Relatório Intercalar

Redes Neuronais para a identificação de Pulsares

Mestrado Integrado em Engenharia Informática e Computação

IART - Inteligência Artificial

Turma 3MIEIC02, Grupo E1_3 :
André Miguel Ferreira da Cruz - 201503776
Edgar Filipe Amorim Gomes Carneiro - 201503784
João Filipe Lopes de Carvalho - 201504875

Faculdade de Engenharia da Universidade do Porto
Rua Roberto Frias, s/n, 4200-465 Porto, Portugal

8 de Abril de 2018

Conteúdo

1 Objetivo

Este trabalho tem como objetivo a aplicação de Redes Neuronais artificiais na identificação de Pulsares. Neste sentido, começamos por expor e explicitar o tema, com uma pequena introdução à tarefa em questão, e às dificuldades à sua resolução (secção 1). De seguida, analisamos o *dataset* em uso, descrevendo características e possíveis tendências (*bias*) indesejadas (secção 2.1.1). No mesmo sentido, explicamos o pré-processamento necessário (secção 2.1.2), bem como o modelo de aprendizagem a aplicar (secção 2.1.3) e a sua arquitetura (2.1.4). Finalmente, expomos o trabalho efetuado até ao momento (secção 2.2), os resultados esperados, e as métricas usadas para avaliação (secção 2.3). Para concluir, extraímos algumas conclusões dos resultados obtidos, e traçamos metas para trabalho futuro.

Os pulsares são um tipo de estrela de neutrões raro que, devido ao seu intenso campo magnético, transforma energia cinética (rotacional) em energia eletromagnética. Além disso, o seu campo magnético é suficientemente forte para arrancar partículas da sua superfície (na sua maioria eletrões), que são depois aceleradas na *magnetosfera* e emitidas sob a forma de um feixe estreito e contínuo. Este feixe de radiação, juntamente com a contínua rotação do pulsar, gera um sinal característico, análogo ao de um farol em rotação, de intensidade suficiente para ser detetado a milhões de anos-luz de distância. Desta forma, a deteção de um pulsar corresponde à procura e reconhecimento dos seus sinais periódicos.

Cada pulsar produz um padrão próprio, distinto dos restantes corpos celestes, e que varia ligeiramente entre diferentes pulsares. Assim, podemos detetar um sinal realizando uma análise de alguns parâmetros, como por exemplo o tempo médio que um pulsar demora a reemitir o seu feixe para a terra. À primeira vista, concluir se um sinal é realmente proveniente de um pulsar pode parecer pouco complexo. No entanto, devido a fortes interferências de frequências de rádio e de sinais ruidosos que dificultam a receção de sinais genuínos, a grande maioria das supostas deteções são falsas.

Devido às dificuldades referidas anteriormente, e ao grande volume de dados de sinais (induzido maioritariamente por falsas deteções), esta tarefa torna-se extenuante e dispendiosa para humanos, tornando-a uma muito boa candidata para a aplicação de técnicas de *Machine Learning*. *Machine Learning* é um subcampo da inteligência artificial no qual as aplicações de *software* têm a capacidade de aprender e obter resultados precisos, sem que no entanto sejam explicitamente programadas. Os algoritmos de *machine learning* funcionam através da construção de um modelo a partir de inputs amostrais, com a finalidade de fazer previsões ou decisões orientadas pelos dados e respetivas estatísticas ao invés de seguir estruturas de decisão estáticas.

Assim sendo, neste trabalho será usado um sub-ramo da área de *machine learning*, redes neuronais artificias, como forma de resolução do problema de classificação apresentado (se um dado sinal foi ou não emitido por um pulsar).

