Reconhecimento de caracteres por Inteligência Artificial

Mateus Siqueira Carneiro
Universidade Federal de São Paulo
Instituto de Ciência e Tecnologia
São José dos Campos, São Paulo
mateussiqueiracarneiro@outlook.com

Matheus Gomes de Paula
Universidade Federal de São Paulo
Instituto de Ciência e Tecnologia
São José dos Campos, São Paulo
matheusgomes062@gmail.com

***Abstract* — Nesse paper iremos iremos propor o uso de um método analítico para o reconhecimento de palavras manuscritas. Esse método é proposto para o problema do reconhecimento de escrita cursiva offline. Primeiro a obtenção de parâmetros globais, como ângulo de inclinação, linhas de base, largura da linha e altura. Segundo o tratamento da imagem pelo método de segmentação combinando gray scale e informação binária. Terceiro, o uso do Tesseract-OCR da Google para o treinamento e reconhecimento da imagem. Finalmente ocorre a decisão do melhor caractere a partir da junção das informações do Tesseract.**

Palavras-Chave—Inteligência Artificial, Reconhecimento de padrões, Escrita, OCR, HMM, Hidden Markov Models, Tesseract.

I. INTRODUÇÃO E MOTIVAÇÃO

Apesar de estarmos na era digital, a escrita ainda é uma das atividades mais importantes. Muito do nosso conhecimento ainda se encontra descrito cursivamente em papel. Tais métodos de escrita utilizam diferentes caligrafias variando de pessoa para pessoa, o que pode se mostrar um problema para muita das transcrições para o meio digital, com riscos de não se entender o que está escrito. Propomos o uso de uma Inteligência Artificial para resolver esse problema.

II. OBJETIVOS

Neste trabalho focaremos em reconhecer escritas cursivas utilizando Inteligência Artificial. Esperamos que a longo prazo possamos conseguir transcrever para um arquivo digital qualquer texto manuscrito com uma taxa de erro segura. De forma que mesmo o texto possuindo rasuras ele possa ser identificado corretamente dentro da taxa de acerto. Assim confiamos que ao final do curso teremos os conhecimentos necessários para desenvolver um software de Inteligência Artificial para digitalizar escritas manuais que seja relativamente seguro e eventualmente rentável.

III. METODOLOGIA EXPERIMENTAL

Utilizaremos a linguagem de programação Python, sendo utilizado em conjunto bibliotecas como scikit-learn, pytesseract, pillow, numpy, openCV.

Faremos o uso da ferramenta de Inteligência Artificial de reconhecimento de caracteres (Optical Character Recognition - OCR) da Google, o Tesseract-OCR para reconhecer os caracteres. Também será utilizado o Leptonica[8], uma biblioteca open source contendo software que é comumente utilizado para processamento e análise de imagens, este em específico será utilizado para o treino do Tesseract, juntamente com o OCR-D[9], este facilitará o treinamento ao permitir o uso de um MAKEFILE para realizar o treinamento, este nos auxilia automatizando toda a criação dos arquivos necessários para iniciar o treinamento. Iremos utilizar como base o paper proposto por Nafiz Arica e T. Yarman-Vural "Optical Character Recognition for Cursive Handwriting"[1] pois apresenta uma abordagem factível e explicativa sobre o processo. Assim, utilizaremos uma estratégia analítica, que utiliza uma abordagem de "baixo para cima", começando da letra até a palavra. Uma segmentação da palavra se faz necessária para essa

estratégia. Com isso reduzimos o reconhecimento para caracteres individuais.

O Tesseract-OCR apesar de ser uma boa ferramenta possui limitações, como visto no artigo mencionado anteriormente quanto no paper de Ray Smith, "An Overview of the Tesseract OCR Engine"[2], assim teremos uma etapa de pré processamento onde trataremos a imagem com o uso de níveis de cinza, normalização do ângulo e da linha de base, e por fim, aplicaremos então o uso de normalização binária da imagem. A nosso favor temos a base do NIST que já vem nesse formato binário.

