Redes Neuronais para o Diagnóstico de Parkinson

Relatório Final

3º ano do Mestrado Integrado em Engenharia Informática e Computação

Inteligência Artificial

Elementos do grupo:
João Pedro Miranda Maia - 201206047 - ei12089@fe.up.pt
Miguel Oliveira Sandim - 201201770 - miguel.sandim@fe.up.pt

31 de Maio de 2015

1 Introdução

No âmbito da Unidade Curricular de Inteligência Artificial foi proposta a resolução de um problema de classificação fazendo uso de técnicas de *Machine Learning*, nomeadamente Redes Neuronais Artificiais.

O problema abordado consiste no diagnóstico de Síndrome de Parkinson com base na análise das características de várias gravações da voz de um determinado paciente. A criação do modelo de diagnóstico será baseado numa base de dados com gravações de quarenta indivíduos (vinte dos quais apresentam Síndrome de Parkinson e os restantes são saudáveis).

O presente relatório procura tornar transparentes a especificação do trabalho desenvolvido, nomeadamente os algoritmos e metodologias utilizados, e as propostas de solução apresentadas, acompanhadas das respetivas avaliações numéricas. Finalmente, será feito um balanço final do trabalho desenvolvido e das soluções em análise.

2 Especificação do Trabalho

2.1 Base de Conhecimento

A base do conhecimento fornecida [1] divide-se em duas categorias:

- Conjunto de dados de treino do modelo: destinados a treinar a rede (nomeadamente os pesos entre os neurónios das diferentes camadas) para uma categorização eficaz de cada uma das *samples*.
- Conjunto de dados de teste do modelo: destinados a avaliar a performance da rede no diagnóstico de novos casos.

No caso do conjunto de dados de treino, este é constituído por entradas relativas a gravações de voz de 40 pacientes, 20 dos quais apresentam Síndrome de Parkinson e os restantes 20 são saudáveis. Cada individuo apresenta 26 gravações (e portanto 26 entradas no conjunto de dados) com a entoação de diversos conteúdos:

- Vogal "a"(1 gravação);
- Vogal "o"(1 gravação);
- Vogal "u"(1 gravação);
- Números de 1 até 10 (10 gravações);
- Pequenas frases (4 gravações);
- Palavras específicas (9 gravações).

Cada registo apresenta 26 atributos relacionadas com o registo voz do participante durante a gravação, assim como o UPDRS (*Unified Parkinson's Disease Rating Scale*) e a classe ao qual o participante pertence (Pacientes com Síndrome de Parkinson ou Pacientes Saudáveis).

Relativamente ao conjunto de dados de teste, este possui entradas relativas a 28 doentes de Síndrome de Parkinson. Cada indíviduo apresenta 6 gravações com a entoação de diversos conteúdos:

- Vogal "a"(3 gravação);
- Vogal "o"(3 gravações).

Cada registo apresenta os mesmos atributos que o conjunto de dados de treino, à exceção do UPDRS.

2.2 Abordagem com Redes Neuronais Artificiais

2.2.1 Conceito e Arquitectura

Para este problema de classificação é proposta uma resolução utilizando Redes Neuronais Artificiais multi-camada do tipo *Feedforward*, fazendo uso do algoritmo de treino *Backpropagation*. Este tipo de redes também são designados por *Multilayer Perceptrons*.

Uma Rede Neuronal Arificial corresponde a um modelo computacional com base num grafo cujos nós representam estruturas designadas neurónios. Uma ligação entre dois neurónios i, j tem como objetivo propagar um valor numérico a_i de um neurónio i para j, estando a cada ligação associado um peso $w_{i,j}$. Assim, o *output* (a_j) de um neurónio j, que depende do seu *input* (in_j) - resultante dos neurónios que se ligam a este - é definido segundo a equação 1 [2]. A função $g()$ corresponde à função de ativação do neurónio.

$$a_j = g(in_j) = g\left(\sum_{i=0}^{n} w_{i,j} a_i\right) \tag{1}$$

No caso dos *Multilayer Perceptrons*, a rede é dividida em camadas constituídas por neurónios (uma *input layer*, uma *output layer* e várias *hidden layers*), e cada neurónio de cada camada recebe *inputs* de neurónios da camada anterior, processa-os e transmite-os a nós da camada seguinte, formando um grafo direto acíclico [2].