2 Descrição

2.1 Especificação

2.1.1 Descrição e análise do dataset.

Na base de dados fornecida, cada candidato é caraterizado por oito variáveis continuas e uma classe. As oito variáveis contínuas distinguem-se em dois grupos, sendo as primeiras quatro referentes a estatísticas relativas ao *folded profile* de um pulsar, e as restantes quatro relativas às características da curva *DM-SNR* (*Delta Modulation with Signal-to-Noise Ratio*). O primeiro grupo consiste num *array* de variáveis contínuas que descrevem uma versão pós-análise do pulsar. E as variáveis do segundo grupo são obtidas através da análise da curva *DM-SNR*. Esta consiste na modulação de um sinal analógico para digital ou o seu reverso. Por sua vez, a classe fornecida como última parâmetro tem um binário, indicando se o candidato era de facto um pulsar ou não. Assim, os nove parâmetros analisados são:

1. Média do *folded profile*

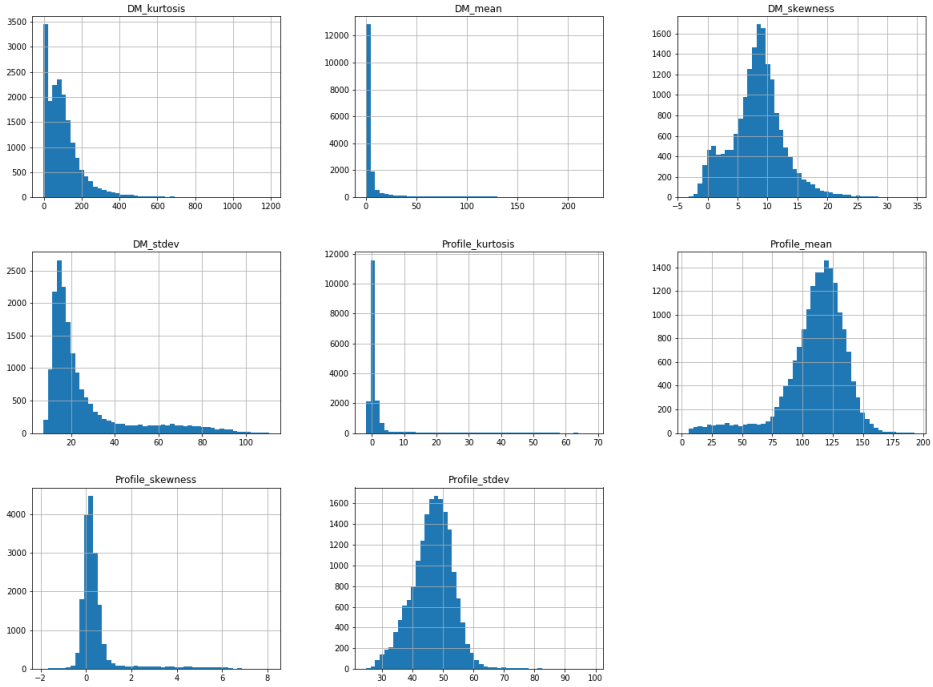

Figura 1: Distribuições relativas às 8 variáveis contínuas

2. Desvio padrão do *folded profile*
3. Curtose do *folded profile*
4. Assimetria do *folded profile*
5. Média da curva *DM-SNR*
6. Desvio Padrão da curva *DM-SNR*
7. Curtose da curva *DM-SNR*
8. Assimetria da curva *DM-SNR*
9. Classe resultado

A base de dados fornecida contém um total de 17,898 entradas, sendo que destas, 16,259 tratam-se de dados espúrios causados pelo ruído e as restantes 1,639 tratam-se de dados relativos a exemplos de pulsares reais. Foi ainda possível obter a distribuição associada a cada uma das variáveis contínuas, estando estas presentas na figura 1. Essas mesmas distribuições encontram-se pormenorizadas na figura 2, sob a forma de tabela.

Nas situações de teste, os valores das oito variáveis são usados na análise do candidato de forma a determinar o valor binário que terá a classe passada como nono parâmetro.