IV. BASES

Utilizaremos como base um banco público de imagens fornecido pelo Instituto Nacional de Padrões e Tecnologia (National Institute of Standards and Technology - NIST). O Special Database 19 (nome do banco) contém todo o corpo de materiais de treinamento do NIST para documentos impressos e reconhecimento de caracteres. Ele possui Formulários de Amostra manuscrita de 3600 escritores, 810.000 imagens com classificações verificadas à mão. As imagens possuem dígitos separados, letras maiúsculas e minúsculas e campos de texto livre.

Apesar do alto potencial que a base oferece, será necessário alterações na mesma para entrar em conformidade com os requisitos para o treino de forma que será necessário transformar as imagens do tipo "*.png*" em "*.tif*". Não só isso mas será necessário recortar as imagens para melhorar o treino.

V. TÉCNICAS E MEDIDAS DE AVALIAÇÃO

Para obter uma medida que é independente do tamanho do texto o número de erros é em geral normalizado para o tamanho do conteúdo esperado (Texto Original). O quociente entre o número de erros e o tamanho do texto é conhecido como taxa de erro.

A taxa de erro é geralmente calculada em dois níveis diferentes:

- Taxa de erro de letra (Character Error Rate - CER).
- Taxa de erro de palavra (Word Error Rate - WER).

Vale lembrar que a taxa de erros em palavras é geralmente maior do que a taxa de erros em letras. Por exemplo, uma taxa de erro de caracteres como 10% significa que aproximadamente metade das palavras de 6 letras conterão pelo menos um caractere errado. Experimentos sugerem que humanos são relativamente intolerantes a erros seguindo a seguinte lógica: uma acurácia por palavra abaixo de 85% leva a um baixo rendimento de produtividade se correção manual é aplicada, isto comparado a escrever tudo do zero.

De acordo com Rose Holley no seu artigo "How Good Can It Get? - Analysing and Improving OCR Accuracy in Large Scale Historic Newspaper Digitisation Programs"[4] uma boa medida de OCR significa acertar em torno de 98 a 99% das palavras, enquanto que abaixo de 90% é ruim. Portanto, adotaremos essa medida de avaliação.

Sendo assim, a avaliação será inspirada no modelo proposto pelo artigo de Rose Holley, em conjunto usaremos um modelo de avaliação semelhante ao proposto por Amalia R. e Venu G. no paper "Handwritten CAPTCHA: Using the difference in the abilities of humans and machines in reading handwritten words"[10].

O treinamento usou as 10 primeiras páginas da base, ja avaliação se dará da seguinte forma: Após o treinamento haverá um teste de um dos textos do banco de dados do NIST, este será escolhido aleatoriamente, excluindo-se as já usadas para o treinamento, de forma que possamos obter uma abordagem mais próxima de um contexto real. Após, teremos uma verificação manual do texto, onde verificaremos os acertos e os erros. A verificação se dará em várias categorias, todas com o propósito geral de medir a acurácia do treino, estas sendo: número de caracteres, caracteres correspondentes, número de palavras e palavras correspondentes.

Número de caracteres: Caso o resultado obtido pelo teste possua o mesmo número de caracteres que o texto original concluiremos como um resultado positivo neste aspecto. Será considerado um resultado mediano caso o número de caracteres ultrapasse em até 2% do texto original, tanto para cima quanto para baixo. Menos que isso será considerado um desempenho ruim ou péssimo.

Caracteres correspondentes: Será feito uma comparação manual dos caracteres do treino sendo avaliado posições correspondentes, obtendo assim uma Taxa de erro de letra, caso essa taxa seja menor que 10% consideraremos como um resultado positivo nesse critério, sendo que caso a taxa seja menor que 5% o resultado não só será considerado positivo como será considerável ótimo. Caso o resultado da taxa seja maior que 10% será considerado um desempenho ruim ou péssimo.

Número de palavras: Será analisado a quantidade de palavras resultantes, se elas se adequarem ao número de palavras do texto original será considerado um resultado positivo ou ótimo, caso obtivemos 5% de erro, tanto para mais quanto para menos, será considerado ruim, e maior que isso será considerado péssimo. Não há espaço para resultados medianos pois erro de palavras causa má informação ou até mesmo outro contexto.

