

# Trabalho II do Componente Curricular de Inteligência Artificial de 2019

## Redes Neurais

Éverton de A. Vieira e Gabriel H. Moro

17 de maio de 2019

### Introdução

#### Objetivo

O objetivo do trabalho é o desenvolvimento e uso de rede neural que resolve *Captchas*.

#### Problema Escolhido

O problema escolhido é o problema de quebra de *Captchas*. Um *Captcha* é uma que contém um texto que deve ser digitado para se ter permissão de acesso a ou um site. Para este trabalho serão utilizados *Captchas* simples, contendo quatro caracteres alfanuméricos em um fundo branco, excluindo os caracteres *O*, *I*, *0* e *1*.

### Desenvolvimento

Para o desenvolvimento do trabalho foi utilizada a linguagem *Python* 3.5+ e as bibliotecas *OpenCV*, para o processamento das imagens, *Keras* e *TensorFlow*, para a rede neural.

#### Base de dados

O banco de imagens contém 10000 *Captchas* simples descritos anteriormente<sup>1</sup>. Todas as imagens são nomeadas com o resultado esperado para

---

<sup>1</sup>Fonte: <https://drive.google.com/file/d/1jDa5UE1jSGr5apotQUzjdC9Wrkj0SKSj/view>

uso no treinamento da rede.

Primeiramente as imagens passam por um processo de limiarização para que só possuam tons de preto e branco para a detecção dos contornos dos caracteres utilizando a função *findContours()* da biblioteca *OpenCV*. A partir dos contornos gerados são salvas imagens dos caracteres separados, e caso haja sobreposição de caracteres isso é detectado pelas dimensões do contorno e corrigido dividindo a região contornada em duas, cada uma sendo um caractere. Cada imagem de caractere é salva em uma pasta com a *label* correspondente ao caractere e nomeado com a *label* e um número. Os caracteres separados são usados como entrada para a etapa de treinamento da rede

## Treinamento da rede

A rede treinada é uma rede neural convolucional simples, com duas camadas convolucionais (*Max pooling*), uma com 20 neurônios e outra com 50, que usam uma função de ativação *Rectified Linear Unit* ou *ReLU*, uma camada oculta completamente conectada com 500 neurônios, que também usa a função *ReLU*, e uma camada de saída com 32 neurônios, um para cada caractere possível, que usa a função *SoftMax*.

Inicialmente são carregadas as letras extraídas na primeira etapa que são redimensionadas para 20x20px e são armazenadas em uma lista de dados de treino da rede junto com a *label* correspondente, que é o nome da pasta na qual a imagem se encontra. A lista de dados de treinamento é separada em dados de treino (75%) e dados de teste (25%), a rede é construída e treinada com dez épocas e o modelo é salvo.

## Resultados

Os resultados obtidos são imagens nas quais os caracteres estão contornados e é exibido o caractere correspondente em cima do contorno. A taxa de acerto da rede foi de aproximadamente 100%, como mostrado na Figura 1. A Figura 2 mostra alguns exemplos de resultados obtidos com o modelo treinado.

```

26115/26115 [=====] - 22s 824us/step - loss: 0.2402 - acc: 0.9415 - val_loss: 0.0192 - val_acc: 0.9955
Epoch 2/10
26115/26115 [=====] - 19s 710us/step - loss: 0.0119 - acc: 0.9972 - val_loss: 0.0134 - val_acc: 0.9959
Epoch 3/10
26115/26115 [=====] - 17s 639us/step - loss: 0.0060 - acc: 0.9985 - val_loss: 0.0069 - val_acc: 0.9984
Epoch 4/10
26115/26115 [=====] - 17s 643us/step - loss: 0.0021 - acc: 0.9994 - val_loss: 0.1093 - val_acc: 0.9671
Epoch 5/10
26115/26115 [=====] - 17s 647us/step - loss: 0.0078 - acc: 0.9974 - val_loss: 0.0116 - val_acc: 0.9972
Epoch 6/10
26115/26115 [=====] - 16s 629us/step - loss: 0.0028 - acc: 0.9992 - val_loss: 0.0140 - val_acc: 0.9967
Epoch 7/10
26115/26115 [=====] - 17s 652us/step - loss: 0.0026 - acc: 0.9992 - val_loss: 0.0078 - val_acc: 0.9979
Epoch 8/10
26115/26115 [=====] - 17s 664us/step - loss: 0.0027 - acc: 0.9991 - val_loss: 0.0116 - val_acc: 0.9975
Epoch 9/10
26115/26115 [=====] - 17s 632us/step - loss: 0.0025 - acc: 0.9992 - val_loss: 0.0080 - val_acc: 0.9983
Epoch 10/10
26115/26115 [=====] - 17s 646us/step - loss: 0.0028 - acc: 0.9993 - val_loss: 0.0071 - val_acc: 0.9985

