UNIVERSIDADE DE SÃO PAULO

Instituto de Ciências Matemáticas e de Computação

Identificação de Fraudes de Pagamentos no Tribunal de Justiça de São Paulo

Leonardo Carneiro Feltran

São Carlos – SP

Identificação de Fraudes de Pagamentos no Tribunal de Justiça de São Paulo

Leonardo Carneiro Feltran

Orientador: **Prof. Dr. André Carlos Ponce de Leon Ferreira de Carvalho**

Coorientador: **Prof. Dr. João Eduardo Ferreira (IME-USP)**

Monografia final de conclusão de curso apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP, como requisito parcial para obtenção do título de Bacharel em Engenharia de Computação.
Área de Concentração: Aprendizado de máquina

USP – São Carlos
Junho de 2021

Feltran, Leonardo Carneiro
Identificação de Fraudes de Pagamentos no Tribunal de Justiça de São Paulo / Leonardo Carneiro Feltran. - São Carlos - SP, 2021.
36 p.; 29,7 cm.

Orientador: André Carlos Ponce de Leon Ferreira de Carvalho.
Coorientador: João Eduardo Ferreira (IME-USP).
Monografia (Graduação) - Instituto de Ciências Matemáticas e de Computação (ICMC/USP), São Carlos - SP, 2021.

1. Inteligência artificial. 2. Aprendizado de máquina. 3. Aprendizado profundo. 4. Processamento de imagens. 5. Busca reversa de imagens. I. Carvalho, André Carlos Ponce de Leon Ferreira de. II. Instituto de Ciências Matemáticas e de Computação (ICMC/USP). III. Título.

AGRADECIMENTOS

Agradeço, primeiramente, aos meus pais, João e Vita, que não tiveram oportunidades de estudo mas sempre viram na educação uma ferramenta transformadora, dando toda a base e força durante toda a caminhada de seus filhos. Agradeço ao meu irmão Bruno que sempre me apoiou dentro da universidade com sua experiência e conselhos que foram muito valiosos. Agradeço às amizades que construí durante a graduação que me permitiram a aprender, ter momentos e experiências que são de grande valor para meu desenvolvimento pessoal.

Por fim, agradeço aos professores André Carlos e João por me acolherem dentro do projeto e me permetirem realizar este trabalho. Agradeço também aos colegas do projeto Daniela, Kelly, Danilo e Pedro que tiveram grande importância para o desenvolvimento do projeto.

RESUMO

FELTRAN, L. C.. **Identificação de Fraudes de Pagamentos no Tribunal de Justiça de São Paulo**. 2021. 36 f. Monografia (Graduação) – Instituto de Ciências Matemáticas e de Computação (ICMC/USP), São Carlos – SP.

O Tribunal de Justiça de São Paulo (TJSP) é o maior tribunal do mundo em volume de processos, apresentando um acervo muito grande de processos e que estão sendo digitalizados. Sabendo das dificuldades de detectar fraudes de pagamentos nos processos de seu acervo e aproveitando a digitalização dos processos físicos, o TJSP desenvolveu juntamente com a Universidade de São Paulo (USP) um sistema para a detecção automática de fraudes das taxas processuais por meio da identificação de documentos de pagamento repetidos. A abordagem que é apresentada neste documento consiste numa comparação visual, feita a partir da extração de características das imagens e a comparação dos seus vetores descritores. Foi obtido um algoritmo robusto a alterações de rotações, ruídos, transladações, escalas e borrões na extração de características com vetor descritor de 765 posições.

