Instituto Superior de Engenharia de Coimbra

Licenciatura Eng. Informática

3ºAno/1ºSemestre

2020/2021

INTELIGÊNCIA COMPUTACIONAL

PROCESSO DE OTIMIZAÇÃO DO MODELO

Projeto – Fase III – Fashion Mnist – PSO - Otimização do Modelo

Carolina Ferreira nº 2018018459

Eduardo Pina nº20170101

ÍNDICE

INTRODUÇÃO

No âmbito da unidade curricular de Inteligência Computacional foi proposta a realização de um Projeto cujo processo se baseia na implementação e validação de técnicas aplicadas na área de Inteligência Computacional.

O objetivo deste trabalho consiste na resolução de um problema de classificação já proveniente da Fase I. Dos temas propostos nessa fase, a nossa escolha incidiu no Fashion Mnist. É importante referir que na primeira fase do projeto a implementação do problema foi realizada em Matlab e, portanto, achamos pertinente proceder novamente à implementação, mas desta vez em Python de modo a facilitar a comparação diferenciativa do "antes" e do "depois" de otimização através de gráficos/plots iguais.

A resolução deste problema apresenta um grau de dificuldade algo complexo, uma vez que recorre a um algoritmo de Redes Neuronais combinado com um algoritmo de otimização, que por sua vez procura uma melhor aproximação aos parâmetros mais adequados para a Rede – os pesos e viés de cada neurónio. A nossa expectativa inicial é que esta solução consiga gerar melhores resultados comparativamente a algoritmos mais simples, no que diz respeito à sua qualidade de predição, que é traduzida por uma maior taxa de acerto em termos da classificação.

Numa primeira instância apresentamos uma descrição do caso de estudo, onde será exposto de forma mais detalhada o problema que se pretende resolver. Segue-se uma exposição da metodologia de trabalho, aludindo aos passos a realizar desde a aquisição do conjunto de dados até à fase de classificação. Em sequência a isto, apresenta-se uma ilustração do algoritmo através do desenho de um diagrama de classes, no qual serão definidas as principais classes e a relação estabelecida entre elas.

A implementação dos algoritmos é abordada com exposição de todas as etapas referidas na metodologia. Após a aplicação do algoritmo de Swarm Intelligence do tipo de PSO ao problema de classificação inicial, serão apresentados e discutidos os resultados obtidos. Por fim, serão apresentadas as principais conclusões do trabalho. Nesta secção serão apresentadas adicionalmente as vantagens e/ou desvantagens desta metodologia relativamente a outras técnicas de pesquisa em grelha.

1. DESCRIÇÃO DO CASO DE ESTUDO

O objetivo deste trabalho consiste na obtenção de uma Rede Neuronal que efetue a classificação de peças de vestuário em ***T-shirt/top, Trouser, Pullover, Dress, Coat, Sandal, Shirt, Sneaker, Bag, Ankle boot***, a partir de um dataset disponibilizado na fase I. Este dataset conta com 60000 exemplos para treino e 10000 exemplos para teste e encontra-se nos ficheiros excel "fashion-mnist_test" e "fashion-mnist_train" onde constam as imagens convertidas para decimal. Dada a elevada dimensionalidade dos conjuntos de dados, este trabalho propõe a extração das características mais significativas para a classificação, ou seja, as que terão maior correlação com o output final. Isto permite evitar o fenómeno designado como a "maldição da dimensionalidade", que prejudica o desempenho do algoritmo – tanto em relação à sua performance na classificação como no domínio temporal – com variáveis com pouco ou nenhum valor acrescentado. A técnica utilizada para este propósito será a *Particle Swarm Optimization*.

