

notMinist Data Set  
Inteligência Computacional  
2021/2022  
Projeto - Fase I

# Índice

- Cap. 1: Descrição do caso de estudo e objetivos do problema;
- Cap. 2: Descrição da Implementação dos algoritmos;
- Cap. 3: Análise de Resultados;
- Cap. 4: Conclusões;
- Referências

## Cap. 1: Descrição do caso de estudo e objetivos do problema

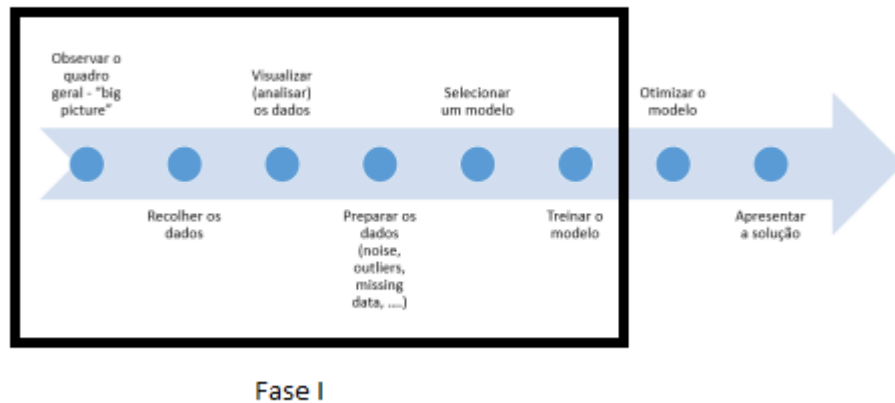


Fig. 1 – Etapas de um processo de Machine learning

Quanto à observação do quadro geral “big picture”.

O caso de estudo em questão tem por base o notMnist Data set, um dataset que é uma alternativa ao dataset tradicional Mnist para reconhecimento das letras do alfabeto, sendo mais complexo.

Este mesmo dataset foi desenvolvido por Yaroslav Bulatov.

Sendo que para o trabalho usamos a sua versão mais curta que contém as suas especificações:

o Número de exemplos: 18724

o Número de atributos: Imagem 28x28 (754)

o Número de classes: 10 classes (letras A-J)

Este dataset é disponibilizado através de um diretório raiz chamado notMNIST\_small que contém várias pastas com o nome de cada letra sendo que em cada uma destas pastas se encontram os exemplos de cada tipo de letra correspondente.

## Cap. 2: Descrição da Implementação dos algoritmos

Fizemos a implementação deste algoritmo em matlab.

Para isso tivemos que usar uma função em python que convertia o dataset inicialmente disponibilizado para um ficheiro .mat, sendo assim possível a sua utilização em matlab. Esta função python encontra-se no ficheiro de submissão.

A arquitetura da rede neuronal utilizada foi **MLP**

### **MLP-Redes de múltipla camada**

As redes de múltipla camada resolvem problemas não linearmente separáveis. Este algoritmo permite modificar os pesos de qualquer rede de modo que, sendo-lhe apresentado um padrão, ele tenha como output o padrão pretendido.

### **DEFINIÇÃO DAS ENTRADAS E SAÍDAS DA REDE**

Os dados para o treino e de teste usámos o dataset notMNIST\_small. Cada imagem tem uma única letra de tamanho 28x28, para o treino temos dois conjuntos em que o primeiro é uma matriz 784x18724 com todas as imagens (em grayscale) e outra matriz 10x18724 que corresponde às 10 classes, onde cada classe é representada por um valor de 0 a 9 que corresponde uma letra de A-Z respetivamente.

Para a saída temos valores de 1 a 10 que corresponde a cada letra.

