Relatório de I.A.: Redes Neurais (Trabalho 5)

Cauê Baasch de Souza
João Paulo Taylor Ienczak Zanette

23 de Novembro de 2018

1 Resumo do projeto

Linguagem: Python 3.7

Biblioteca de Redes Neurais utilizada: sklearn [1].

2 Configuração dos experimentos

Os experimentos foram realizados tomando como base dois conjuntos de dados já disponibilizados pelo professor na plataforma Moodle, sendo um para treinamento da rede neural e outro para testes. Ambos os conjuntos são formados por tuplas no formato $(output, pixel_1, pixel_2, \ldots, pixel_n)$, em que $output$ é um número simbolizando a categoria esperada para a análise do conjunto de pixeis denotados por $pixel_i$.

O tratamento da rede neural foi separado em duas etapas: uma de treinamento, enviando à rede todas as tuplas do conjunto de treinamento em um grande lote, e outra para testes enviando as tuplas do conjunto de testes e, para cada teste, validando se a previsão da rede foi feita corretamente ou não. Em ambas as etapas, todos os pixeis foram normalizados para o intervalo $[0, 1]$ por uma divisão simples (Equação 1).

$$NormPixel_i = \frac{Pixel_i}{255} \tag{1}$$

A arquitetura da rede neural é formada por: uma camada de entrada com número de neurônios dependente do tamanho da entrada; um conjunto camadas intermediárias variando entre $\{1, 2\}$, cada uma com número de neurônios variando entre $\{10, 50, 100\}$; e uma camada de saída com 10 neurônios (um para cada classificação possível).

Quanto ao número de execuções, foi feito um treinamento e teste para cada configuração possível dentre o número de camadas intermediárias e o número de neurônios, totalizando 12 testes.

2.1 Quantos e quais experimentos foram feitos até chegar no resultado final

Vários (mentira, foram só uns 2 depois que funcionou com a lib).

	Actual class									
	0	1	2	3	4	5	6	7	8	9
0										
1										
2										
3										
4										
5										
6										
7										
8										
9										

Tabela 1:

2.2 Como foi o treinamento

Foi bem divertido, obrigado.

3 Resultados obtidos

3.1 Qual a taxa de acertos da rede

66% eu acho, por aí.

3.2 Matriz de confusão

3.3 Exemplos de objetos que foram mal-classificados pela rede

Sigh isso vai dar um trabalho…

3.4 Fatos que você achou interessante

Implementar backtracking é um belo cu mas foi legal.

Referências

[1] scikit-learn: Machine Learning in Python. `https://scikit-learn.org/`.