2. DESCRIÇÃO DA METODOLOGIA

Neste capítulo serão abordadas as etapas de todo o processo, ilustradas pela Figura 1 para uma melhor compreensão. A primeira etapa consiste na aquisição dos conjuntos de dados do problema. O pré-processamento é o próximo passo, e consiste na correção de alguns problemas dos *datasets*, como por exemplo o tratamento de valores omissos ou inválidos. A fase que se segue é a de segmentação e consiste no tratamento dos dados em secções de valores. As fases de *Data Acquisition*, *Pre-processing* e *Segmentation* não são efetuadas no decorrer deste trabalho, uma vez que já o foram previamente realizadas pelos autores dos conjuntos de dados. Assim, as fases trabalhadas neste Projeto serão as fases de **Extração de Características** e de **Classificação** e **Otimização**, que serão explicitadas mais detalhadamente.

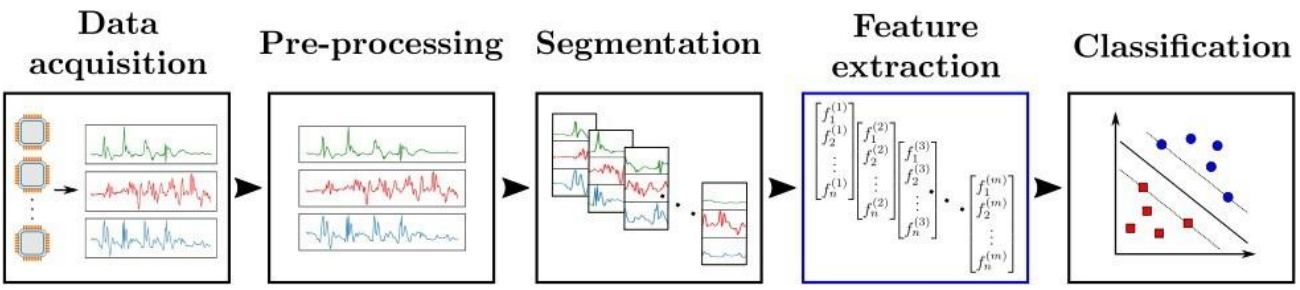


Figura 1. Ilustração geral da metodologia implementada

2.1. SELEÇÃO DAS CARACTERÍSTICAS

A seleção dos inputs mais relevantes foi feita com recurso a um *Particle Swarm Optimization* (PSO) Binário. Com esta técnica é possível retirar do conjunto de dados as características que não têm grande relevância para o modelo, enquanto reduzem o tempo de execução do algoritmo de treino da rede neuronal.

A implementação deste "BinaryPSO" foi analisada na aula laboratorial 7 e o código utilizado no trabalho é proveniente da ficha realizada nessa mesma aula, disponibilizada no moodle.

