forma a prever ou detetar situações de crises. Para isso foram usadas redes neuronais disponibilizadas na Neural Networks Toolbox.
De forma a avaliar o desempenho e performance da nossa aplicação, calculamos a sensibilidade e especificidade em cada teste, ou seja, calculamos a percentagem de situações ictais verdadeiras detetadas (sensibilidade) e a percentagem de situações não-ictais falsas detetadas (especifidade). Relembramos que uma grande especificidade subentende uma boa deteção de momentos em que não há crises e, em contra-partida, uma grande sensibilidade remeta para uma boa deteção de situações de crise. Deixamos aqui as fórmulas usadas para estes cálculos:

$$SE = Sensitivity = \frac{True\_Positives}{True\_Positives + False\_negatives} = \frac{TP}{TP + FN}$$

$$SP = Specificity = \frac{True\_Negatives}{True\_Negatives + False\_positives} = \frac{TN}{TN + FP}$$

Pretendemos então no decorrer deste relatório apresentar mais detalhadamente a nossa aplicação, debatendo a sua implementação e refletindo criticamente sobre o seu desempenho e performance, mais especificamente sobre a sua sensibilidade e especificidade.

Implementação

CLASSIFICAÇÃO:

A aplicação pretende então, a partir de dados oriundos da EEG de um paciente, detetar situações de epilepsia. Essa classificação pode ser feita recorrendo a duas formas.

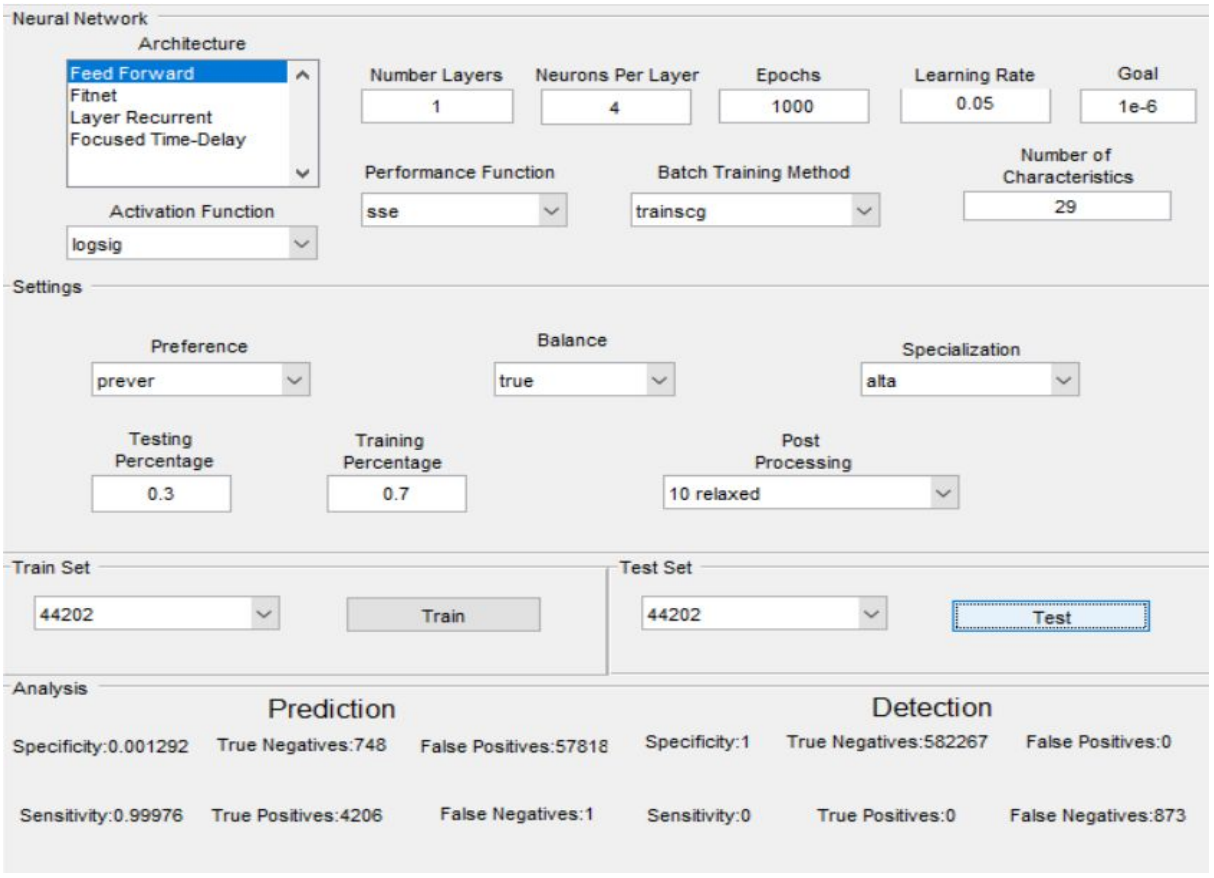
Uma dessas formas é recorrendo a uma classificação individual. Assim, a cada elemento do conjunto de dados de entrada é atribuída uma das seguintes 4 classes diferentes:

- Classe Ictal que corresponde a uma situação de crise, é representada pelo vetor [0 0 1 0]
- Classe Pré-Ictal que corresponde a uma situação de pré-crise, é representada pelo vetor [0 1 0 0]
- Classe Pós-Ictal que corresponde a uma situação de pós-crise, é representada pelo vetor [0 0 0 1]
- Classe Inter-Ictal que corresponde a uma situação onde não ocorre crise, é representada pelo vetor [1 0 0 0]

Na outra classificação nós decidimos classificar em grupo. A classificação das entradas é similar, apenas com a pequena diferença de que em vez de considerarmos cada elemento individual, classificamos conjuntos de dados. Atentemos agora à Interface desenvolvida para a aplicação.

GRAFICAL USER INTERFACE:

Tal como foi exigido no enunciado, criámos uma GUI que pudesse facilitar o utilizador realizar os testes e treinar as redes neuronais modificando diferentes parâmetros. Abaixo encontra-se uma imagem da nossa GUI:



Gostaríamos de destacar que é possível ao utilizador modificar os seguintes parâmetros:

- **Rede Neuronal:** Feed Forward, Fitnet, Layer Recurrent, Focused Time-Delay
- **Funções de Aprendizagem para o Treino da Rede Neuronal:** trainscg, traingd, traingda, traingdm, trainlm e traingdx.
- **Número de Características a Analisar:** por defeito é 29.
- **Função de Performance:** sse, mse, sae e mae
- **Função de Ativação:** logsig, purelin e transig
- **Número de Camadas Escondidas:** por defeito é 1.
- **Número de Neurónios por Camada Escondida:** por defeito é 10
- **Número de Épocas para o Treino:** por defeito é 1000.
- **Ritmo de Aprendizagem:** por defeito é 0.05.
- **Objetivo da Aprendizagem:** por defeito é 1e-6.
- **Preferência da classificação:** prever ou detetar.
- **Balancear ou Não os pontos Interictais:** Sim ou Não.
- **Especialização:** Alta, Média ou Baixa
- **Percentagem de Crises para Teste:**por defeito é 0.3

- **Percentagem de Crises para Treino:**por defeito é 0.7
- **Tipo de Post-Processing:** 10 relaxed, 5 strict, nothing
- **Escolha do Dataset:** 44202.mat, 54802.mat

REDES NEURONAIS USADAS:

Como já citado damos a hipótese ao utilizador de escolher 4 arquiteturas diferentes, a saber: Feed Forwar, Fitnet, Layer Recurrent, Focused Time-Delay.

A rede Layer Recurrent permite introduzir os chamados delays nalgumas características o que lhe permite prever qualquer output dinâmico baseando-se em inputs passados. Assemelha-se assim ao cérebro de uma pessoa dado que tem memória e é um sistema dinâmico.

Assim sendo, estas redes dinâmicas, uma vez que se assemelham mais ao funcionamento do cérebro humano, são melhores candidatas para identificar as crises epiléticas. Falamos não só da Layer Recurrent mas também da Focused Time-Delay.

Em relação agora à rede Feed Forward gostaríamos de mencionar que também é uma boa rede neuronal candidata dado que permite uma boa implementação de qualquer função de entradas e saídas, desde que sejam levados em conta neurónios suficientes nas camadas escondidas.

TRAINING:

Para o presente trabalho foram-nos fornecidos dados relativos a dois pacientes (48202.mat e 54802.mat).

Gostaríamos desde já mencionar um desafio inicial que tivemos. Naturalmente, a quantidade de dados que são relativos a momentos em que não ocorre crise (pontos interictais) é muito maior que os dados em que ocorrem crises epiléticas. Nesses casos, iremos verificar uma especialização da rede na identificação de situações não-ictais, sem que faça uma classificação de casos ictais igualmente fiável.

Para evitar uma especialização em casos não-ictais deixamos que o utilizador defina a percentagem de crises reservadas para treino e para teste. Depois disso e dado que o número de casos inter-ictais é bem maior que as restantes classes, ao selecionarmos um número de situações não-ictais igual ao de situações ictais temos, necessariamente de não incluir a maior parte das situações não-ictais.