	Profile_mean	Profile_stdev	Profile_skewness	Profile_kurtosis	DM_mean	DM_stdev	DM_skewness	DM_kurtosis
count	17898.000000	17898.000000	17898.000000	17898.000000	17898.000000	17898.000000	17898.000000	17898.000000
mean	111.079968	46.549532	0.477857	1.770279	12.614400	26.326515	8.303556	104.857709
std	25.652935	6.843189	1.064040	6.167913	29.472897	19.470572	4.506092	106.514540
min	5.812500	24.772042	-1.876011	-1.791886	0.213211	7.370432	-3.139270	-1.976976
25%	100.929688	42.376018	0.027098	-0.188572	1.923077	14.437332	5.781506	34.960504
50%	115.078125	46.947479	0.223240	0.198710	2.801839	18.461316	8.433515	83.064556
75%	127.085938	51.023202	0.473325	0.927783	5.464256	28.428104	10.702959	139.309330
max	192.617188	98.778911	8.069522	68.101622	223.392141	110.642211	34.539844	1191.000837

Figura 2: Estatísticas descritivas sumárias, relativas às 8 variáveis de *input*.

2.1.2 Pré-processamento dos dados.

Relativamente à etapa de pré-processamento dos dados, temos duas considerações importantes: por um lado, todas as colunas de *input* contêm valores de vírgula flutuante, de distribuição contínua; por outro lado, a coluna de *output*, corresponde a uma de duas classes (classe positiva ou negativa, caso seja ou não um pulsar).

Assim, para facilitar a aprendizagem por parte do algoritmo e contribuir para convergir mais rapidamente, fazemos normalização dos dados de input (*feature scaling*), para uma escala no alcance [-1,1]. Por sua vez, relativamente à classe de *output*, extraímos os dados para uma matriz unidimensional em que cada elemento tem um valor binário: 1 caso seja referente a um pulsar, e 0 em caso contrário.

2.1.3 Modelo de aprendizagem a aplicar: redes neuronais.

Neste trabalho será usada uma rede neuronal artificial que segue um modelo especial de **aprendizagem supervisionada** denominado ***backpropagation***. Assim, torna-se importante esclarecer cada um destes conceitos.

Algoritmos que seguem o modelo da aprendizagem supervisionada são algoritmos nos quais os possíveis *outputs* do algoritmo já são conhecidos e nos quais os dados usados para treinar o algoritmo já se encontram mapeados à resposta correta. Pode-se verificar que este modelo é aplicável ao problema dos pulsares na medida em que a totalidade das entradas na base de dados se encontram mapeados a um resultado, como já foi referido previamente.

Algoritmos de *backpropagation* são algoritmos de aprendizagem supervisionada que usam a noção de gradiente descendente.

O algoritmo do gradiente descendente é um algoritmo iterativo de otimização de primeira ordem para encontrar o mínimo de uma função. Em cada iteração, de forma a encontrar o mínimo da função, o algoritmo avança de forma proporcional ao valor negativo do gradiente naquele ponto da função

Dada uma rede neuronal artificial e um função de erro o algoritmo de *backpropagation*, também conhecido por algoritmo da propagação de erros para trás (*backward propagation of errors*), calcula o gradiente da função de erro considerando sempre o peso das arestas na rede neuronal. Este cálculo é propagado para trás na rede neuronal, com o gradiente da última camada de pesos a ser calculado primeiro e o gradiente da primeira camada de pesos em último lugar (daí o seu nome *backpropagation*). Este fluxo invertido da informação de erro permite uma computação eficiente do gradiente em cada camada, ao invés da abordagem ingénua de calcular o gradiente de cada camada separadamente.

O algoritmo desenvolve-se em duas fases. Numa primeira fase, propaga-se através da rede de forma a chegar aos valores *output*. De seguida, calculo o custo da propagação para cada output, sendo que esta estará intrinsecamente relacionada com a função de erro usada no algoritmo do gradiente descendente. Por

fim, propaga-se em sentido contrário de forma a gerar um valor de erro para cada um dos neurónios. Na segunda fase, os pesos das arestas são constantemente atualizados conforme o resultado desempenhado por cada um dos candidatos. Na atualização das arestas é usado uma percentagem previamente definida como fator de atualização dos pesos. Quanto mais elevado for este fator, mais rapidamente será o treino da rede neural. Por outro lado, quanto mais baixo for este fator, mais preciso será o treino.