```

Figure 1: Erro e acerto da rede durante as épocas do treinamento

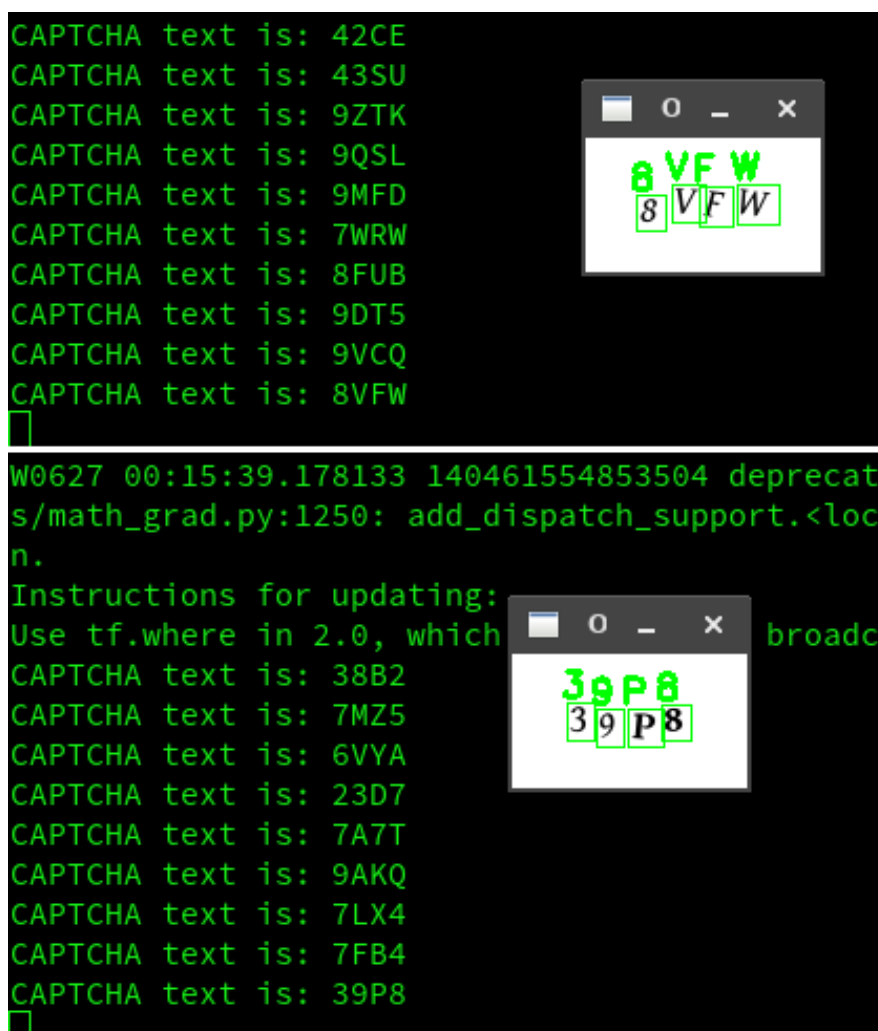


Figure 2: Exemplos de resultados obtidos