Palavras correspondentes: Será avaliado quantas palavras existem no texto original que foram encontradas no teste, caso a taxa de erro seja menor que 5% será

considerado positivo ou ótimo, caso seja menor que 10% será considerado positivo e mediano e caso seja maior que 10% será considerado péssimo.

Tais avaliações são necessárias para medir a acurácia do algoritmo, de forma que esteja dentro dos parâmetros descritos por Rose Holley como mencionado anteriormente.

VI. TREINO E EXPERIMENTOS

Primeiramente baixamos dois pacotes do NIST, o "*hsf_page.zip*", que contem todas as páginas escritas a mão que serão usadas para o treino e para a validação, e o "*by_write.zip"* que possui todas as letras recortadas e divididas por escritor. O treino foi feito da seguinte forma, pegávamos o banco de dados do *by_write* selecionamos um escritor e adicionamos na pasta "*ground-truth*" do OCR-D, este algoritmo em específico auxilia e facilita a construção de arquivos que o Tesseract usa para o treino. Para usar o programa no entanto era necessário que os arquivos do *ground-truth* fossem do tipo "*.tif*" o NIST possui apenas imagens "*.png"* assim, desenvolvemos um algoritmo em python que transforma todos as imagens no formato necessário. Além disso, o *ground-truth* necessita que haja um arquivo de texto do tipo "*.gt.txt*" correspondente por imagem a ser usada no treino. Assim foi adicionado essa funcionalidade no próprio arquivo python, de forma que ele tanto transforma em "*.tif*" quanto cria um arquivo "*.gt.txt"* com o texto correspondente de cada imagem. Ao fazer os treinos, no entanto, ocorreram diversos problemas e a taxa de erro do treino era de 100%, portanto inviável a utilização.

Assim, refizemos o treino dessa vez ao invés de utilizar as letras disponibilizadas pelo NIST nós cortamos as imagens de texto, disponíveis no arquivo *hsf_page,* em linhas e palavras singulares. Dessa forma a eficiência do treino aumentou consideravelmente chegando a aproximadamente 3% de taxa de erro no treino.

Aqui estará disposto os passos do treino, após as imagens estarem tratadas e seus textos correspondentes estarem criados:

1. **Make training:** Esta é a etapa inicial, onde o comando "*make training"* irá realizar todas as próximas etapas automaticamente através do nosso programa OCR-D.
2. **Linebox**: Nessa etapa são gerados os quadrados que dividem as letras e palavras para cada arquivo *.tif*. É gerado um arquivo correspondente do tipo "*.box*"
3. **Unicharset:** Nessa etapa é criado um arquivo do tipo *unicharset* onde fica disposto a informação para cada simbolo (unichar).
4. **Traineddata:** Faz o inicio do arquivo *traineddata* que será usado para reconhecer as imagens no Tesseract. Esse arquivo é feito a partir dos *unicharsets.*
5. **Processamento de imagem + box file:** O tesseract é inciado para processar a imagem junto de seus *box files* para criar o conjunto *training data* necessário.
6. **Treino no conjunto training data:** É feito o treinamento no conjunto training data.
7. **Combina:** Nessa última etapa é combinado os arquivos para gerar um *trainneddata* final que poderá ser usado para o reconhecimento efetivo.

Após esses passos principais é colocado no diretório principal do tesseract o arquivo *trainneddata* gerado e é possível colocar em prática o treino.

Sobre o Tesseract:

O tesseract é uma poderosa ferramenta para o reconhecimento de caracteres. Inicialmente desenvolvido pela Hewlett Packards Labs o tesseract-ocr foi aberto em forma Open Source em 2005 numa parceria com a Universidade de Nevada. A partir de então tem sido desenvolvido junto com a Google e outras dezenas de contribuidores.

Para o trabalho foi usado o tesseract 4.00, este mostra uma maturidade em relação a sua versão anterior (3.00) pois implementa em seu core uma Rede Neural Recorrente (RNN) do tipo *Long Short Term Memory* (LSTM) como engine de reconhecimento.

