Projeto de automatização na detecção de CAPTCHAs a partir de arquivos de áudio, utilizando técnicas de mineração de dados e aprendizado de máquina — Relatório I

Ewerton Carlos Assis

RA 21201820791

Abstract

O presente trabalho tem como finalidade apresentar um projeto de automatização na detecção de CAPTCHAs de tamanho fixo, a partir de arquivos de áudio, utilizando técnicas de aprendizado de máquina. Uma coleção com 200 arquivos de áudio para treinamento e outra coleção com 198 arquivos de áudio para validação foram utilizadas. Abordagens de aprendizado supervisionado foram comparadas, assim como estratégias de segmentação dos arquivos de áudio, com base em pesquisas feitas na literatura especializada. Como abordagens de classificação, utilizamos o *Random Forest Classifier* (RFC) e o *Support Vector Machines* ou *Support Vector Classification* (SVM). As técnicas de segmentação do áudio são baseadas em uma abordagem "ingênua" (segmentação baseada no número de caracteres a ser detectados) e uma abordagem baseada na energia detectada no áudio (*evenlope strength*), dividida em envelopes, a partir de picos de força. As abordagens utilizadas mostram-se promissoras, garantindo uma acurácia média de X% para o RFC e de Y% para o SVM, evidenciando a qualidade de ambas as alternativas para classificar arquivos de áudio, ainda que sob influência de ruídos. As abordagens de segmentação também influenciaram na acurácia, com uma melhora expressiva a partir da abordagem baseada em picos de energia e divisão por envelopes de força.

Keywords: Aprendizado de máquina, Classificação, Segmentação de arquivos de áudio, CAPTCHAs

1. Introdução

CAPTCHAs [1] baseados em pequenas mensagens de texto têm sido utilizados em diversos sistemas baseados na Web como alternativa para garantir a distinção entre a utilização destes sistemas por um robô (sistema automatizado) ou por um humano. Alguns destes sistemas de software baseado na Web que utilizam CAPTCHAs provêm meios de se obter o dado a ser fornecido (mensagem de texto) através de áudio, com a finalidade de garantir um design universal e que respeite pessoas com deficiências visuais. Diversas técnicas de análise destes áudios, a partir de abordagens da área de aprendizado de máquina, foram propostas a fim de analisar a acurácia das técnicas, ainda que sob forte influência de ruídos [2]. Também, neste contexto, faz-se necessário estabelecer técnicas para segmentação do áudio a fim de tirar sentido para cada segmento — ou em outras palavras, obter um caractere a partir de cada segmento, ainda que um ou mais segmentos apresentem ruídos.

O presente trabalho tem como finalidade apresentar um projeto de automatização na detecção de CAPTCHAs de tamanho fixo, a partir de arquivos de áudio, utilizando técnicas de aprendizado de máquina. Uma coleção com 200 arquivos de áudio para treinamento e outra coleção com 198 arquivos de áudio para validação foram utilizadas. Cada áudio representa uma combinação de 4 caracteres, dentre as seguintes 10 possibilidades: `'a'`, `'b'`, `'c'`, `'d'`, `'h'`, `'m'`, `'n'`, `'x'`, `'6'` e `'7'`. Cada uma das possibilidades de caracteres podem repetir em um único CAPTCHA/áudio.

As abordagens de aprendizado supervisionado *Random Forest Classifier* (RFC) e *Support Vector Machines* (SVM) — ou *Support Vector Classification* — foram utilizadas e comparadas a fim de se obter uma melhor acurácia entre as duas técnicas. Os áudios foram segmentados utilizando duas técnicas: uma abordagem "ingênua", que particiona o áudio em 4 segmentos fixos; e uma abordagem baseada na energia detectada no áudio (*evenlope strength*), dividida em envelopes, a partir de picos de força.

2. Análise preliminar dos arquivos de áudio

Os 398 arquivos de áudio foram analisados e as seguintes estatísticas foram obtidas: os arquivos apresentam um tamanho médio de 9,13 segundos, um tamanho mínimo de 7,91 segundos e tamanho máximo de 21,10 segundos — o que pode ser melhor analisado no histograma da Figura 1 —; existe uma maior ocorrência da letra `'a'` entre os arquivos de áudio, embora os