No treino destas redes neuronais recorremos ainda a diferentes funções de batch training, disponíveis e implementadas pela Neural Network Toolbox do Matlab tais como: trainscg, traingd, traingda, traingdm, trainlm e traingdx.

Dado que treinar redes exige muito poder computacional visto que o volume de dados até é grande, estas redes neuronais podem ser treinadas recorrendo ao GPU do computador. Para isso adicionámos os parâmetros à função de train(): 'useGPU', 'yes'.

TESTING:

O treino acabou e um ficheiro .mat com as especificações usadas no treino é criado. Este ficheiro é mais tarde para se poder testar esta rede neuronal treinada de forma a verificar a sua performance, o seu bom ou mau funcionamento.

Testes e Observações

Em anexo a este documento encontra-se uma lista detalhada dos resultados obtidos em cada teste realizado, que poderá ser consultada pelo leitor.

Na lista apresentada optámos por não incluir os resultados relativos aos testes para as redes treinadas com a função sum squared error como função de performance, uma vez que as redes treinadas com esta função obtiveram valores de performance registados pela Neural Network Toolbox muito superiores aos registados para as redes treinadas com a função de performance mean squared error.

Por essa razão, ao procedermos à análise dos resultados obtidos não iremos incidir sobre as redes treinadas com a função de performance sse, uma vez que os seus resultados não apresentam grandes diferenças, nem se destacam por realizarem uma classificação aceitável.

Regra geral, as diferentes redes testadas apresentaram valores de especificidade e sensibilidade que consideramos aceitáveis para o trabalho em questão, isto é, iguais ou superiores a 0.8.

De facto, seria de esperar que possuindo um número de neurónios igual ao número de características consideradas no treino, a rede realizasse melhores classificações, no entanto tal não se verificou.

Ao treinar o paciente 54802.mat, considerando 10 neurónios e testes com 30% das situações ictais com a função trainscg verificou-sea ocorrência de valores altos de especificidade, mas reduzidos para a sensibilidade.

Relativamente redes Focused Time Delay e Fitnet obtiveram resultados muito próximos, registando-se muitas oscilações entre os valores de especificidade e sensibilidade para o mesmo teste. Os melhores resultados registados para estas redes foram conseguidos utilizando a função trainscg como função de treino.

Regra geral, as redes mostraram o mesmo comportamento que nos testes anteriores, tendo-se registado algumas variações face ao observado anteriormente, que queremos destacar:

Em primeiro lugar, nos testes agora realizados as variações, para a mesma situa ção, de valores muito elevados de especificidade e muito reduzidos de sensibilidade (e vice-versa) não só estão presentes como são mais regulares e evidentes: Por exemplo, nas redes Layer Recurrent, Focused Time Delay e Fitnet, registámos várias situações com especificidades de 1, e sensibilidades de 0, e vice-versa.

TODOS OS TESTES ENCONTRAM-SE NUM FICHEIRO EXCEL DENTRO DO ZIP ENTREGÁVEL COM O NOME: “resutadosAC.xlsx”

Conclusão

Depois de analisar os testes e resultados, algumas conclusões surgem. Em ambos os pacientes notámos que as configurações que apresentaram melhores resultados foram as seguintes:

- Rede FeedForward com 10, 20 e 30 neurónios, usando o trainscg
- Rede Layer Recurrent com 10, 20 e 30 neurónios, usando o trainscg
- Rede Distributed Time Delay, com 10, 20 e 30 neurónios, usando o trainscg

Relativamente aos pacientes/ficheiros onde se obteve melhores resultados foi sem dúvida o 44202.mat.

Com respeito às funções de batch training experimentadas, a trainscg foi aquela que nos permitiu ter melhor desempenho independentemente da rede utilizada.

Tentámos ainda diminuir as características numa tentativa de aumentar o desempenho. Contudo, verificou-se variações pouco significativas na sensibilidade e especificidade dado que o número de características não é assim tão grande para justificar um grande impacto.

Ao considerarmos somente as características que têm uma maior contribuição para a saída desejada é de esperar que a aprendizagem realizada pela rede não seja muito prejudicada. Tal só será possível unicamente se o número de características em análise for muito baixa, pondo de parte as características que têm influência significativa na determinação do valor obtido à saída da rede.

Procedendo à eliminação das características mais ambíguas, é de esperar que a rede apresenta uma maior e mais rápida convergência, melhorando também a sua performance e o seu tempo de treino, o que foi por nós verificado.