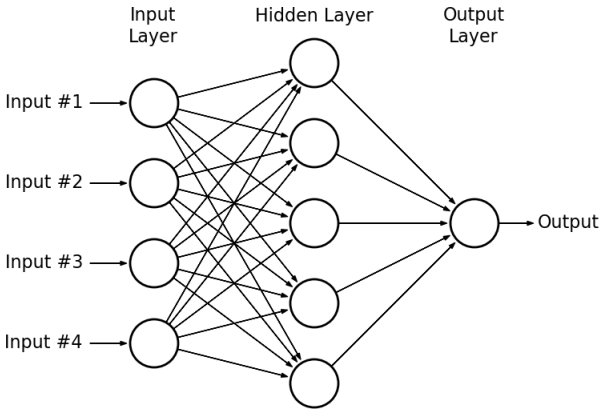

Figura 1: Exemplo da arquitectura de uma Rede Neuronal Artificial Feedforward com uma *hidden layer*.

Desde modo, os *inputs* fornecidos na camada inicial (que ao contrário das outras camadas, não processam os valores de acordo com a equação 1, passando-

os apenas à próxima camada) podem ser propagados através da rede, de modo a obter-se os valores da camada de *output*.

Durante a resolução deste problema de classificação, foi também adicionado um neurónio adicional em cada camada (à exceção da de *output*) de nome *Bias Neuron* [3], possuindo ligações a todos os neurónios da camada seguinte e nenhuma ligação aos neurónios da camada anterior, e tendo a particularidade de possuír sempre *output* igual a 1. Imaginando uma rede neuronal como um modelo que implementa uma determinada função que resulta da combinação de *outputs* dos neurónios em cada camada, este tipo de neurónios atua como um *shift*/deslocamento dessa função de modo a aproximar ainda mais o *output* resultante ao *output* esperado.

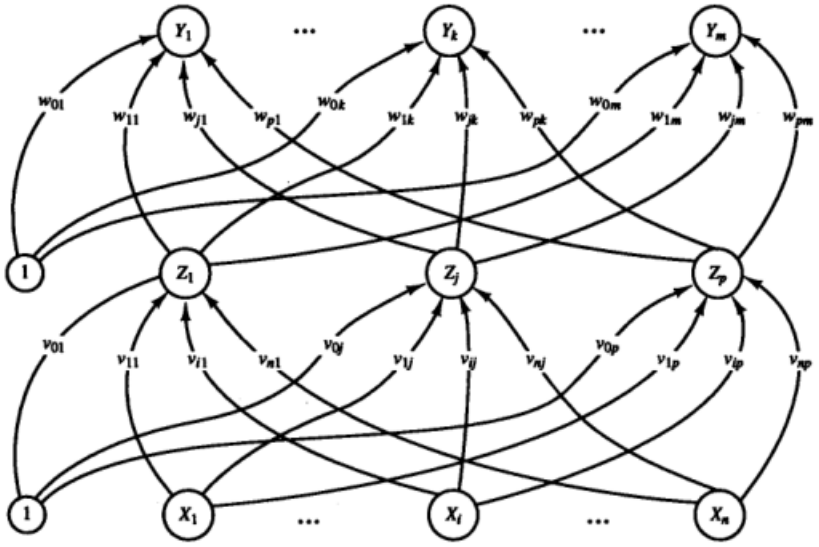

Figura 2: Exemplo da arquitectura de uma Rede Neuronal Artificial Feed-forward com uma *hidden layer* com neurónios *bias* - representados com um "1".