Especificações Técnicas da Máquina

Processador: Intel® Core™ i5-5200U Dual Core 2.2 GHz

Sistema Operacional: Ubuntu 18

Tela: Tela LED HD Widescreen, com resolução de 1366 x 768

Cache: 3 mb L3

Memória RAM: 8 GB DDR3L 1600 MHz

HD: 1 TB 5400 RPM + SSD M2 2280 240gb

PLACA DE VÍDEO: NVIDIA® GeForce® 920MX Graphics 2 GB gDDR3 Dedicada

Aqui estão dispostos os treinos realizados:

Treino 01	
Página selecionada	hsf_page/hsf_0/f0001_41
Iterações	10000
Média rms	0,99%
Delta	0,74%
Char train	2,73%
Word train	9,04%
Skip ratio	0,00%
Imagens	52
Tempo de treino	50 minutos
Error rate	2,73%

Treino 02	
Página selecionada	hsf_page/hsf_0/f0000_14
Iterações	10000
Média rms	2,72%
Delta	4,78%
Char train	23,78%
Word train	35,97%
Skip ratio	0,00%
Imagens	52
Tempo de treino	37 minutos
Error rate	10,02%

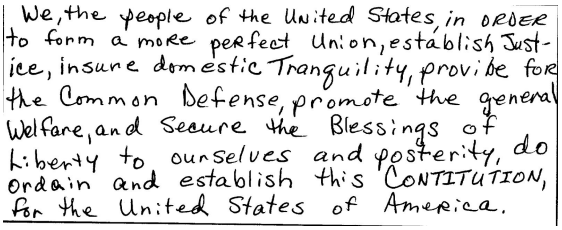
Treino 03	
Página selecionada	hsf_page/hsf_0/f0002_01
Iterações	10000
Média rms	5,57%
Delta	15,49%
Char train	83,72%
Word train	87,91%
Skip ratio	0,00%
Imagens	56
Tempo de treino	25 minutos
Error rate	83,72%

Treino 04	
Página selecionada	hsf_page/hsf_0/f0003_42
Iterações	10000
Média rms	2,20%
Delta	3,80%
Char train	16,95%
Word train	27,90%
Skip ratio	0,00%
Imagens	52
Tempo de treino	40 minutos
Error rate	16,95%

Treino 05	
Página selecionada	hsf_page/hsf_0/f0004_09
Iterações	10000
Média rms	1,34%
Delta	2,33%
Char train	7,08%
Word train	15,93%
Skip ratio	0,00%
Imagens	52
Tempo de treino	36 minutos
Error rate	0,07%

Para o Treino 01 verificamos que pelas letras serem mais uniformes e melhor segmentadas a taxa de erro foi bem abaixo da média dos outros treinos. Da mesma forma, o Treino 05 possui a menor taxa de erro dos treinos apresentados.

Rodando todos os treinos para uma imagem aleatoriamente selecionada do database do NIST.



Obtivemos os seguintes resultados:

Utilizando o Treino 01

Amerinse efea Wn the wOcte, pommode gen m
to the in ini wevire COSSITTtN thise
We, CONSo an CONSTiiie the
the COraS to Cure irce thie ei
COSSy Defre the com tor t
an scpe to We, CeONONe ane pefale gen
CONii, ane pOsuuooin th AmrTUTION
geeca. W,Pite prooone pod poommmmoic

Utilizando o Treino 02

Bloean. prot westIUiisOch to pror
dp ppr Blessinh pprrrprprprprprp tund
poeciiiiic pprrprprrrprprprrrpr po
pro Wot WessTbieh our pom
We te h te me ore
p oom o csie ppr prom pp
prannitoioiiiy cotic,
prr pernqi,,,, pepooooh

Utilizando o Treino 03

s s s s s
s o s s s
s s s
s s s s s
s
s coro tn gesquabli s UPnids to
s s
srr s

Utilizando o Treino 04

Stat perot pCONSTIITIU Ttee Ue gand
s toe feste, oumeca. te
e, UTION Se es fiION Uni
gene United fo iION thie prode
or ftheeeeee onns,ish
ge Untns, s Aemesh tnie Unite ge
Ste T TtONe Stere,
gesss St f America