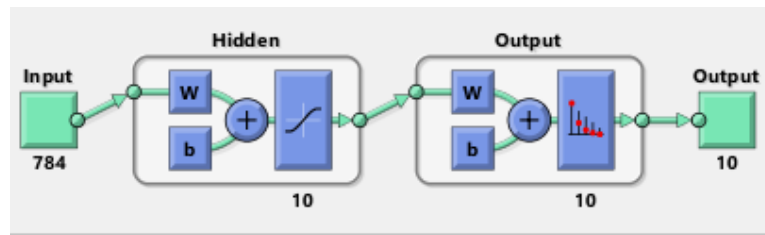
### Cap. 3: Análise de Resultados

Para analisar resultados partimos de um teste default, em que os seus parâmetros são os que o próprio matlab define por default para um tipo de rede patternet, sendo por isso o seu resultado bastante bom.

Os restantes testes são testes que tem o teste default por base sendo que são alterados certos parâmetros.

Os resultados destes mesmos testes estão indicados na página seguinte.

## Teste Default



Arquitetura da rede

Coefficiente de Aprendizagem=0.01

Épocas= 1000

Função de treino : traincg

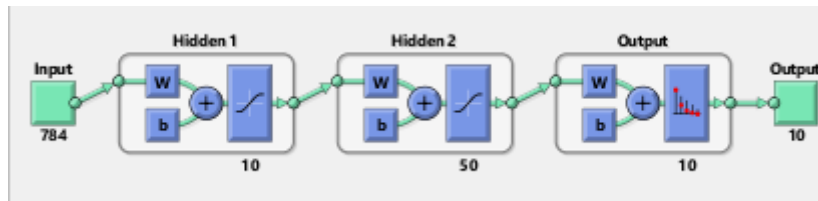
Tipo de rede: patternet

Confusion Matrix											
Output Class	1	2	3	4	5	6	7	8	9	0	
	1694 9.0%	36 0.2%	1 0.0%	17 0.1%	9 0.0%	12 0.1%	11 0.1%	44 0.2%	9 0.0%	39 0.2%	90.5%
	209 1.1%	1498 8.0%	5 0.0%	37 0.2%	30 0.2%	11 0.1%	22 0.1%	19 0.1%	11 0.1%	31 0.2%	80.0%
	63 0.3%	2 0.0%	1682 9.0%	23 0.1%	38 0.2%	9 0.0%	40 0.2%	3 0.0%	6 0.0%	7 0.0%	89.8%
	122 0.7%	18 0.1%	25 0.1%	1632 8.7%	7 0.0%	17 0.1%	13 0.1%	3 0.0%	10 0.1%	26 0.1%	87.1%
	129 0.7%	28 0.1%	72 0.4%	3 0.0%	1527 8.2%	28 0.1%	41 0.2%	11 0.1%	25 0.1%	9 0.0%	81.5%
	143 0.8%	2 0.0%	6 0.0%	13 0.1%	11 0.1%	1636 8.7%	8 0.0%	1 0.0%	30 0.2%	22 0.1%	87.4%
	120 0.6%	11 0.1%	55 0.3%	10 0.1%	13 0.1%	12 0.1%	1595 8.5%	5 0.0%	12 0.1%	39 0.2%	85.2%
	181 1.0%	16 0.1%	3 0.0%	9 0.0%	14 0.1%	22 0.1%	9 0.0%	1587 8.5%	21 0.1%	10 0.1%	84.8%
	187 1.0%	19 0.1%	12 0.1%	10 0.1%	30 0.2%	18 0.1%	11 0.1%	7 0.0%	1502 8.0%	76 0.4%	80.2%
	156 0.8%	8 0.0%	5 0.0%	9 0.0%	6 0.0%	17 0.1%	25 0.1%	1 0.0%	48 0.3%	1597 8.5%	85.3%
	56.4%	91.5%	90.1%	92.6%	90.6%	91.8%	89.9%	94.4%	89.7%	86.0%	85.2%
Target Class											10.2%
											9.5%
											20.0%
											12.9%
											18.5%
											12.6%
											14.8%
											15.2%
											19.8%
											14.7%
											14.8%