2.2.2 Algoritmo de aprendizagem

Para que a rede possa prever os *outputs* desejados dado um certo conjunto de *inputs*, é necessário treinar a rede, i.e. determinar os pesos $w_{i,j}$ que garantem o menor erro na previsão dos valores de *output* face aos valores esperados. Para isso, foi implementado o algoritmo *Backpropagation* na variante *Momentum* (algoritmo 1) [3]: a ideia deste algoritmo é propagar o erro no *output* para as restantes camadas (de acordo com os pesos $w_{i,j}$) e aplicar um fator de correção dos pesos das ligações (δ) entre os neurónios utilizando otimização convexa (e.g. *Gradient Descent*). A variante *Momentum*, além de possuir um parâmetro $\alpha \in [0, 1]$ que indica a "velocidade da aprendizagem" -"learning rate"- (i.e. incremento na direção da minimização da função de erro através das mudanças nos pesos), possui um parâmetro adicional $\eta \in [0, 1]$ (designado de "momentum") que indica a dependência da direção de minimização atual face à da iteração

anterior. Esta dependência é calculada segundo a seguinte fórmula:

$$\Delta w_{i,j}(t+1) = w_{i,j}(t) - w_{i,j}(t-1) \tag{2}$$

A inclusão do parâmetro η permite aumentar a rapidez na convergência do algoritmo e uma diminuição da probabilidade de aprisionamento em mínimos locais [3].

Algoritmo 1 Implementação do algoritmo de aprendizagem baseado no *Back-propagation*.

```
1:  procedure BACKPROPAGATION(Rede, Conjunto de Treino)
2:      for all pesos w_{i,j} na Rede do
3:          w_{i,j} ← número aleatório entre -1 e 1
4:      repeat
5:          for all exemplos (x,y) no Conjunto de Treino do
6:              exemplos ← shuffle(exemplos)
7:              /* Inicialização dos inputs */
8:              for all neurónios i na InputLayer do
9:                  a_i ← x_i
10:             /* FeedForward dos inputs */
11:             for all Hidden/OutputLayers l na Rede do
12:                 for all neurónios j na Layer l do
13:                     in_j ← Σ_i w_{i,j} a_i
14:                     a_j ← g(in_j)
15:             /* Backpropagation */
16:             for all neurónios j na OutputLayer do
17:                 δ_j ← g'(in_j) × (y_j − a_j)
18:             for all Hidden/InputLayers l na Rede do
19:                 for all neurónios i na Layer l do
20:                     δ_i ← g'(in_i) × Σ_j w_{i,j} δ_j
21:             /* Atualização dos pesos */
22:             for all pesos w_{i,j} na Rede do
23:                 w_{i,j} ← w_{i,j} + α × a_i × δ_j + η × Δw_{i,j}
24:     until aprendizagem bem sucedida
```

2.2.2.1 Cálculo do Erro Para avaliação do erro durante a aprendizagem, é calculado o MSE (*Mean Squared Error*) do conjunto de treino. Esta métrica, dado um conjunto de treino com n exemplos e o *outputs*, é dada pela seguinte fórmula:

$$\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{o}(\widehat{y}_i^k - y_i^k)^2 \tag{3}$$

sendo $\widehat{y}_i^k$ o valor obtido no neurónio de *output* k com o exemplo de *input* i e y_i^k o valor esperado.

2.2.2.2 Condição de Paragem da Aprendizagem Como condição de paragem do algoritmo (para determinar que a rede aprendeu suficientemente

bem o conjunto de treino) é aplicado um valor máximo para o MSE (*Mean squared error*) do conjunto de treino e um valor máximo de iterações. Assim que um destes limites seja atingido, a aprendizagem termina e dá-se lugar ao teste do modelo com um conjunto separado.