2.2. CARACTERÍSTICAS

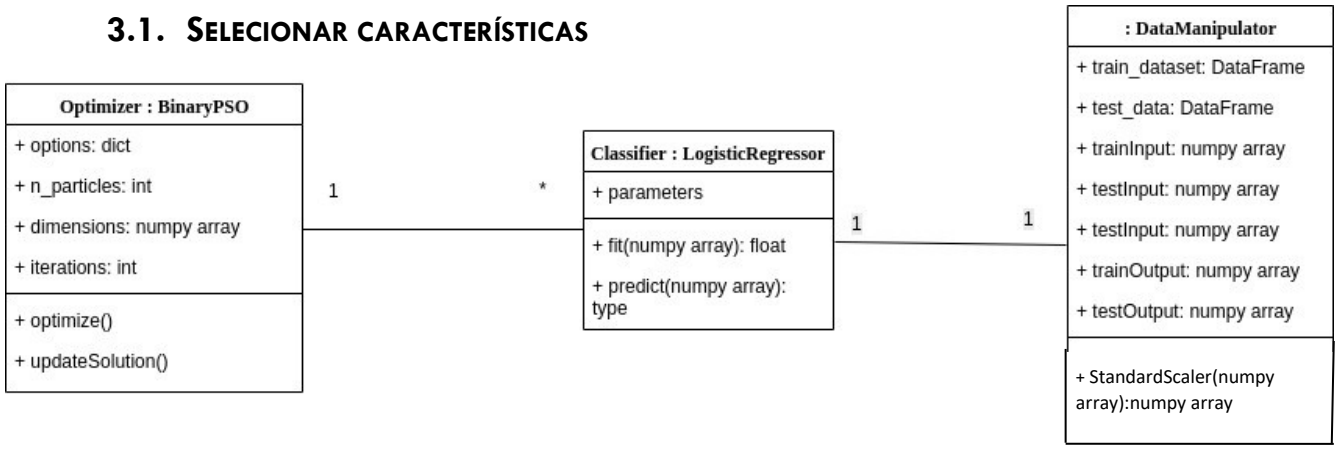
A classificação é realizada ao subconjunto de dados derivado do conjunto inicial com o intuito de se poder extrair resultados comparáveis com os da fase final deste projeto. É usada uma rede MLP (Perceptrão Multicamada) tanto para treino como para predição a partir do subconjunto dos dados de teste. Os resultados a analisar são a matriz de confusão, com valores normalizados e não normalizados, Accuracy, Precisão, Recall, F1-Score e F-Measure globais e Relatório de Classificação detalhado por classe e onde constam as curvas ROC e AUC.

Após esta fase, segue-se a aplicação do PSO a uma rede neuronal, cujo objetivo é o de descobrir os melhores parâmetros para a mesma. Estes parâmetros correspondem aos pesos (quanto maior o valor do *peso*, mais excitatório é o estímulo sináptico) e viés (elemento que serve para aumentar o grau de liberdade dos ajustes dos pesos) de cada neurónio. A partir de um conjunto possível de soluções, cada partícula do PSO equivale a uma rede neuronal que irá devolver o resultado do melhor custo local e a melhor posição de todas as partículas. No final, a melhor partícula corresponderá à rede neuronal otimizada, ou seja, aquela que apresentará menor custo e os melhores parâmetros.

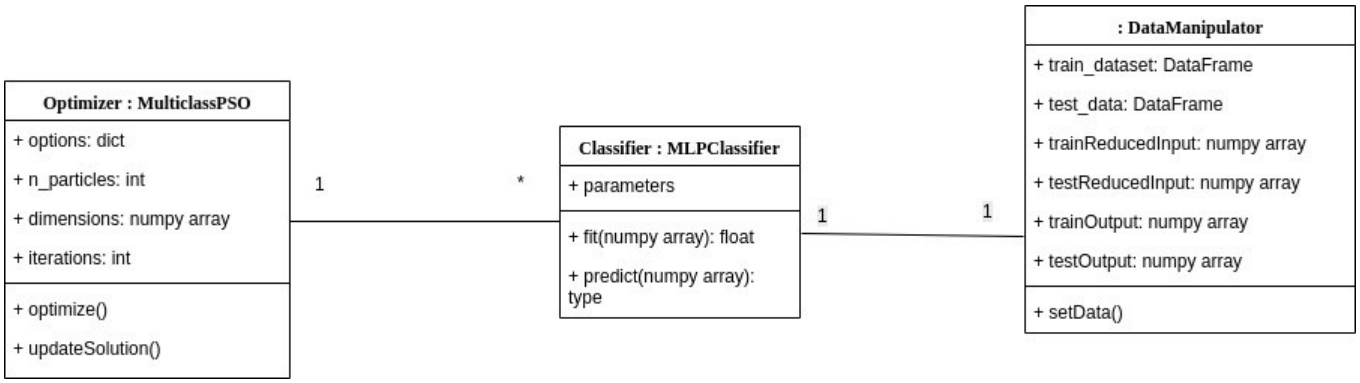
Os resultados (métricas) a observar mantêm-se para que se possa efetuar a análise comparativa válida. Por outro lado, serão efetuados vários testes a diferentes tipologias de rede, o que significa que diferentes camadas de diferentes neurónios serão consideradas para cada teste.