Matriz de confusão

Classes/Métricas	0	1	2	3	4	5	6	7	8	9
Precisão	0.90	0.80	0.90	0.87	0.82	0.87	0.85	0.85	0.80	0.85
Sensibilidade	0.56	0.91	0.90	0.93	0.91	0.92	0.90	0.94	0.90	0.86
F-meseaure	0.69	0.851	0.90	0.89	0.86	0.89	0.87	0.89	0.84	0.85

## Teste 1- Alteração da arquitetura



## Arquitetura da rede

Coefficiente de Aprendizagem=0.01

Função de treino : traincg

Tipo de rede: patternet

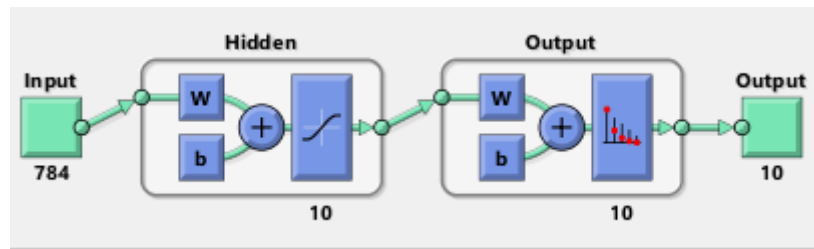
Confusion Matrix											
Output Class	1	2	3	4	5	6	7	8	9	10	
	1733 9.3%	13 0.1%	6 0.0%	9 0.0%	10 0.1%	9 0.0%	16 0.1%	23 0.1%	15 0.1%	38 0.2%	92.6% 7.4%
	152 0.8%	1563 8.3%	7 0.0%	49 0.3%	12 0.1%	9 0.0%	22 0.1%	15 0.1%	20 0.1%	24 0.1%	83.4% 16.6%
	101 0.5%	5 0.0%	1662 8.9%	7 0.0%	25 0.1%	10 0.1%	40 0.2%	5 0.0%	10 0.1%	8 0.0%	88.7% 11.3%
	128 0.7%	18 0.1%	9 0.0%	1654 8.8%	4 0.0%	17 0.1%	17 0.1%	2 0.0%	12 0.1%	12 0.1%	88.3% 11.7%
	167 0.9%	9 0.0%	80 0.4%	5 0.0%	1533 8.2%	18 0.1%	15 0.1%	10 0.1%	28 0.1%	8 0.0%	81.8% 18.2%
	123 0.7%	2 0.0%	3 0.0%	7 0.0%	12 0.1%	1673 8.9%	5 0.0%	12 0.1%	18 0.1%	17 0.1%	89.4% 10.6%
	167 0.9%	41 0.2%	39 0.2%	28 0.1%	16 0.1%	12 0.1%	1547 8.3%	14 0.1%	2 0.0%	6 0.0%	82.6% 17.4%
	175 0.9%	13 0.1%	5 0.0%	9 0.0%	16 0.1%	32 0.2%	11 0.1%	1576 8.4%	24 0.1%	11 0.1%	84.2% 15.8%
	180 1.0%	10 0.1%	11 0.1%	12 0.1%	21 0.1%	18 0.1%	11 0.1%	5 0.0%	1543 8.2%	61 0.3%	82.4% 17.6%
	97 0.5%	5 0.0%	8 0.0%	14 0.1%	13 0.1%	25 0.1%	12 0.1%	7 0.0%	31 0.2%	1660 8.9%	88.7% 11.3%
											Target Class
											1
											2
											3
											4
											5
											6
											7
											8
											9
											10
											Accuracy
											57.3% 42.7%
											93.1% 6.9%
											90.8% 9.2%
											92.2% 7.8%
											92.2% 7.8%
											91.8% 8.2%
											91.2% 8.8%
											94.4% 5.6%
											90.6% 9.4%
											90.0% 10.0%
											86.2% 13.8%