2.2.3 Propostas de Solução

2.2.3.1 Pré-Processamento dos Dados Para a introdução dos dados na rede, é feita uma normalização dos valores de cada atributo, segundo a fórmula 4 [4].

$$X_n = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{4}$$

2.2.3.2 Função de ativação Durante o curso deste trabalho, será utilizada a função sigmoide $f(x) = \frac{1}{1+e^x}$ como função de ativação, já que esta apresenta $D'_f = [0, 1]$ e os valores de *input/output* estarão normalizados entre 0 e 1.

2.2.3.3 Arquitetura da rede A arquitetura implementada da rede apresenta:

- 26 neurónios na camada de *input* (para os 26 atributos de cada entrada na base do conhecimento - é ignorado o atributo UPDRS por não estar presente no conjunto de dados de teste);
- Um número variável de *hidden layers* e de neurónios nestas;
- 2 neurónios na camada de *output*.

Dado que o problema é de natureza de classificação, procedeu-se à criação de dois neurónios na camada de *output* cujo valor representa a pertença de uma *sample* à categoria em questão (Pacientes com Parkinson ou Pacientes Saudáveis). No caso da situação da classificação de uma *sample* considera-se que esta se insere na categoria que possuir o *output* com maior valor.

2.2.3.4 Treino da rede Foram identificadas quatro possíveis estratégias para treino do modelo:

- Treinar a rede utilizando as 26 *audio samples* de cada um dos 40 sujeitos de treino como *samples* independentes, fazendo uso de todas as *samples* de teste durante o processo de teste;
- Treinar a rede utilizando apenas as *audio samples* correspondentes às vogais "a"e "o", de cada um dos 40 sujeitos de treino como *samples* independentes, fazendo uso de todas as *samples* de teste durante o processo de teste (de modo a treinar e testar o modelo com sons semelhantes);
- Treinar uma rede neuronal, utilizando apenas as *audio samples* correspondentes à vogal "a"como *samples* independentes, fazendo uso apenas das *samples* relativas a esta vogal durante o processo de teste;

- Treinar uma rede neuronal, utilizando apenas as *audio samples* correspondentes à vogal "o"como *samples* independentes, fazendo uso apenas das *samples* relativas a esta vogal durante o processo de teste.

Para cada um destes casos serão testados diferentes valores de α e η no processo de treino, assim como diferente número de camadas intermédias e neurónios.