3. DESENHO – DIAGRAMA DE CLASSES

3.1. SELECIONAR CARACTERÍSTICAS



3.2. SELECIONAR A MELHOR REDE



4. IMPLEMENTAÇÃO

O algoritmo implementado anteriormente implementado em Matlab e agora em Python está estruturado da seguinte forma:

I. Fase 1 – Treino da rede:

1) Carregamento do dataset para o programa, através da leitura do ficheiro .csv, onde estão armazenados os dados para treino;

2) Pré-processamento dos dados, a partir da padronização dos inputs, como foi explicado anteriornente, já que as MLP são sensíveis a valores que não estão na mesma escala;

3) Parametrização da rede: neste passo serão especificados (e modificados de teste para teste) os seguintes hiperparâmetros: número de camadas internas da rede e número de neurónios por camada e função de ativação;

4) Treino da rede

II. Fase 2 – Teste da rede e métricas de desempenho:

1) Carregamento do dataset para o programa, através da leitura do ficheiro .csv, onde estão armazenados os dados para teste;

2) Pré-processamento dos dados para uma escala standard;

3) Teste à rede usando a melhor configuração obtida na fase I do projeto (como a análise de resultados e retirada de conclusões foi realizada nesta altura, não há necessidade de repetir esse longo processo), através da predição das classes de cada exemplo de teste, ou seja, a rede irá classificar cada um dos exemplos de teste numa das 10 classes definidas;

4) Computação e apresentação da matriz de confusão geral, com valores absolutos – quantos exemplos foram classificados corretamente, quantos foram classificados como sendo da classe positiva e pertencem a outra (falsos positivos), e quantos foram classificados como outra classe e pertencem à classe real (falsos negativos) – e com valores relativos;

5) Cálculo da accuracy, precisão, sensibilidade, f1-score e f-measure, como valores médios;

6) Apresentação de quadro com valores de precisão, sensibilidade e f1-score, detalhados por classes;

7) Cálculo das probabilidades das instâncias pertencerem a cada uma das classes, para efeitos de construção da curva ROC e para cálculo da AUC;

8) Visualização das curvas ROC e respetivo valor da AUC.

4.1. Selecionar as melhores características do conjunto de dados para o modelo

1) Criar um classificador binário de Regressão Logística para fazer a classificação dos melhores inputs para a rede;
2) Criar função para classificação de cada partícula e devolução do resultado da computação da função objetivo;
3) Criar função que realize a tarefa do ponto 2. para todas as partículas;
4) Inicializar os parâmetros para o otimizador: opções e dimensões;
5) Criar uma instância de um PSO binário e inicializar com os parâmetros do ponto 4;
6) Realizar a otimização com a instância do PSO criada;
7) Criar o subconjunto de dados classificados positivamente no passo anterior – de treino e de teste. Estes serão os inputs utilizados para treinar as futuras redes;
8) Realizar o treino de uma rede neuronal a partir do subconjunto de treino obtido;
9) Realizar a classificação dos exemplos do subconjunto de teste com a rede treinada no ponto 8;

4.2. Visualização De Métricas

10) Criar matriz de confusão com identificação das classes;

11) Cálculo da accuracy, precision, recall, f1-score e f-measure médias em relação à classificação realizada no ponto 9;
12) Criar um relatório de classificação com as métricas precision, recall e f1-score, detalhadas por classe;
13) Apresentar os gráficos das curvas ROC e calcular o AUC para cada classe.