Palavras-chave: Inteligência artificial, Aprendizado de máquina, Aprendizado profundo, Processamento de imagens, Busca reversa de imagens.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS E SIGLAS

BRIEF ... *Binary Robust Independent Elementary Features*
CNN ..... Rede Neural Convolucional
FAST .... *Features from Accelerated Segment Test*
HOG ..... *Histogram of Oriented Gradients*
OCR ..... Reconhecimento Ótico de Caracteres
ORB ..... *Oriented FAST and Rotated BRIEF*
SIFT ..... *Scale Invariant Feature Transform*
SURF .... *Speeded Up Robust Features*
TJSP ..... Tribunal de Justiça de São Paulo
USP ..... Universidade de São Paulo
VGG ..... *Visual Geometry Group*

SUMÁRIO

Capítulo 1

INTRODUÇÃO

1.1 Motivação e Contextualização

Os processos judiciais são ferramentas do sistema judiciário que buscam eliminar conflitos e fazer justiça por meio das leis e jurisprudências. Estes processos são compostos, principalmente, pelos sujeitos e pelo objeto do processo (CINTRA; GRINOVER; DINAMARCO, 1995).

Os sujeitos do processo tem uma configuração tríplice, sendo composta pelo Estado, pelo autor e pelo réu. O Estado é representado pelo juiz que tem o papel de um sujeito sem interesse nas causas apresentadas, sendo responsável por comandar a atividade processual e buscar uma solução para o bem geral, pacificando as pessoas em conflito. O autor e o réu são as partes parciais do processo que possuem conflitos entre si, sendo o autor aquele que promove uma ação judicial contra um sujeito, e o réu aquele contra quem a ação foi promovida. É necessária a existência de pelo menos duas partes conflitantes no processo de tal forma que todos sejam tratados com total igualdade e imparcialidade no tratamento processual.

Segundo a doutrina brasileira, predomina-se a ideia de uma relação triangular entre estes sujeitos, sendo que o autor e o réu devem ter lealdade recíproca e a parte vencida tem a obrigação de reembolsar a vencedora as custas despendidas, enquanto que a relação juiz-autor e juiz-réu são realizadas por meio de um advogado. A presença do advogado é obrigatória por lei em qualquer processo judicial devido à condição inicial de ser um sujeito estranho ao conflito, que tem condições psico-intelectuais e conhecimento do direito para resolver os conflitos por meio da realização da justiça.

O outro elemento do processo, o objeto, consiste no elemento causador da disputa entre as partes. Este objeto possui uma grande abrangência, podendo ser desde objetos concretos, como bens móveis ou imóveis, até objetos abstratos como a moral de uma empresa ou alguém.

De acordo com o objeto da disputa, ele possui um valor, podendo ser o valor do objeto material em disputa ou o valor que deve ser pago como compensação aos danos causados à moral do autor. Este valor é definido no momento da abertura do processo e interfere nos custos de taxas que devem ser pagas durante a tramitação do processo.

No ato da abertura de um processo, devem ser informados, portanto, os seus sujeitos, o advogado do autor, o valor da causa, ou do objeto em disputa, e as provas que o autor pretende

apresentar. A abertura do processo, então, irá gerar um identificador que é usado pelo tribunal para referência e organização interna.

O Tribunal de Justiça de São Paulo (TJSP) tem como função disponibilizar os recursos para a execução de processos buscando uma solução de acordo com o cumprimento das leis e das jurisprudências do Estado de São Paulo. O TJSP é o maior tribunal do mundo com relação ao volume de processos, possuindo 25% dos processos em andamento no Brasil. Para garantir o bom funcionamento e a eficiência na tramitação destes processos, o TJSP possui 320 comarcas no estado e mais de 40 mil funcionários.

Pela estrutura mostrada acima, podemos ter uma noção dos custos que a operação e manutenção desta infraestrutura pode exigir. A verba para operação do TJSP vêm de duas principais fontes, do governo estadual e do pagamento de taxas processuais.

Estas taxas processuais são geradas e devem ser pagas pelas partes do processo enquanto este avança dentro do tribunal. Estas cobranças são feitas pelos documentos chamados de guias que possuem diferentes tipos e valores de acordo com o estágio do processo. As guias, assim como os comprovantes de pagamento das mesmas, devem ser apresentadas ao tribunal para anexação ao processo, que passam por uma conferência pelo tribunal.