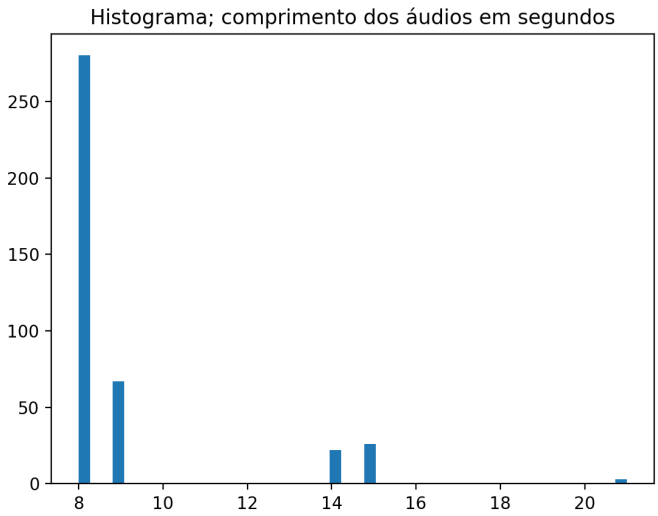

Figure 1: Histograma feito a partir da discretização dos comprimentos dos áudios em segundos — coleções de treinamento e validação

caracteres sejam relativamente bem distribuídos, conforme pode ser visto no histograma da Figura 2.

2.1. Parâmetros para definição das abordagens de segmentação dos arquivos de áudio

A fim de melhor definir os parâmetros para as abordagens de segmentação dos arquivos de áudio, foram analisados 39 arquivos (aproximadamente, 10% do total da base) com a finalidade de extrair conhecimento sobre estes e definir estratégias de segmentação. Os dados foram analisados a partir da biblioteca para a linguagem Python `librosa` [3].

A partir do *waveplot*, renderizado pela biblioteca `librosa` [3], conseguimos verificar que os segmentos de áudio estão bem acentuados, principalmente os pontos de silêncio, favorecendo a segmentação baseada em uma abordagem que particiona o áudio em 4 segmentos fixos. No entanto, como os áudios não têm um comprimento fixo em segundos e, assim, o áudio pode apresentar variação no comprimento de cada parte significante (o som do caractere) ou não ter uma distribuição pré-estabelecida para cada som de caractere no áudio, essa técnica de segmentação, que aqui chamamos de *segmentação*

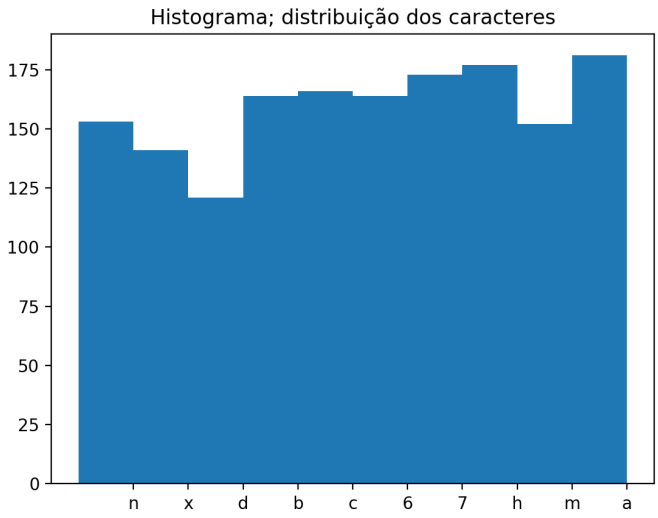

Figure 2: Histograma feito a partir da contagem dos caracteres presentes nos nomes de cada arquivo — coleções de treinamento e validação

"ingênua", pode não ser a ideal; sobretudo quando os áudios podem apresentar ruídos do ambiente onde foram obtidos. As Figuras 3 e 5 apresentam exemplos de distribuição das frequências em dois arquivos de áudio distintos.

Uma outra abordagem desenvolvida baseia-se na energia detectada no áudio (*evenlope strength*), dividida em envelopes, a partir de picos de força ou energia. A biblioteca utilizada (`librosa` [3]) apresenta uma solução customizada que auxilia na resolução deste tipo de problema, através da função `'librosa.onset.onset_detect'`. As Figuras 4 e 6 apresentam exemplos de distribuição de energia e espectrograma de força em dois arquivos de áudio distintos.

Ambas as abordagens de segmentação dos arquivos de áudio foram comparadas para cada tipo de classificador.

3. Performance no uso do *Random Forest Classifier*

Maiores detalhes no segundo relatório.

4. Performance no uso de *Support Vector Machines*

Maiores detalhes no segundo relatório.