4.3. APLICAR O *PARTICLE SWARM OPTIMIZATION* COMO OTIMIZADOR DOS PARÂMETROS DA REDE NEURONAL

14) Selecionar os dados de treino e de teste;
15) Definir os vetores de pesos e bias para fornecer à rede;
16) Definir a função de custo;
17) Fornecer os inputs da rede e a função de custo ao otimizador;
18) Calcular a melhor solução para os parâmetros da melhor rede neuronal até à condição de paragem (número de iterações, por norma);
19) Testar a rede obtida com o dataset de teste;
20) Realizar as tarefas dos pontos 8 ao 13 – classificação com os parâmetros encontrados pelo otimizador e visualização dos resultados.

5. APRESENTAÇÃO DOS RESULTADOS E ANÁLISE

PSO – Parâmetros	
c1	**0,5**
c2	**0,5**
W	**0,9**
K	**8**
P	**2**
Alfa	**0,88**
Iterações	**50**
Partículas	**20**

Este subconjunto foi utilizado para treino e classificação nos passos seguintes deste projeto que correspondem aos momentos antes e após a obtenção dos melhores pesos e *bias* para a rede.

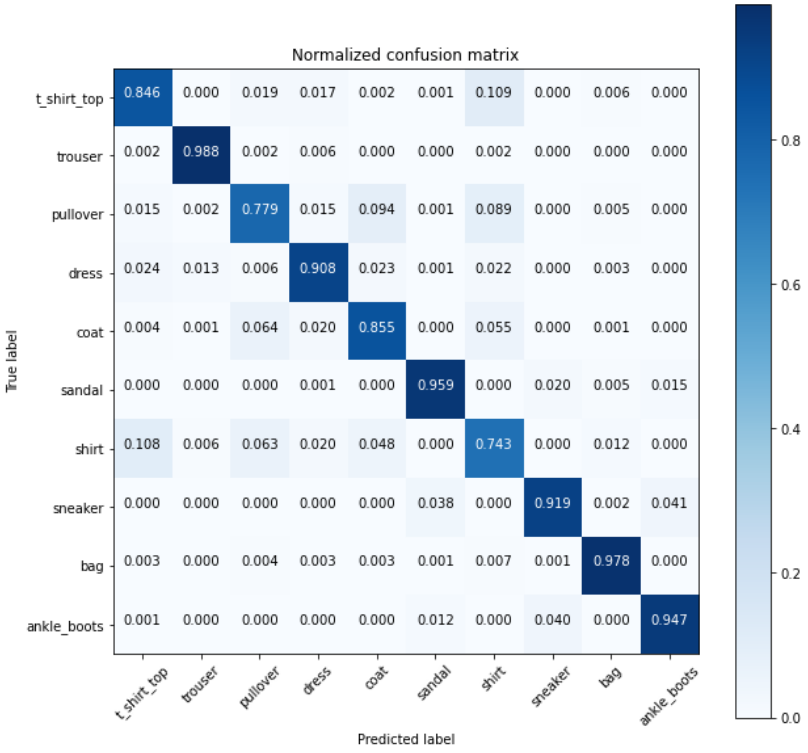
5.1. CLASSIFICAÇÃO ANTES DA OTIMIZAÇÃO DOS PARÂMETROS DA REDE NEURONAL

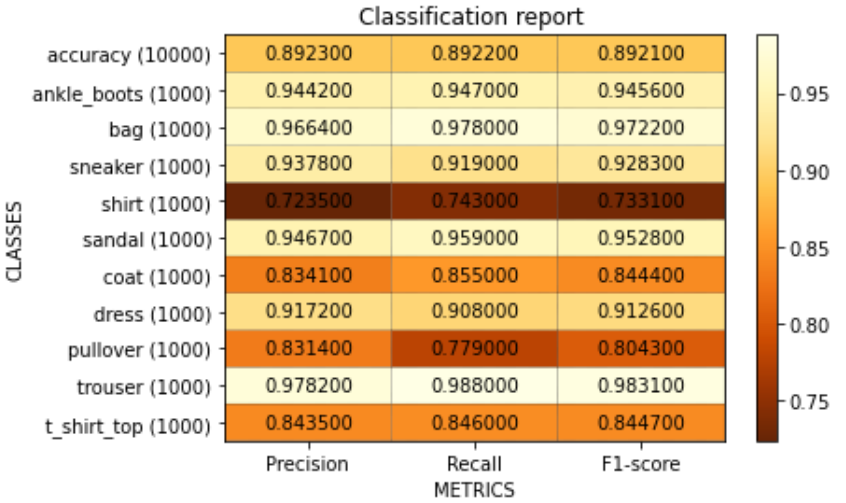
Após a seleção das melhores características para o modelo, submeteu-se este subconjunto a um classificador do tipo Perceptrão Multi Camada.