Mesmo com esta conferência das guias, podem ocorrer falhas, o que permite a fraude de reúso de guias e comprovantes de pagamentos para comprovação de pagamento de outras taxas. Como o acervo do TJSP é físico e muito volumoso, a tarefa de identificar estes reusos se torna impossível.

Atualmente, o TJSP está em processo de digitalizar todo o seu acervo físico de processos, o que facilitará a conferência dos documentos de pagamento. Para fazer esta conferência, o tribunal solicitou à Universidade de São Paulo (USP) o desenvolvimento de uma aplicação que fosse capaz de encontrar os reusos de guias e comprovantes de forma automática na parte do seu acervo já digitalizado.

Este projeto conjunto entre TJSP e USP foi dividido em três fases, sendo que cada uma possui um modo para fazer a identificação de documentos repetidos e apontar possíveis fraudes. O trabalho ao qual este documento se trata apresenta uma fase do projeto todo, que será descrito por completo nos próximos capítulos.

1.2 Objetivos

O processo de identificar o reúso de guias e comprovantes possui três fases, sendo que as duas primeiras buscam fazer a extração de informações dos documentos por meio de um sistema de Reconhecimento Ótico de Caracteres (OCR) para povoar uma base de dados, enquanto que a terceira fase busca fazer a extração de características visuais das imagens e compilá-las em um vetor de características. Na primeira e segunda fase, a fraude é identificada a partir do cruzamento

de informações da base de dados, enquanto que na terceira fase, que é o foco deste trabalho, a provável fraude é encontrada a partir dos documentos que possuem descritores, ou vetores de características, mais próximos.

Focando na fase 3, pode-se dizer que há duas partes principais, a extração de características e a busca pelos vetores próximos. Para ambos há muitos algoritmos com bom desempenho, sendo que no armazenamento e busca dos vetores existem ótimas soluções que conseguem retornar a resposta rapidamente.

O elemento mais complexo desta fase é o extrator de características, que além de extraí-las deve escrevê-las em um vetor descritor. Como já foi dito, muitos algoritmos já fazem isso, mas o problema é que muitos não funcionam bem quando há alguma modificação nas imagens como rotação ou adição de ruídos. Então, é desejada uma forma de construir um vetor descritor para as imagens que seja robusto a um número maior de alterações, garantindo um tamanho de descritor que permita uma busca rápida pelas imagens similares.

1.3 Organização

Este documento está dividido em mais três capítulos. No Capítulo 2 são apresentadas as literaturas que embasaram e permitiram o desenvolvimento deste trabalho. No Capítulo 3, é descrito como a extração de características das imagens é feita e como os descritores das imagens são construídos. Por fim, no Capítulo 4 este trabalho é concluído, mostrando possíveis trabalhos futuros assim como uma opinião sobre o curso de graduação de Engenharia de Computação.

Capítulo 2

TÉCNICAS E TECNOLOGIAS UTILIZADAS

2.1 Considerações Iniciais

Um sistema de busca reversa de imagens se baseia basicamente em duas partes principais, a extração de características das imagens e o sistema de busca. Para o bom desempenho do sistema, precisa-se de uma extração que consiga capturar as nuances das imagens e uma busca rápida mesmo em um ambiente superpovoado. Abaixo são apresentadas as técnicas que são utilizadas e algumas referências que serviram de base para o desenvolvimento do projeto.