Utilizando o Treino 05

Lerit. emeie, es CON Tedbt Jt n o oodaue,
Win m f NTe pe er oluih Ue,
re pites otore po tntiied niore for
oahis res oln of rT pSte tanid Und
efrho eth UJUntiqh Jes, Jess, ors a
or oedrn fo sefuie Ued prmt toe
o e prrdetoe
ooiy o

Utilizando o *trainneddata* nativo do Tesseract

We The people of the United States in pLOEGe
to form amore perfect Union, establish wust-
jee, insure dem esti. Tranguility, prov: he for
tre. Comm on hefense promote tHve. qrenera\
Welfare ana Seaure Whe Blessings of
Ls hewrty tp ourselves and posterty, do
ordain and establish this CONTITU TION,
fon the United States of Amepriea,

Texto original

We, the people of United States, in order
to form a more perfect Union, establish Just-
ice, Insure domestic Tranquility, provide for
the Common Defense, promote the general
Welfare, and secure the Blessings of
Liberty to ourselves and posterity, do
ordain and establish this CONSTITUTION,
for the United States of America.

Temos os seguintes resultados por treino:

Treino 01	
Imagem selecionada	f0081_26_complete_text
Número de caracteres	264
Caracteres correspondentes	18
Número de palavras	52
Palavras correspondentes	9
Quase palavras	1

Treino 02	
Imagem selecionada	f0081_26_complete_text
Número de caracteres	216
Caracteres correspondentes	21
Número de palavras	36
Palavras correspondentes	2
Quase palavras	0

Treino 03	
Imagem selecionada	f0081_26_complete_text
Número de caracteres	72
Caracteres correspondentes	0
Número de palavras	30
Palavras correspondentes	1
Quase palavras	0

Treino 04	
Imagem selecionada	f0081_26_complete_text
Número de caracteres	204
Caracteres correspondentes	0
Número de palavras	42
Palavras correspondentes	2
Quase palavras	4

Treino 05	
Imagem selecionada	f0081_26_complete_text
Número de caracteres	222
Caracteres correspondentes	18
Número de palavras	51
Palavras correspondentes	1
Quase palavras	0

ORIGINAL	
Imagem selecionada	f0081_26_complete_text
Número de caracteres	327
Caracteres correspondentes	327
Número de palavras	52
Palavras correspondentes	52
Quase palavras	0

Análise dos Resultados:

Ao analisar os resultados percebemos a necessidade de colocar mais um dado, o de “quase palavras” onde após uma análise é determinado quantas palavras é possível identificar que sofreram alterações por estarem com caracteres errados.

Percebemos que o treino que mais se aproximou ao número de caracteres foi o Treino 1, obtendo 80% do

número de caracteres do texto original, assim como 34,6% dos caracteres correspondentes, 100% do número de palavras, 17% de palavras correspondentes e 1 quase palavra. Esses dados mostram que apesar do Treino 1 reconhecer todas as palavras, ele não foi capaz de transcrever os dados corretamente, visto o desempenho ruim quanto a palavras correspondentes. Da mesma forma, vemos um desempenho ruim de caracteres correspondentes.

Temos evidenciado que a ordem semântica de todos os testes foi ruim. Isso mostra um problema grave quanto aos treinos.

Nenhum dos treinos conseguiu passar no teste de validação até o momento.

VI. CONCLUSÃO

Apesar da ferramenta do Tesseract ser muito eficiente sozinha, para treinos é visto uma dificuldade por conta da complexidade e do trabalho manual necessário para se preparar os arquivos. O banco de dados é muito bom, porém não foi possível trabalhar usando apenas os arquivos fornecidos, foi necessário despender um tempo útil apenas para recortar e nomear as imagens, isso torna o trabalho cansativo e pouco eficiente. Em razão disso não foi possível obter mais imagens para o treino no tempo disponível.

Um problema que tivemos foi o tempo de treino, este levando em média 40 minutos. Talvez esse problema seja contornado utilizando uma *cloud*, pois com maior poder de processamento seria mais fácil identificar quando houver erros, ao invés de esperar 40 minutos para perceber que o treino falhou.