3 Desenvolvimento da Aplicação

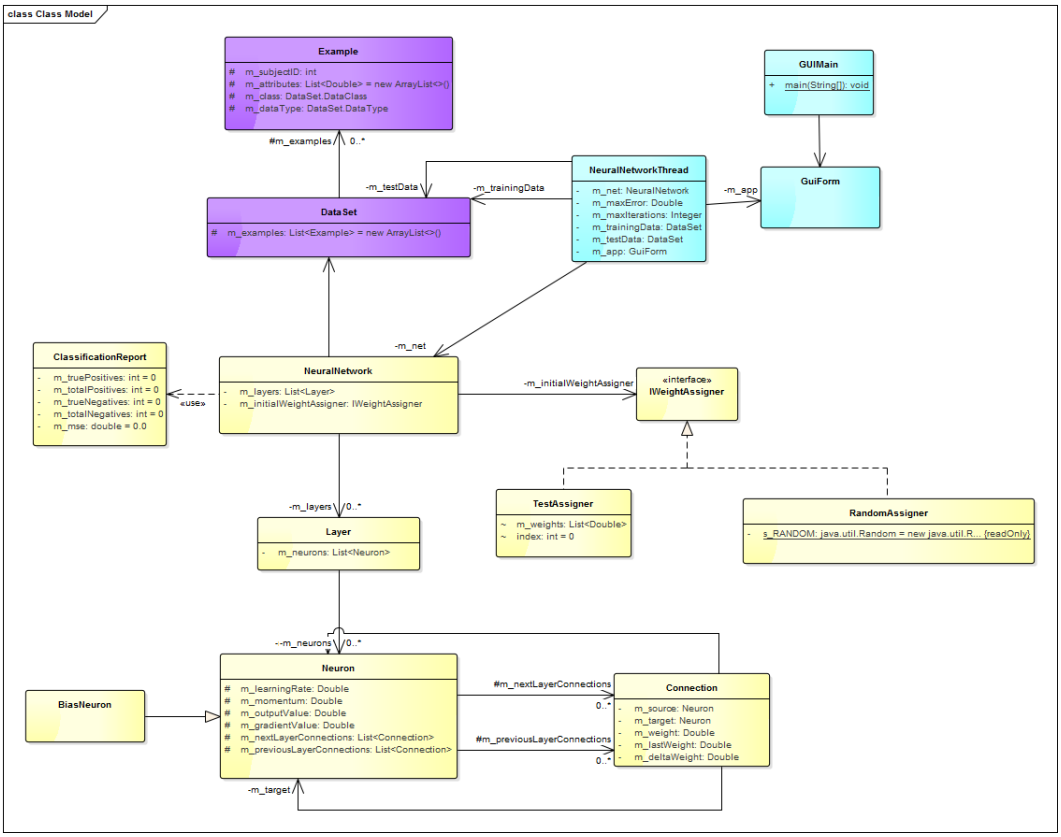

Figura 3: Diagrama de classes da aplicação desenvolvida. A amarelo o *package neuralNetwork*, a azul o *package data* e a violeta o *package gui*.

Para o desenvolvimento desta aplicação foi utilizado o IDE IntelliJ IDEA em ambiente Windows 8.1 com a linguagem Java 8, assim como o uso da biblioteca de interfaces gráficas "Swing". Foram também utilizados testes unitários, com o auxílio do JUnit, para validar o sucesso do algoritmo de aprendizagem e outras operações na aplicação.

Apresenta-se na figura 3 o diagrama de classes da aplicação, constituído pelos seguintes módulos:

- Módulo da Rede Neural ("neuralNetwork"), responsável pela implementação da ANN e do algoritmo de aprendizagem;
- Módulo de Dados ("data"), responsável pelo *parsing* dos ficheiros de dados fornecidos;
- Módulo de Interface Gráfica ("gui"), responsável pela criação da interface gráfica com o utilizador.

4 Experimentação das Propostas de Solução e sua Análise

4.1 Método de Análise dos Resultados

		Previsto	
		Paciente com Parkinson	Paciente Saudável
Atual	Paciente com Parkinson	*True positive* (TP)	*False negative* (FN)
	Paciente Saudável	*False positive* (FP)	*True negative* (TN)

Tabela 1: Definição das medidas de desempenho.

Dada a natureza do problema (diagnóstico de pacientes), foram definidas medidas de desempenho definidas de acordo com a tabela 1 (com base em [1]). Contudo, dado que o conjunto de teste fornecido apenas contém gravações de pacientes com Síndrome de Parkinson, apenas as medidas TP (*True positive*) e FN (*False negative*) serão utilizadas. Optou-se por utilizar este conjunto de teste, mesmo apresentado esta desvantagem na avaliação dos resultados, de modo a preservar-se o maior número de *samples* na fase de treino, dado que algumas das abordagens a analisar usam apenas uma *sample* por individuo durante o treino.

Assim define-se a seguinte métrica como avaliação da exatidão de cada modelo treinado, relativamente ao número de *samples* bem classificadas:

$$\frac{TP}{TP + FN} \tag{5}$$

Tal como definido nas secções anteriores, coloca-se a *sample* na categoria que possuir o neurónio com *output* mais elevado. Em caso de empate, a *sample* é classificada como sendo de um paciente com Parkinson.

Dado que possam existir casos em que uma *sample* possui ambos os *outputs* elevados (ou ambos baixos) - e.g. *output* "Parkinson": 0.93 / *output* "Saudável": 0.95, utiliza-se também o MSE para avaliar a solução (dado que este considera a distância do *output* esperado ao realmente obtido em ambos os neurónios).