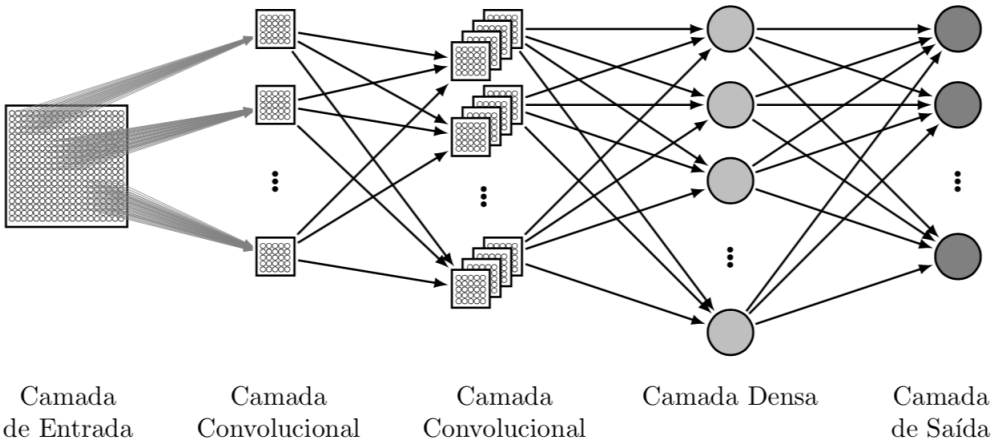
2.2 Conceitos e Técnicas Relevantes

2.2.1 Transfer learning e extração de características

Uma Rede Neural Convolucional (CNN) (SIMONYAN; ZISSERMAN, 2015) é composta por diferentes camadas que possuem diferentes papéis, sendo as mais importantes as camadas convolucionais e as camadas densas. As camadas convolucionais possuem filtros que extraem características das imagens de entrada e as condensam em um vetor, enquanto que as camadas densas são compostas por neurônios artificiais e são responsáveis pela classificação. Estas camadas podem estar dispostas de diferentes modos, o que criam arquiteturas com variados propósitos, por exemplo, a arquitetura *Visual Geometry Group* (VGG) (SIMONYAN; ZISSERMAN, 2015) foi criada para análise de profundidade nas CNNs, enquanto que a MobileNet (HOWARD *et al.*, 2017) foi desenvolvida para ter um número reduzido de parâmetros. De modo geral, as CNNs apresentam uma estrutura base semelhante que é mostrada na Figure 1.

A fase de treinamento de uma CNN é feita a partir da apresentação de uma entrada juntamente com a saída desejada. Com essas informações, a rede é capaz de ajustar os parâmetros internos a fim de maximizar os acertos para o conjunto de treinamento. Esta etapa é fundamental para o bom desempenho do algoritmo, sendo um processo demorado, que demanda muito poder computacional e um conjunto de dados numeroso. Estas exigências acabam limitando muito o uso das CNNs em diversos casos, principalmente nos casos em que se tem poucos recursos computacionais. Um modo de reduzir as exigências é o uso da transferência de conhecimento (ou *transfer learning*), que é o uso das redes já treinadas, que já aprenderam a extrair características

Figura 1 – Representação da arquitetura de uma CNN.



Fonte: http://rafaelsakurai.github.io/images/posts/2017-12-20-cnn-mapreduce.png

das imagens. Devido a este aprendizado prévio, o treinamento destas redes leva menos tempo e exige uma base de dados menor para se obter um bom treinamento.

No caso apresentado, a base de dados não deve ser classificada, mas deve ter suas características extraídas e compiladas em um vetor, por isso é utilizado apenas a parte responsável por esta tarefa, a estrutura composta pelas camadas convolucionais. Keras (CHOLLET *et al.*, 2015) disponibiliza um conjunto de arquiteturas já treinadas nas quais pode-se facilmente fazer a seleção de usar ou não as camadas densas.

2.2.2 Extratores de características

Em processamento de imagens, existem diversos extratores de características que buscam descrever o conteúdo a partir de características locais das imagens. Muitos se baseiam em encontrar pontos, cantos ou bordas nas imagens e descrevê-las de acordo com a sua vizinhança.

As vantagens destes algoritmos estão na aplicação de um contexto mais genérico já que não depende de uma base de dados para treinamento que ensina como fazer as extrações de características, e exigem menos recursos computacionais já que os algoritmos não apresentam tanto parâmetros internos como a maioria das arquiteturas de redes neurais.