Ao analisar a literatura notamos que é possível melhorar o desempenho do algoritmo ao utilizar um Modelo Oculto de Markov (HMM) na etapa de reconhecimento de forma. As características extraídas das linhas seriam alimentadas para um HMM esquerda-direita[3] que junto ao tesseract conseguiria adotar uma melhor escolha do caracteres e eventualmente da palavra com o uso de um dicionário apropriado. O algoritmo poderia ser melhorado também caso houvesse uma segmentação das palavras em seções como seção superior, inferior e meio, isso torna a classificação mais otimizada.

Uma última observação em relação ao treino e teste é em relação ao nível atual do aprendizado de máquina para OCR *offline* para manuscritos, temos na literatura que o reconhecimento de manuscritos é um trabalho de nicho, sendo bom apenas para alguns padrões de escrita, assim obtendo algo próximo de 80% de taxa de acerto de palavra no melhor dos casos, diferente do setor de OCR para textos impressos que obtém cerca de 99% de acerto no melhor dos casos.

Por fim, temos que o Tesseract com o uso do LSTM é uma ótima ferramenta, seu reconhecimento padrão é bom, de forma que ele supera todos os nossos treinos, isso pode ser por conta do número maior de imagens de treino (400.000 com variação de 4.500 fontes). O modelo de HMM, de acordo com a literatura, pode aumentar consideravelmente a taxa de acerto no reconhecimento, mesmo para o tesseract. Infelizmente nosso projeto não conseguiu atingir esse valor, no entanto, em trabalhos futuros pretendemos ajustar estes valores para torná-los atrativos para comercialização anteriormente idealizada.

VIII. O QUE SERÁ ENTREGUE?

Uma IA capaz de reconhecer padrões em letras e números sendo assim capaz de colocar os caracteres de forma digital em um eventual pdf.

IX. REFERÊNCIAS

1. N. Arica, Fatos T. Yarman-Vural, “Optical character recognition for cursive handwriting”, IEEE Transactions on Pattern Analysis and Machine Intelligence.
Disponível em:
<https://ieeexplore.ieee.org/abstract/document/1008386>

2. Ray Smith, “An Overview of the Tesseract OCR Engine”, Google Inc.
Disponível em:
<https://static.googleusercontent.com/media/research.google.com/pt-BR//pubs/archive/33418.pdf>

3. Herv´e Bourlard, “Introduction to Hidden Markov Models”, Ecole Polytechnique F´ed´erale de Lausanne.
Disponível em:
<https://www.cs.ubc.ca/~murphyk/Software/HMM/labman2.pdf>

4. Rosey Holley, “How Good Can It Get? - Analysing and Improving OCR Accuracy in Large Scale Historic Newspaper Digitisation Programs”, D-Lib Magazine.
Disponível em:
<http://www.dlib.org/dlib/march09/holley/03holley.html>

5. J. M. White, G. D. Rohrer, “Image Thresholding for Optical Character Recognition and Other Applications Requiring Character Image Extraction”, IBM Journal of Research and Development.
Disponível em:
<https://ieeexplore.ieee.org/abstract/document/5390437>

6. Ravina Mithe, Supriya Indalkar, Nilam Divekar, “Optical Character Recognition”, International Journal of Recent Technology and Engineering. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.673.80614&4rep=rep14&4type=pdf>

7. Sandip Rakshit, Subhadip Basu, Hisashi Ikeda, “Recognition of Handwritten Textual Annotations using Tesseract Open Source OCR Engine for information Just In Time (iJIT)”, Proc. Int. Conf. on Information Technology and Business Intelligence. Disponível em: <https://arxiv.org/ftp/arxiv/papers/1003/1003.5893.pdf>

8. Dan Bloomberg, Leptonica. Disponível em: <https://github.com/DanBloomberg/leptonica>

9. wrznr, OCR-D, Github. Disponível em: <https://github.com/OCR-D/ocrd-train>

10. Amalia R., Venu G., “Handwritten CAPTCHA: Using the difference in the abilities of humans and machines in reading handwritten words”, Center of Excellence for Document Analysis and Recognition (CEDAR), Department of Computer Science and Engineering, University at Buffalo Buffalo, NY, USA. Disponível em: <https://ieeexplore.ieee.org/abstract/document/1363915>