## Matriz de confusão

Classes/Métricas	0	1	2	3	4	5	6	7	8	9
Precisão	0.93	0.83	0.88	0.88	0.81	0.89	0.82	0.84	0.82	0.88
Sensibilidade	0.57	0.93	0.90	0.93	0.92	0.91	0.91	0.94	0.90	0.90
F-meseaure	0.70	0.87	0.90	0.89	0.86	0.90	0.87	0.89	0.86	0.90

## Teste 2- Alteração do coeficiente de aprendizagem

Diogo Dias Lopes– a2018019746  
Leonardo Marques - a2019123778



Arquitetura da rede

Coefficiente de Aprendizagem=0.1

Função de treino : traincg

Tipo de rede: patternet

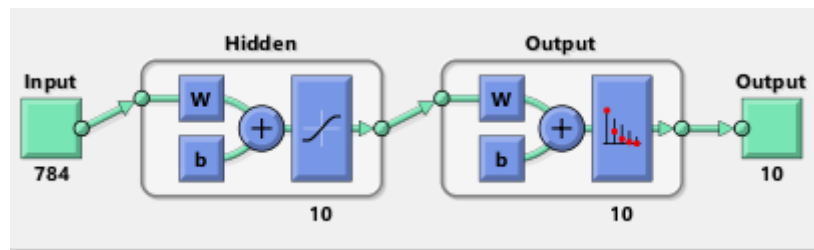
Confusion Matrix											
Output Class	1	2	3	4	5	6	7	8	9	10	
	1791 9.6%	1 0.0%	3 0.0%	3 0.0%	3 0.0%	7 0.0%	6 0.0%	27 0.1%	14 0.1%	17 0.1%	95.7% 4.3%
	121 0.6%	1657 8.8%	2 0.0%	29 0.2%	17 0.1%	7 0.0%	12 0.1%	12 0.1%	14 0.1%	2 0.0%	88.5% 11.5%
	59 0.3%	0 0.0%	1725 9.2%	2 0.0%	36 0.2%	5 0.0%	35 0.2%	6 0.0%	4 0.0%	1 0.0%	92.1% 7.9%
	72 0.4%	17 0.1%	5 0.0%	1715 9.2%	6 0.0%	10 0.1%	11 0.1%	5 0.0%	17 0.1%	15 0.1%	91.6% 8.4%
	105 0.6%	19 0.1%	53 0.3%	2 0.0%	1636 8.7%	15 0.1%	16 0.1%	5 0.0%	21 0.1%	1 0.0%	87.3% 12.7%
	76 0.4%	5 0.0%	4 0.0%	8 0.0%	10 0.1%	1730 9.2%	8 0.0%	5 0.0%	13 0.1%	13 0.1%	92.4% 7.6%
	120 0.6%	16 0.1%	39 0.2%	9 0.0%	9 0.0%	8 0.0%	1651 8.8%	4 0.0%	10 0.1%	6 0.0%	88.2% 11.8%
	130 0.7%	19 0.1%	3 0.0%	8 0.0%	2 0.0%	16 0.1%	6 0.0%	1667 8.9%	13 0.1%	8 0.0%	89.0% 11.0%
	143 0.8%	5 0.0%	6 0.0%	18 0.1%	20 0.1%	9 0.0%	5 0.0%	5 0.0%	1609 8.6%	52 0.3%	86.0% 14.0%
	91 0.5%	2 0.0%	5 0.0%	11 0.1%	6 0.0%	12 0.1%	3 0.0%	1 0.0%	31 0.2%	1710 9.1%	91.3% 8.7%
Target Class											

Matriz de confusão

Classes/Métricas	0	1	2	3	4	5	6	7	8	9
Precisão	0.95	0.88	0.92	0.91	0.87	0.92	0.88	0.89	0.86	0.91
Sensibilidade	0.66	0.95	0.93	0.95	0.93	0.95	0.94	0.96	0.92	0.93
F-meseaure	0.78	0.91	0.92	0.93	0.93	0.90	0.92	0.89	0.89	0.92