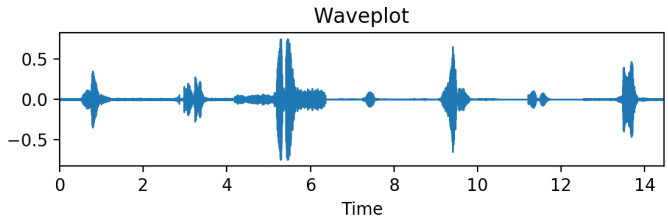

Figure 3: *Waveplot* criado a partir do arquivo de áudio para validação `'66m6.wav'`

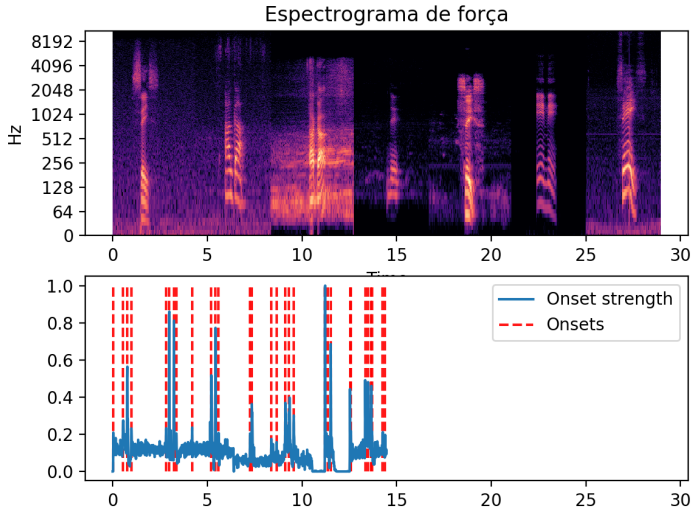

Figure 4: Espectrograma de força criado a partir do arquivo de áudio para validação `'66m6.wav'`

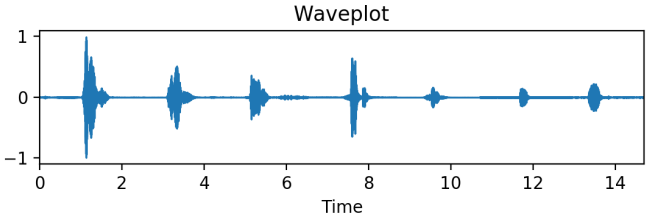

Figure 5: *Waveplot* criado a partir do arquivo de áudio para validação `'xxdc.wav'`

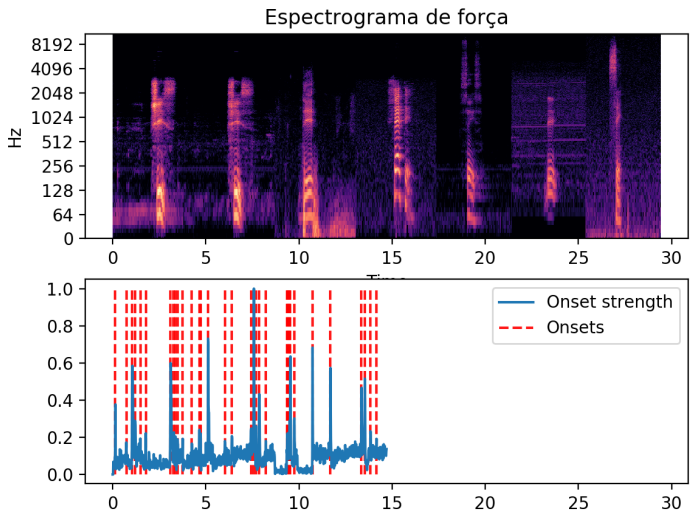

Figure 6: Espectrograma de força criado a partir do arquivo de áudio para validação `'xxdc.wav'`

5. Resultados

Com base nos testes e análises realizadas, ambas as abordagens de classificação mostram-se promissoras, garantindo uma acurácia média de X% para o RFC e de Y% para o SVM, evidenciando a qualidade de ambas as alternativas para classificar arquivos de áudio, ainda que sob influência de ruídos. As abordagens de segmentação também influenciaram na acurácia, com uma melhora expressiva a partir da abordagem baseada em picos de energia e divisão por envelopes de força.

Maiores detalhes no segundo relatório.

6. Conclusões e trabalhos futuros

No presente trabalho, não foi analisado os parâmetros de configuração dos classificadores utilizados, o que poderia comprometer a qualidade dos resultados obtidos. Em geral, comparando com outros resultados citados, as soluções propostas apresentam-se como alternativas promissoras e que poderiam ser utilizadas em ambientes reais para classificação e subsequente inferência de caracteres em um CAPTCHA.

Maiores detalhes no segundo relatório.

7. Referências

[1] L. von Ahn, M. Blum, J. Langford, Telling humans and computers apart automatically, Communications of the ACM 47 (2004) 57–60.

[2] J. Tam, S. Hyde, J. Simsa, L. V. Ahn, Breaking audio captchas, NIPS'08 Proceedings of the 21st International Conference on Neural Information Processing Systems (2008) 1625–1632.

[3] Librosa - librosa 0.6.0 documentation, `http://librosa.github.io/`, 2013. Acessado em 26 de Julho de 2018.