

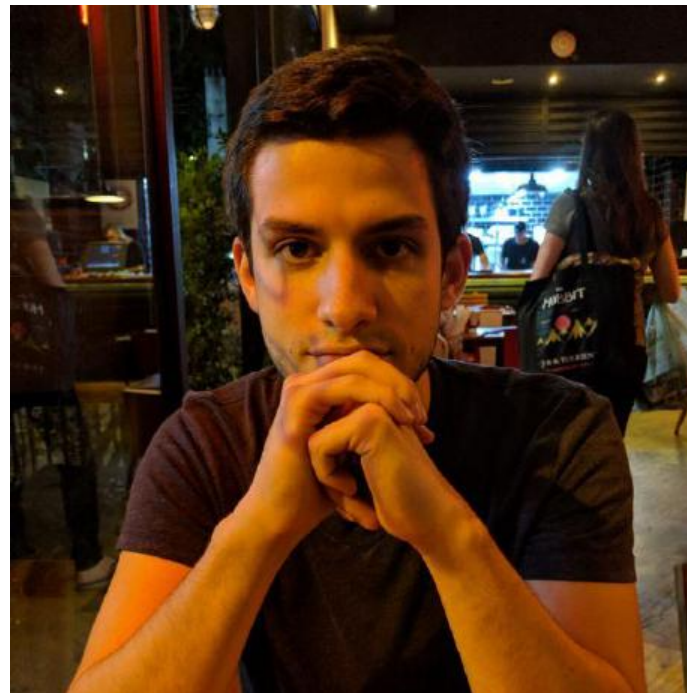
R DAY 2018: UMA API EXTENSÍVEL PARA QUEBRAR CAPTCHAS

Caio Lente (ctlente @curso-r.com)

2018-05-22

Sobre Mim

- Nome: Caio Truzzi Lente
- Idade: Mais do que parece
- Cidade: São Paulo, SP
- Graduação: Ciência da Computação no IME-USP
- Estágio: Platipus + Associação Brasileira de Jurimetria
- Ensino: Curso-R + R6



Decryptr

Motivação

- O decryptr nasceu com o objetivo de facilitar a quebra de captchas textuais de serviços públicos
- Muitas vezes os dados são públicos, mas não *acessíveis*, impedindo que um cidadão comum analise-os



- Talvez seja possível quebrar o captcha usando apenas transformações na imagem e um OCR padrão, mas isso acarreta dois problemas:
 - O tratamento das imagens dos captchas de cada fonte não pode ser generalizado para outra fonte
 - Transcrição de textos através de OCR não só tem uma taxa de acerto muito pequena, bem como não é escalável

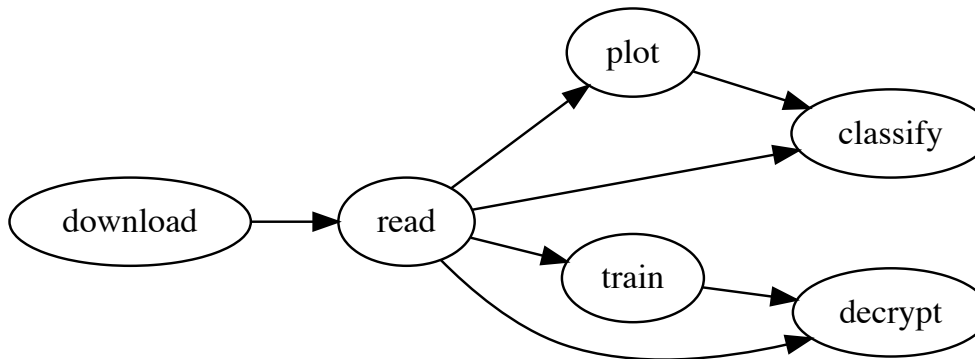
Keras

- Nossa solução para esses problemas foi o Keras, uma API de redes neurais de alto nível que pode rodar em cima de múltiplos backends como TensorFlow
- Usando este framework, Daniel Falbel¹ e Julio Trecenti criaram um uma rede neural que consegue aprender os padrões de famílias de captchas
 - Dado um conjunto de treinamento adequado, é possível quebrá-los com acerto e velocidade impressionantes.
- O pacote já vem com alguns modelos pré-treinados, mas o grande benefício de usar o `decryp`tr é que ele permite que qualquer usuário crie seus próprios modelos
 - Os captchas da família precisam ter o mesmo comprimento e a cor de cada letra não pode ser relevante para a quebra do captcha

[1] O Daniel inclusive chegou a contribuir para o desenvolvimento do *port* da biblioteca Keras para o R.

O Pacote

- São essencialmente dois módulos: um para quebrar captchas (`decrypt()`) e um para treinar novos modelos (`train_model()`)
- Temos funções auxiliares como `read_captcha()` para ler e plotar uma imagem, `classify()` para auxiliar na criação de bases de treino e `load_model()` para carregar modelos que o usuário já tenha treinado