### Teste 3-Alteração da função de treino



### Arquitetura da rede

Coefficiente de Aprendizagem=0.01

Função de treino : traingd

Tipo de rede: patternet

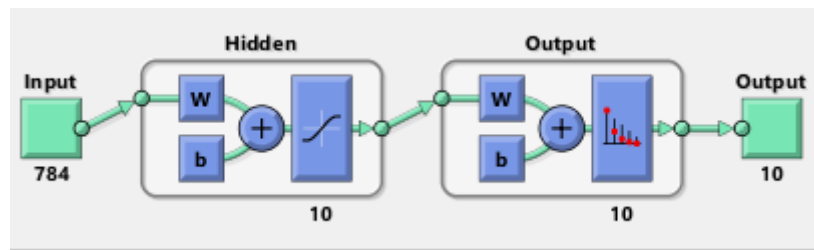
**Confusion Matrix**

Output Class \ Target Class	0	1	2	3	4	5	6	7	8	9
0	1755 9.4%	2 0.0%	1 0.0%	0 0.0%	1 0.0%	4 0.0%	85 0.5%	0 0.0%	22 0.1%	2 0.0%
1	623 3.3%	1221 6.5%	0 0.0%	13 0.1%	0 0.0%	3 0.0%	8 0.0%	0 0.0%	5 0.0%	0 0.0%
2	1218 6.5%	9 0.0%	573 3.1%	0 0.0%	2 0.0%	25 0.1%	44 0.2%	0 0.0%	2 0.0%	0 0.0%
3	421 2.2%	3 0.0%	0 0.0%	1416 7.6%	0 0.0%	7 0.0%	24 0.1%	0 0.0%	1 0.0%	1 0.0%
4	1519 8.1%	15 0.1%	1 0.0%	3 0.0%	290 1.5%	16 0.1%	19 0.1%	0 0.0%	10 0.1%	0 0.0%
5	578 3.1%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	1278 6.8%	8 0.0%	0 0.0%	6 0.0%	2 0.0%
6	395 2.1%	5 0.0%	1 0.0%	2 0.0%	1 0.0%	9 0.0%	1458 7.8%	0 0.0%	1 0.0%	0 0.0%
7	1406 7.5%	7 0.0%	0 0.0%	6 0.0%	1 0.0%	7 0.0%	32 0.2%	406 2.2%	6 0.0%	1 0.0%
8	888 4.7%	2 0.0%	0 0.0%	3 0.0%	4 0.0%	12 0.1%	19 0.1%	0 0.0%	939 5.0%	5 0.0%
9	1768 9.4%	2 0.0%	0 0.0%	5 0.0%	0 0.0%	6 0.0%	26 0.1%	0 0.0%	4 0.0%	61 0.3%
Summary	16.6% 83.4%	96.4% 3.6%	99.5% 0.5%	97.8% 2.2%	97.0% 3.0%	93.5% 6.5%	84.6% 15.4%	100% 0.0%	94.3% 5.7%	84.7% 15.3%

### Matriz de confusão

Classes/Métricas	0	1	2	3	4	5	6	7	8	9
Precisão	0.94	0.65	0.31	0.76	0.16	0.68	0.78	0.22	0.50	0.03
Sensibilidade	0.16	0.96	0.99	0.97	0.97	0.93	0.84	1	0.94	0.84
F-meseaure	0.28	0.77	0.47	0.85	0.25	0.78	0.81	0.36	0.65	0.05