2.1.4 Arquitetura das redes neuronais.

As redes neuronais, surgiram com base na perceção biológica das interligações presentes no cérebro humano. Sendo o cérebro humano constituído por neurónios, os quais recebem impulsos nervosos como entrada e saída de dados. Um neurónio tem a capacidade de estimular os seus neurónios vizinhos, quando este é propriamente estimulado (função de ativação). Posto isto, podemos definir matematicamente um neurónio como sendo:

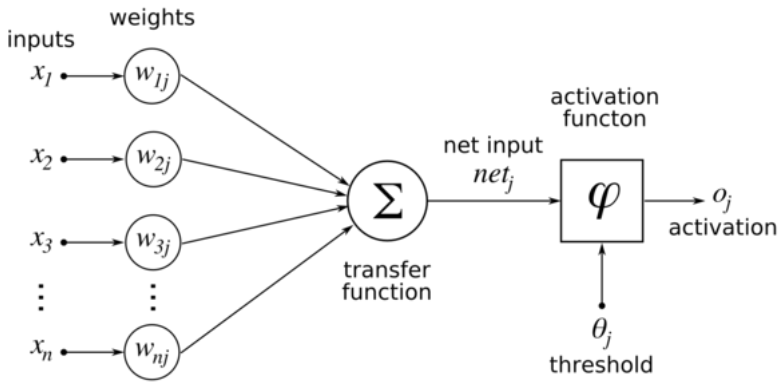

Figura 3: Modelo Matemático de um Neurónio

$$y = f_{ativação}(\sum_i w_i x_i - \theta)$$

Figura 4: Representação Matemática da Saída de um Neurónio

Como o nome indica, é uma rede, por isso a sua consistência detêm vários neurónios conectados na forma de um grafo acíclico. As redes neuronais mais comuns, *feedforward*, são redes em que o *output* de certos neurónios é o *input* de outros. Desta forma é possível separar a rede em diferentes camadas:

1. Uma ***Input Layer***, primeira camada na rede neuronal, responsável por receber os dados, determinar o tamanho do *dataset* e apresentar os padrões de reconhecimento. São apenas constituídas por neurónios de *input*.

2. Uma ou mais ***Hidden Layer***, é nesta(s) camada(s) que é feita maior parte da aprendizagem dos pesos do modelo. Camadas seguidas deverão corresponder a *features* cada vez mais abstratas, permitindo alcançar um modelo generalizável e de bom desempenho.

3. Uma ***Output Layer***, última camada da rede, responsável por apresentar os dados, esta camada é constituída com um número de neurónios geralmente igual ao número de classes de *output* (ou apenas 1 se se tratar de classificação binária). É uma camada que não dispõe de função de ativação, e os cujos são exclusivamente neurónios de *output*.

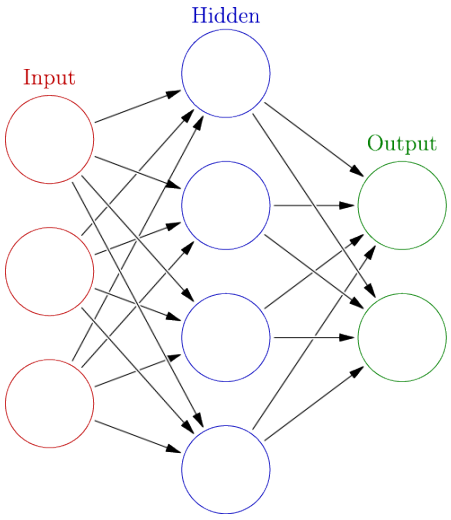

Figura 5: Representação Abstrata da Rede Neural

2.1.5 Configuração da rede neuronal.

Partindo agora para o nosso caso em concreto, é sempre muito complicado de prever ao certo qual é que a configuração/arquitetura ideal para uma rede, isto é, qual o número ideal de neurónios de *input*, qual o número ideal de *hidden layers* e quantos neurónios em cada uma. Por isso é algo que requer alguns testes com diferentes composições para se obter os melhores resultados. Foram então desenvolvidos testes ao número de *hidden layers*

2.2 Trabalho Efetuado

2.3 Resultados esperados e forma de avaliação

3 Conclusões

4 Recursos