Os dois descritores mais conhecidos são o *Scale Invariant Feature Transform* (SIFT) (LOWE, 1999) e o *Speeded Up Robust Features* (SURF)(BAY *et al.*, 2008). O SURF foi desenvolvido para ser uma opção mais rápida ao SIFT, fazendo alterações e aproximações de operações com total foco no ganho de velocidade. Estas alterações levaram a uma perda de desempenho, mas o algoritmo ainda apresenta bom desempenho na extração de características e criação de descritores das imagens.

Embora SURF e SIFT sejam algoritmos com bom desempenho para extração de pontos-chave e de características, eles são algoritmos patenteados, podendo causar problemas futuros

caso o TJSP utilize este sistema. Uma solução foi usar o *Oriented FAST and Rotated BRIEF* (ORB) (RUBLEE *et al.*, 2011), um algoritmo desenvolvido pelo OpenCV Labs buscando maior rapidez e com desempenho semelhante ao SIFT e ao SURF.

O ORB é composto por duas outras técnicas, o *Features from Accelerated Segment Test* (FAST) (ROSTEN; DRUMMOND, 2006) e o *Binary Robust Independent Elementary Features* (BRIEF) (CALONDER *et al.*, 2010). O primeiro algoritmo é um identificador de pontos-chave que utiliza a comparação de intensidade de cada pixel com a sua vizinhança para definir se este ponto é importante ou não. Ele se destaca do SIFT por ser muito mais rápido e também se destaca com relação ao SURF por apresentar uma melhor detecção de pontos-chave.

BRIEF é um algoritmo que cria um vetor de características para pontos-chave que foram previamente calculados, usando, por exemplo, o FAST. Ele é um descritor binário que utiliza das vizinhanças dos pontos-chave para poder criar seus descritores que podem ter um tamanho de 128, 256 ou 516 bits. Estes descritores são montados a partir da seleção de pontos aleatoriamente em volta do ponto-chave, preenchendo as posições do descritor com “0” ou com “1” a partir da comparação das intensidades dos pixels.

Além de utilizar estes dois algoritmos, ORB faz outras operações para poder se tornar mais robusto com relação a rotação e variação de escala. O FAST é aplicado na imagem de entrada em diferentes escalas para selecionar os pontos-chaves nestes diferentes níveis. Em seguida, para cada ponto-chave encontrado, é calculado um ângulo do seu gradiente de intensidade que é usado para calcular o descritor de cada ponto.

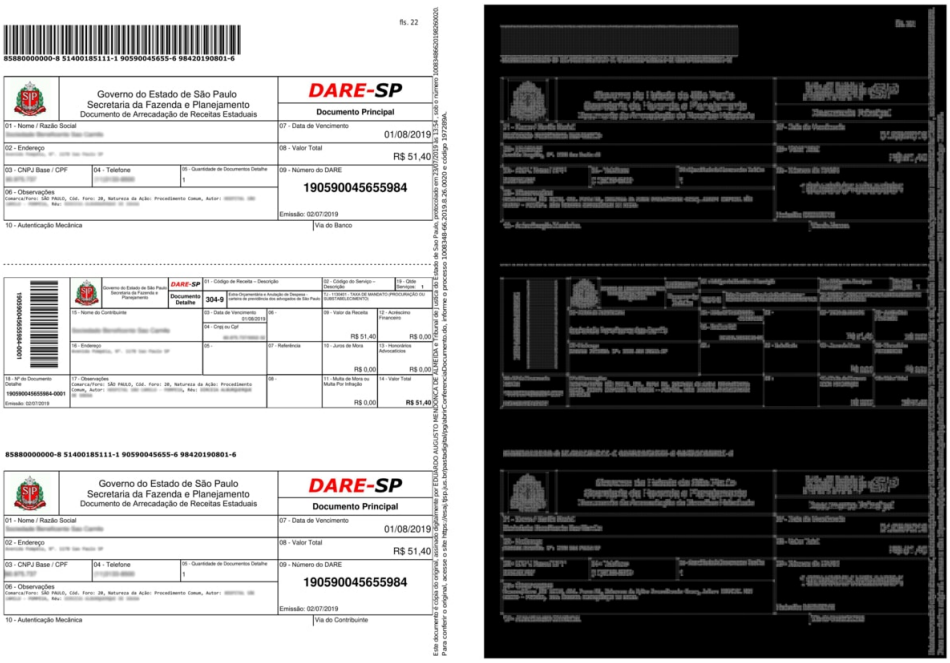
Um outro descritor de características muito usado é o *Histogram of Oriented Gradients* (HOG) (DALAL; TRIGGS, 2005). Ele consiste na aplicação de um filtro derivativo para extração dos gradientes das imagens, fazendo a captura das bordas dos objetos. Em seguida, a imagem é dividida em blocos onde são calculadas as orientações dos gradientes, sendo montado um histograma para cada bloco. Cada histograma é normalizado para tornar o algoritmo mais robusto a mudanças de iluminação e de sombras, sendo o descritor da imagem composto pela concatenação destes histogramas. A Figure 2 apresenta um exemplo visual do HOG aplicado em uma guia da base de dados do TJSP.