Todos os modelos serão treinados com base nos valores dos parâmetros $\alpha = 0.1$, $\eta = 0.7$, MSE máximo igual a 0.001 e número máximo de iterações igual a 10000, por estes serem os parâmetros que melhor se aplicam na generalidade

das situações com o *dataset* apresentado. São utilizados também no treino de todos os modelos os 26 atributos por cada *sample*.

Como sugerido em [4], o número de neurónios nas camadas intermédias deve variar entre o número de neurónios nas camadas de *input* e *output*, podendo também ser igual à soma do número de neurónios na camada de *input* mais o da camada de *output* mais um, pelo que se optou por testar diversos valores de nós entre 2 a 29.

Como defendido por Fausett [3], uma camada intermédia é considerada suficiente para a maioria dos casos. Contudo a existência de 2 camadas intermédias pode facilitar o treino do modelo, pelo que se estudarão as duas hipóteses.

4.2 Análise do Modelo 1

Neste modelo consideram-se no treino todas as 26 *samples* de cada um dos indivíduos, considerando também no teste todas as 6 *samples* de cada um dos indivíduos (vogal "a" e vogal "o").

Nº Neurónios	2	5	10	15	20	25	29
MSE (teste)	0.6374	1.1763	1.0922	1.3084	0.9535	1.0256	1.1578
Exatidão (teste)	**0.6488**	0.4107	0.4167	0.3333	0.5119	0.4821	0.4167

Tabela 2: Resultados para o modelo 1 com uma camada intermédia.

Nº Neurónios	2	5	10	15	20	25	29
MSE (teste)	0.3730	0.8835	0.8218	0.6659	1.1454	0.6753	0.9032
Exatidão (teste)	**0.7798**	0.4762	0.5714	0.6607	0.4167	0.6369	0.4762

Tabela 3: Resultados para o modelo 1 com duas camadas intermédias.

A exatidão máxima deste modelo face ao conjunto de teste foi de 78% (com um MSE de 0.3730), tendo esta sido obtida com duas camadas intermédias, com 2 neurónios cada.

4.3 Análise do Modelo 2

Neste modelo consideram-se no treino todas as *samples* referentes às vogais "a" e "o" de cada um dos indivíduos, considerando no teste todas as 6 *samples* de cada um dos indivíduos (vogal "a" e vogal "o").

Nº Neurónios	2	5	10	15	20	25	29
MSE (teste)	0.6760	0.3186	0.3467	0.3753	0.3983	0.3602	0.3625
Exatidão (teste)	0.6191	**0.8214**	**0.8214**	0.7917	0.7798	0.8036	0.8095

Tabela 4: Resultados para o modelo 2 com uma camada intermédia.

Nº Neurónios	2	5	10	15	20	25	29
MSE (teste)	1.2155	0.4738	0.3717	0.2996	0.3587	0.3751	0.2799
Exatidão (teste)	0.3393	0.7440	0.8036	0.8333	0.8035	0.7917	**0.8452**

Tabela 5: Resultados para o modelo 2 com duas camadas intermédias.

A exatidão máxima deste modelo face ao conjunto de teste foi de 85% (com um MSE de 0.2799), tendo esta sido obtida com duas camadas intermédias, com 29 neurónios cada.

4.4 Análise do Modelo 3

Neste modelo consideram-se no treino todas as *samples* referentes à vogal "a"de cada um dos indivíduos, considerando no teste apenas as *samples* relativas à vogal "a".

Nº Neurónios	2	5	10	15	20	25	29
MSE (teste)	1.2921	0.6272	0.2576	0.2672	0.5374	0.2619	0.3913
Exatidão (teste)	0.3333	0.6786	**0.8690**	0.8571	0.7143	**0.8690**	0.7857

Tabela 6: Resultados para o modelo 3 com uma camada intermédia.