Exemplo

```
library(decryptr)

file <- download_captcha("trt", path = "./img")

captcha <- read_captcha(file)

plot(captcha[[1]])

decrypt(file, model = "trt")

files <- download_captcha("trt", n = 3, path = "./img")

new_files <- classify(files, path = "./img")

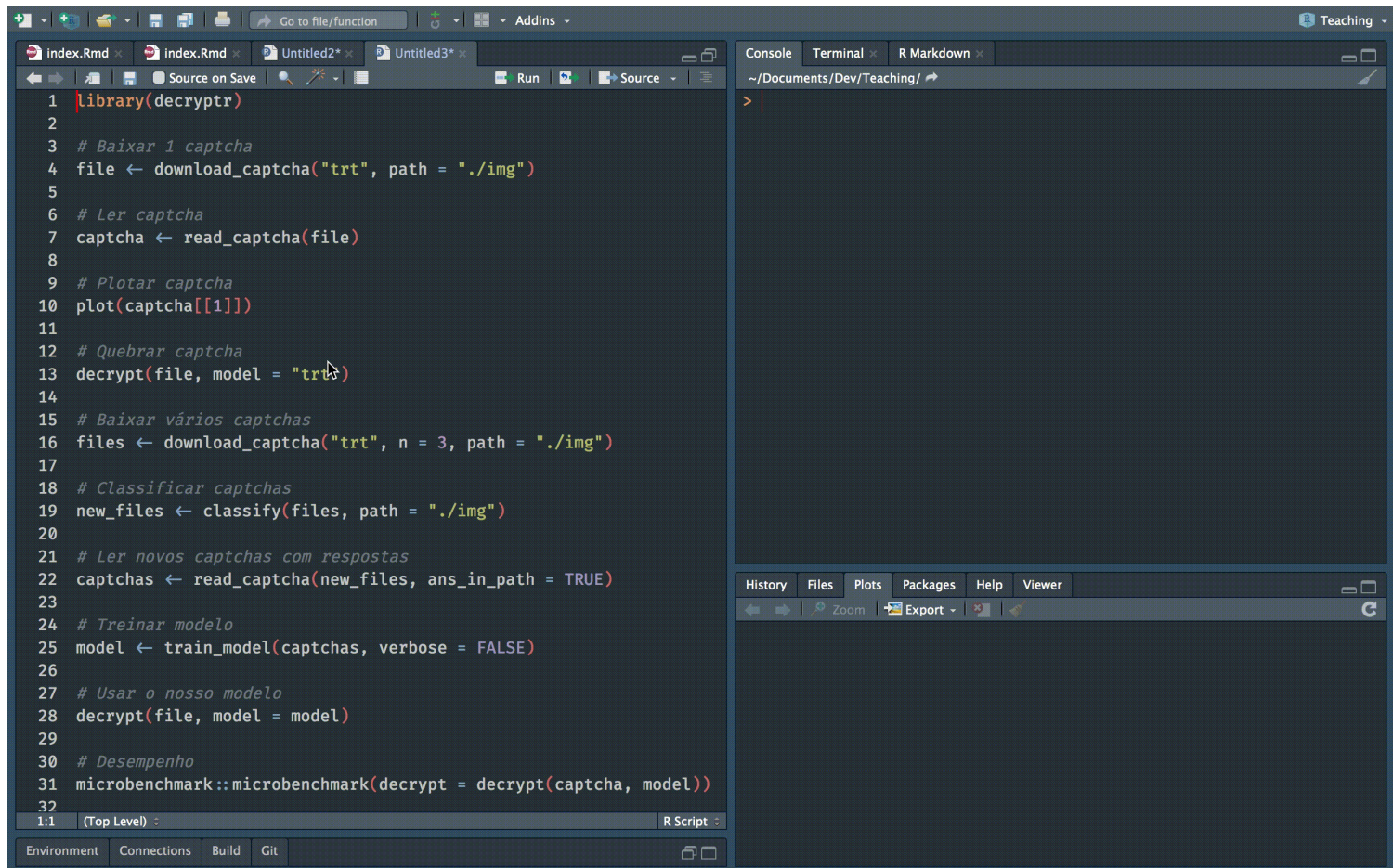
captchas <- read_captcha(new_files, ans_in_path = TRUE)

model <- train_model(captchas, verbose = FALSE)

decrypt(file, model = model)

microbenchmark::microbenchmark(decrypt = decrypt(captcha, model))
```

Demonstração



The screenshot displays the RStudio IDE interface. The main editor window on the left contains an R script with the following code:

```
1 library(decryptr)
2
3 # Baixar 1 captcha
4 file <- download_captcha("trt", path = "./img")
5
6 # Ler captcha
7 captcha <- read_captcha(file)
8
9 # Plotar captcha
10 plot(captcha[[1]])
11
12 # Quebrar captcha
13 decrypt(file, model = "trt")
14
15 # Baixar vários captchas
16 files <- download_captcha("trt", n = 3, path = "./img")
17
18 # Classificar captchas
19 new_files <- classify(files, path = "./img")
20
21 # Ler novos captchas com respostas
22 captchas <- read_captcha(new_files, ans_in_path = TRUE)
23
24 # Treinar modelo
25 model <- train_model(captchas, verbose = FALSE)
26
27 # Usar o nosso modelo
28 decrypt(file, model = model)
29
30 # Desempenho
31 microbenchmark::microbenchmark(decrypt = decrypt(captcha, model))
32
```

The right-hand pane of the IDE is divided into three sections: Console, Terminal, and R Markdown. The Console section shows the command prompt prompt `>` on a new line. The Terminal and R Markdown sections are currently empty. At the bottom of the IDE, there is a toolbar with buttons for History, Files, Plots, Packages, Help, and Viewer. Below this toolbar is a status bar showing the current line and column as `1:1 (Top Level)`, the active file as `R Script`, and tabs for Environment, Connections, Build, and Git.

Obrigado

ctlente@curso-r.com

ctlente.com

github.com/ctlente