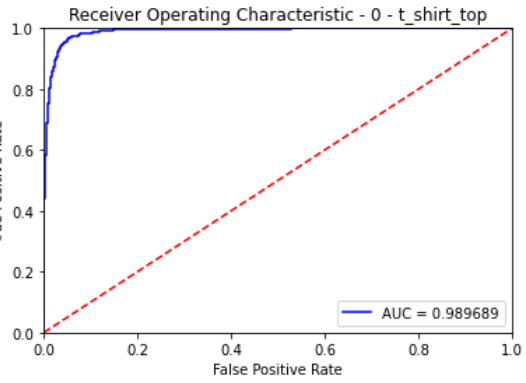
```
Test set ACCURACY: 89.2200
Test set PRECISION SCORE: 89.2287
Test set RECALL SCORE: 89.2200
Test set F1-SCORE: 89.2110
Test set F-MEASURE: 89.2243
```

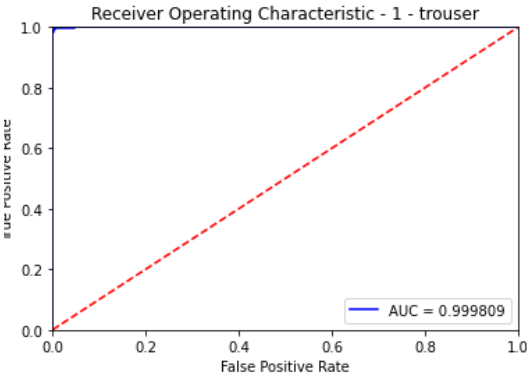
NOTA: 2 camadas de neurónios: (200,100) - os restantes hiperparâmetros da rede mantiveram-se constantes entre os diferentes testes. Sao eles: função de ativação = relu; solver = adam; max_iter=200.

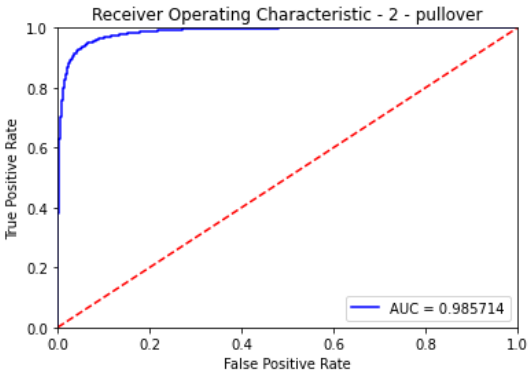
Foram testadas outras configurações com número de neurónios e camadas diferentes mas, de um modo geral, constatou-se que não existem diferenças significativas entre as diferentes configurações para a rede testada em termos de resultados. Apresenta-se a visualização dos resultados detalhados por classe, relativos à configuração de rede descrita:

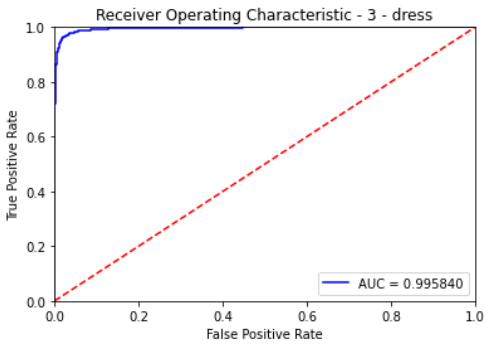


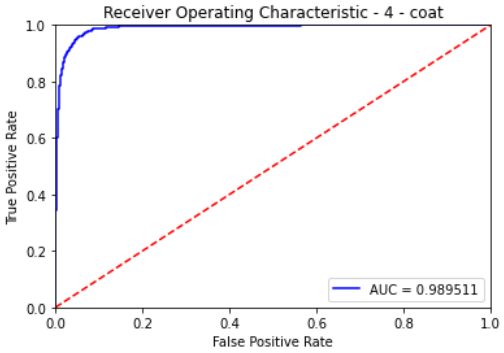


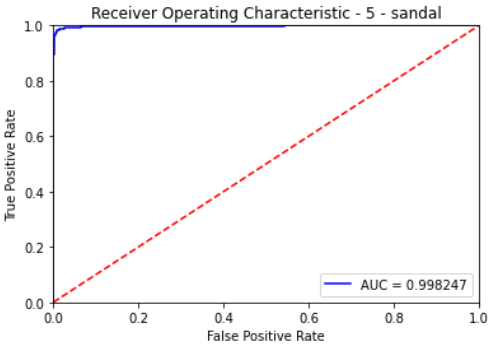


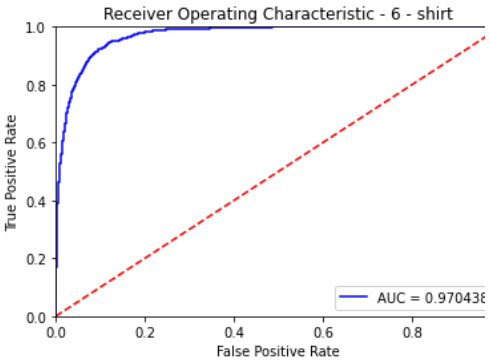


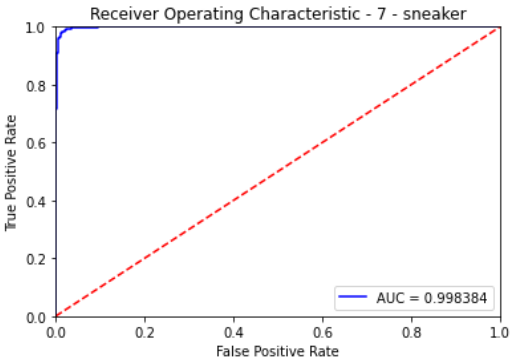


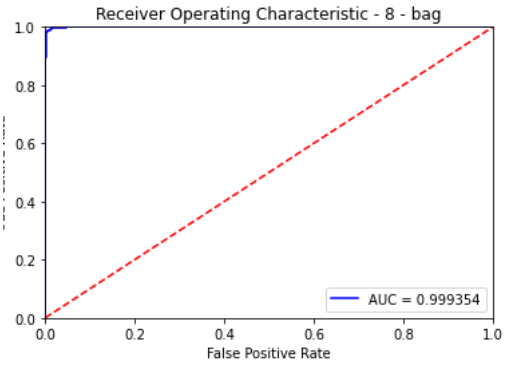


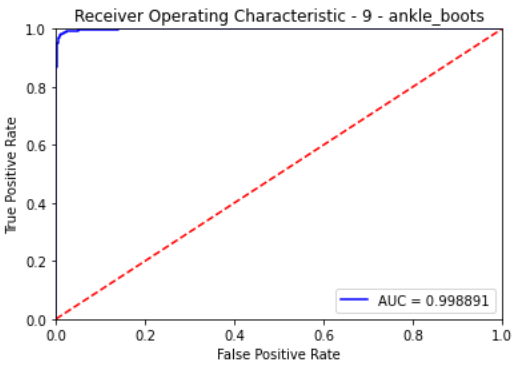












<u>Conclusões:</u> Através da análise dos gráficos, pode verificar-se que a classe 6 apresenta valores de AUC um pouco abaixo do das outras classes. Verifica-se ainda no Classification Report que para a classe 6 (shirt), os valores da Precisão e Recall são bastante mais baixos (na ordem dos 70%) do que para as restantes classes. No entanto, observámos para a classe 2 (pullover), o valor de Recall sendo superior à classe 6, apresenta também valor inferior à ordem dos 80%. Isto poderá significa que a rede não consegue distinguir, para alguns casos, se a peça é uma "shirt" ou um "pullover".

CONCLUSÕES

Este Projeto tinha como objetivo a implementação e validação de uma das técnicas de Inteligência Computacional, ao aplicá-la a um problema de classificação. Inicialmente foi pedido que aplicássemos o algoritmo estudado na fase II mas encontrámos diversas barreiras que não conseguimos contornar sendo que algumas delas foram incompatibilidades irreversíveis com as bibliotecas apesar destas estarem instaladas por completo e corretamente. Como essa situação aconteceu com diversos grupos, o professor Carlos Pereira sugeriu então que implementássemos a fase I em Python e que utilizássemos o algoritmo original PSO sem as variantes.

A implementação dos algoritmos de treino e de teste para o problema do Fashion Mnist realizou-se então com sucesso e encontram-se na pasta "plots" os plots de resultado da sua execução e resultados. Posto isso, seguimos para a fase de seleção de características para o modelo. Recorreu-se então ao algoritmo PSO para executar esta funcionalidade. Esta tarefa foi implementada com sucesso. Com este conjunto de características foi possível obter um bom resultado em termos de classificação do conjunto de teste.