#### Teste 4-Alteração da função de treino com mais iterações



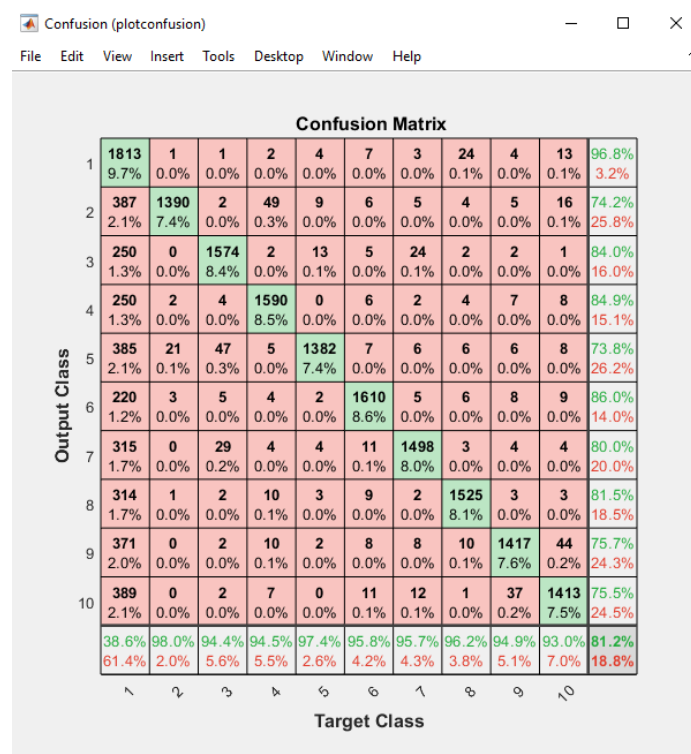
Arquitetura da rede

Épocas=10000

Coefficiente de Aprendizagem=0.01

Função de treino : traingd

Tipo de rede: patternet



Matriz de confusão

Classes/Métricas	0	1	2	3	4	5	6	7	8	9
Precisão	0.96	0.74	0.84	0.85	0.73	0.86	0.80	0.81	0.75	0.75
Sensibilidade	0.36	0.98	0.94	0.97	0.95	0.95	0.96	0.94	0.93	0.94
F-meseaure	0.55	0.84	0.88	0.89	0.83	0.90	0.87	0.87	0.84	0.83

## Cap. 4: Conclusões

Como já foi referido anteriormente o teste default serve como um “benchmark” teste para poder comparar os outros testes com este mesmo teste.

Em relação ao teste 1, foi alterada a arquitetura da rede neuronal passando agora a contar com duas hidden layers [10 50] em vez de uma [10]. Através dos resultados obtidos podemos concluir que a performance da rede subiu ligeiramente, concluindo assim também que o aumentar do número de hidden layers ajuda ao aumento da performance neste caso de estudo.

Em relação ao teste 2, foi alterado o coeficiente de aprendizagem de 0.01 para 0.1. Sendo que os resultados passaram de uma média de 85 para 90. Através dos resultados obtidos podemos concluir que a performance da rede subiu significativamente, concluindo assim também que o aumentar do número do coeficiente de aprendizagem ajuda ao aumento da performance neste caso de estudo.

Em relação ao teste 3, foi alterada a função de treino de traincg para traingd. Sendo que os resultados passaram de uma média de 85 para 50. Através dos resultados obtidos podemos concluir que a performance da rede desceu muito significativamente, sendo que por isso a função traingd não é das melhores funções de treino tendo em conta este caso de uso.

Por fim em relação ao teste 3, foi alterada a função de treino de traincg para traingd e aumentado o número de épocas de 1000 para 10000. Sendo que os resultados passaram de uma média de 85 para 81. Através dos resultados obtidos podemos concluir que a função de treino traincg realmente não é mesmo das melhores funções de treino para usar neste caso de estudo pois mesmo após 10x mais épocas os resultados ainda são piores.

## Referências

<http://yaroslavvb.blogspot.com/2011/09/notmnist-dataset.html>

<https://www.kaggle.com/lubaroli/notmnist>