2.2.3 Estrutura de dados

A busca de imagens similares consiste na busca dos descritores das imagens mais próximas da *query*, podendo usar qualquer medida de similaridade, como cosseno ou euclidiana. O cálculo da similaridade pode requerer muita computação para os cálculos de distância com a base armazenada e ordenação, sendo necessário uma uma estrutura de dados que consiga fazer as comparações e recuperar as imagens similares rapidamente sem exigir muito poder computacional.

A árvore kd (ou *kd-tree*) (BENTLEY, 1975) é uma estrutura de dados em forma de

Figura 2 – Imagem de entrada (a esquerda) e a imagem de saída após aplicação do HOG (a direita).

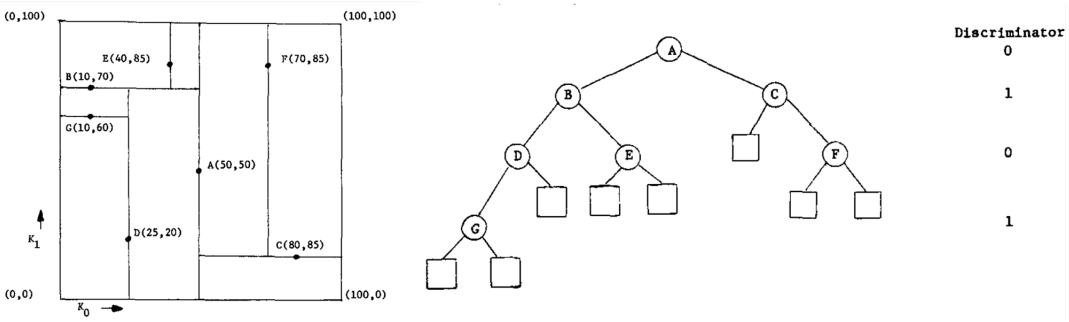


Fonte: Elaborada pelo autor.

árvore binária que dispõe os elementos com base em seus valores de cada dimensão do vetor de características. Ela apresenta uma complexidade média de *O(log n)*para inserção, remoção e procura de elementos, enquanto que no pior caso possui uma complexidade de *O(n)* para estas mesmas operações.

No exemplo da Figure 3, cada nível da árvore está relacionada a uma posição do vetor de características, sendo que a cada nível o espaço de características é dividido em dois de acordo com a a mediana dos valores nesta posição. Deste modo, a árvore estará sempre balanceada, permitindo uma busca que leva praticamente o mesmo tempo para todos os elementos e que não exige cálculos complexos para o cálculo das distâncias

Figura 3 – Representação de uma *kd-tree*



Fonte: Bentley (1975).