No entanto, procurava-se uma melhoria do modelo a partir da fase seguinte. Este objetivo visava obter os melhores parâmetros de uma rede neuronalque por sua vez permitiria aumentar a performance de classificação. Esta última fase foi implementada, mas com pouca eficácia em termos de resultados, possivelmente devido ao facto de serem necessárias bastantes iterações (na ordem dos milhares) para que o algoritmo pudesse encontrar uma solução satisfatória. Ainda assim, o algoritmo foi testado e o programa correu durante 2 dias inteiros, mas sem chegar ao final da sua execução devido ao elevado tempo. Esta questão do elevado tempo de execução é aliás uma das limitações deste algoritmo e sentimos o impedimento de realizar baterias de testes em número ideal. Ainda assim, é um método preferível a outros métodos intensivos de procura da melhor solução, como por exemplo o método de Grid Search já que em relação a este é mais eficiente no método de pesquisa de soluções. Isso acontece porque o PSO permite a convergência para os parâmetros ótimos enquanto o método de Grid Search caracteriza-se precisamente por fazer pesquisa exaustiva, que é um processo ainda mais demorado e menos eficaz.

5. BIBLIOGRAFIA

Fashion Mnist (version 4, 2017). Zalando Research. Kaggle. Disponível em: https://www.kaggle.com/zalando-research/fashionmnist

How to Choose Evaluation Metrics for Classification Models. Data Science Blogathon, october 11, 2020. Disponível em: https://www.analyticsvidhya.com/blog/2020/10/how-to-choose-evaluation-metrics-forclassification-model/

Huilgol, P. Precision vs. Recall – An Intuitive Guide for Every Machine Learning Person. September 4, 2020. Disponível em: https://www.analyticsvidhya.com/blog/2020/09/precision-recall-machine-learning/

Algoritmo do BinaryPSO:

https://moodle.isec.pt/moodle/pluginfile.php/279774/mod_resource/content/2/aula06_PSO_20_21.pdf

Biblioteca Pyswarms: https://pyswarms.readthedocs.io/en/latest/index.html

Biblioteca SwarmPackage: https://pypi.org/project/SwarmPackagePy/