Nº Neurónios	2	5	10	15	20	25	29
MSE (teste)	0.3207	0.3035	0.5229	0.3813	0.5808	0.2683	0.5589
Exatidão (teste)	0.8333	0.8333	0.7262	0.7738	0.6905	**0.8571**	0.7024

Tabela 7: Resultados para o modelo 3 com duas camadas intermédias.

A exatidão máxima deste modelo face ao conjunto de teste foi de 87% (com um MSE de 0.2576), tendo esta sido obtida com uma camada intermédia com 10 neurónios.

4.5 Análise do Modelo 4

Neste modelo consideram-se no treino todas as *samples* referentes à vogal "o"de cada um dos indivíduos, considerando no teste apenas as *samples* relativas à vogal "o".

Nº Neurónios	2	5	10	15	20	25	29
MSE (teste)	0.6210	0.4990	0.9398	0.4751	0.9570	0.6025	0.6066
Exatidão (teste)	0.6786	0.7381	0.5000	**0.7500**	0.4405	0.7024	0.6786

Tabela 8: Resultados para o modelo 4 com uma camada intermédia.

Nº Neurónios	2	5	10	15	20	25	29
MSE (teste)	0.6315	0.3788	0.2810	0.5431	0.3343	0.3148	0.3591
Exatidão (teste)	0.6667	0.8095	**0.8571**	0.6667	0.8214	0.8333	0.8095

Tabela 9: Resultados para o modelo 4 com duas camadas intermédias.

A exatidão máxima deste modelo face ao conjunto de teste foi de 86% (com um MSE de 0.2810), tendo esta sido obtida com duas camadas intermédias, cada uma com 10 neurónios.

5 Conclusões

O resultados experimentais permitem afirmar que o método e a metodologia propostos conseguem garantir uma classificação com alguma exatidão (no melhor caso classificando 87% das *audio samples* corretamente).

Tal como previsto, o modelo 1 obteve as piores classificações (78% no melhor dos casos), dado que o treino é feito com mais tipos de gravações do que os existentes no conjunto de teste, o que introduz ruído nos padrões que poderão ser apreendidos pela rede relativamente às vogais "a" e "o". O terceiro modelo foi o mais exato, obtendo no máximo uma percentagem de 87% de classificações corretas. Seria de esperar que o modelo 2 obtivesse uma exatidão inferior à observada, já que mistura no treino e no teste dois tipos de gravações, o que poderia causar uma aprendizagem errada de padrões e consequentemente classificações erradas. Contudo, este modelo obteve valores de exatidão e MSE muito próximos dos modelos 3 e 4.

Considera-se assim que os modelos 2, 3 e 4 poderiam ser eventualmente aplicados no contexto da vida real, sendo necessário um estudo mais aprofundado na validade destes.

A variabilidade nos valores do MSE (e também no valor da exatidão da classificação) em cada modelo realçam a necessidade de se adaptar a arquitetura da rede (número de camadas intermédias e número de neurónios nestas) aos conjuntos de dados a tratar.

6 Trabalho Futuro

Uma possível ordem de trabalhos futuros neste método poderiam incluir: uso e adaptação de algoritmos para evitar a ocorrência *overfitting* durante o processo de aprendizagem (e.g. *cross-validation*) e estudo de forma mais aprofundada no efeito da escolha do intervalo de atribuição de pesos aleatórios no início do algoritmo de aprendizagem na convergência do algoritmo (já que diferentes execuções do algoritmo originam resultados diferentes por vezes).

7 Percentagens de trabalho do grupo

Considera-se que ambos os membros trabalharam de forma igual para a conclusão deste trabalho e relatório, pelo que se atribui 50% a cada um dos membros do grupo.

Referências

[1] Betul Erdogdu Sakar et al. "Collection and analysis of a Parkinson speech dataset with multiple types of sound recordings". Em: *Biomedical and Health Informatics, IEEE Journal of* 17.4 (2013), pp. 828–834.

[2] Stuart J. Russell e Peter Norvig. *Artificial Intelligence: A Modern Approach*. 2ª ed. Pearson Education, 2003. ISBN: 0137903952.

[3] Laurene Fausett, ed. *Fundamentals of Neural Networks: Architectures, Algorithms, and Applications*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1994. ISBN: 0-13-334186-0.

[4] Milica Stojković. *Wine Classification using Neural Networks*. URL: `http://neuroph.sourceforge.net/tutorials/wines1/WineClassificationUsingNeuralNetworks.html`.

Apêndice

A Manual do utilizador

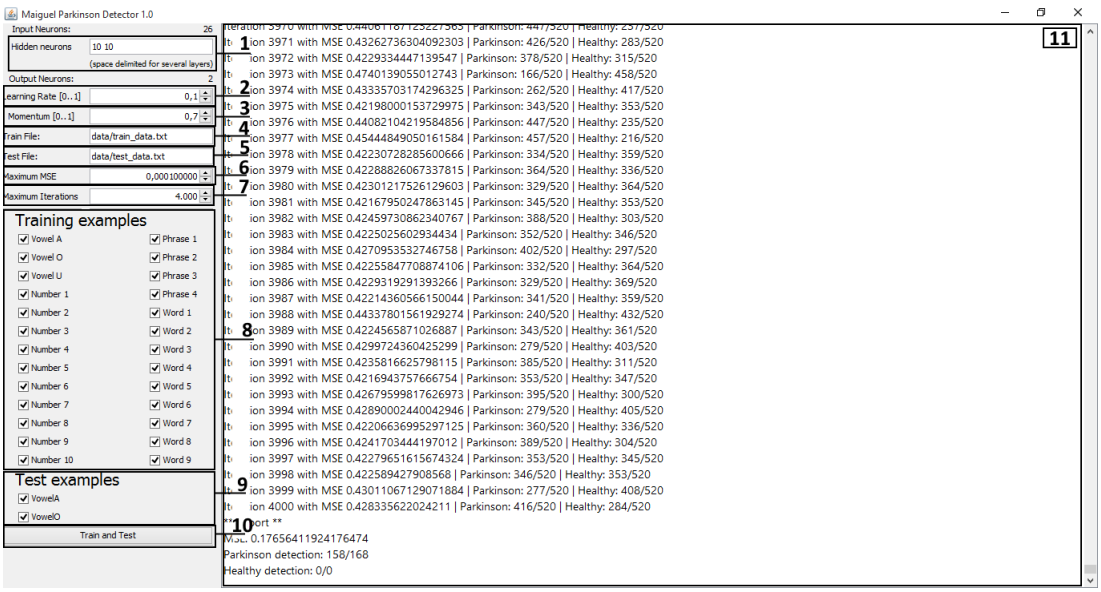

Figura 4: Interface gráfica da aplicação desenvolvida.

O zip de submissão inclui um ficheiro *jar* contendo a aplicação compilada, que pode ser de imediato executada.

Apresenta-se de seguida a descrição textual de cada um dos elementos da UI:

1. Campo para introdução do número de neurónios por cada camada (delimitadas por um espaço em branco).
2. Alteração do *learning rate*.
3. Alteração do *momentum*.
4. Caminho para o ficheiro que contém os dados de treino.
5. Caminho para o ficheiro que contém os dados de teste.
6. Erro máximo (MSE) da condição de paragem do algoritmo de aprendizagem.
7. Número máximo de iterações da condição de paragem do algoritmo de aprendizagem.
8. Seleção dos tipos de exemplos de treino.
9. Seleção dos tipos de exemplos de teste.

10. Botão que inicia o treino da rede neuronal, utilizando o conjunto de exemplos de treino, testando depois cada um dos exemplos do conjunto de teste.

11. Painel onde é mostrado, inicialmente, um sumário da configuração da rede neuronal artificial e onde é, de seguida, exibido o resultado de cada iteração de treino. Depois de concluídas todas as iterações, são mostrado os resultados de teste.