2.3 Trabalhos relacionados

Apesar de atualmente existirem muitos sistemas de busca reversa por imagens com bom desempenho, como o Google Imagens, não se sabe muito bem como funcionam devido aos segredos comerciais. Nos fóruns em que surge o questionamento sobre estes sistemas, muito se fala sobre a fusão de diversas técnicas de extração de características para se obter um bom desempenho. Dentro da academia isso não é diferente, onde diversas publicações mostram que esta fusão pode realmente fazer diferença no resultado final.

Shakaramia e Tarrah (SHAKARAMI; TARRAH, 2020) desenvolveram um sistema de busca reversa por imagens utilizando três técnicas diferentes para extração de características, o HOG, LBP (AHONEN; HADID; PIETIKAINEN, 2006) e AlexNet (KRIZHEVSKY; SUTSKEVER; HINTON, 2012), sendo esta última uma CNN. Os autores fizeram o treinamento da rede neural assim como fizeram o uso do PCA para fazer a redução de dimensionalidade dos vetores de características da AlexNet e do HOG, gerando um vetor com 128 posições após o processamento. De acordo com a publicação, este método se sobressai de alguns métodos que utilizam apenas um extrator de características como AlexNet e SIFT, assim como algumas técnicas que utilizam mais de um algoritmo.

Liu et al. (LIU *et al.*, 2017) fizeram o uso de transferência de conhecimento da uma rede VGG treinada para a base de dados ImageNet (DENG *et al.*, 2009) para identificação de nódulos na tireoide. Além disso, os autores também utilizam os algoritmos HOG, SIFT e LBP para a construção de um novo espaço de características cuja compreensão dos nódulos fica mais clara.

Karami et al. (KARAMI; PRASAD; SHEHATA, 2017) buscaram comparar SIFT, SURF e ORB, já que cada um busca uma forma de melhorar o outro. Em geral, ORB apresentou um desempenho superior ao SURF mas inferior ao SIFT, se destacando no reconhecimento de imagens com diferentes escalas. Além disso, é mostrado também que o ORB extrai a maior parte dos pontos-chave no centro da imagem, enquanto que os outros dois algoritmos apresentam pontos distribuídos por toda a imagem.

A técnica de busca reversa de imagens também de mostrou útil na identificação de caracteres Kannada (PARASHIVAMURTHY; NAVEENA; KUMAR, 2021). Os autores utilizaram os algoritmos SIFT e HOG para a extração de características e construção dos descritores das imagens dos caracteres encontrados no processo de segmentação. No processo de treinamento, há conhecimento da relação descritores-caractere, enquanto que na inferência a *kd-tree* é utilizada melhorar a indexação e a identificação das novas amostras, apresentado uma acurácia de até 90%.

2.4 Considerações finais

Vimos que há diversas opções para se fazer um sistema de busca reversa de imagens usando diferentes métodos, sendo a fórmula básica extrair características das imagens e alimentar uma base de dados. Além disso, as técnicas que utilizaram mais de um algoritmo para extração de características apresentaram maior desempenho em diferentes aplicações, nos dando base para fazer o mesmo.

Buscamos, assim, construir um sistema que seja robusto a pequenas alterações das imagens assim como um sistema que não exija muito poder computacional, tanto no processamento das imagens quanto na recuperação dos objetos similares. Focando nos algoritmos HOG, ORB e MobileNet, buscaremos observar sob quais alterações cada algoritmo é robusto, e em seguida, analisaremos como uma versão híbrida contendo mais de um algoritmo de extração de características se comportam nos mesmos testes.

Capítulo 3

DESENVOLVIMENTO

3.1 Considerações Iniciais

Neste capítulo damos mais detalhes sobre o projeto feito entre o TJSP e a USP e em que ponto trabalhamos. Além disso, buscamos detalhar as metodologias de análise, a base de dados trabalhada e como o algoritmo proposto funciona.

No final do capítulo, apresentamos os resultados obtidos assim como as comparações com outros algoritmos e quais seriam os caminhos alternativos que poderíamos ter seguido durante o desenvolvimento.

3.2 Descrição do Projeto

3.2.1 Contextualização

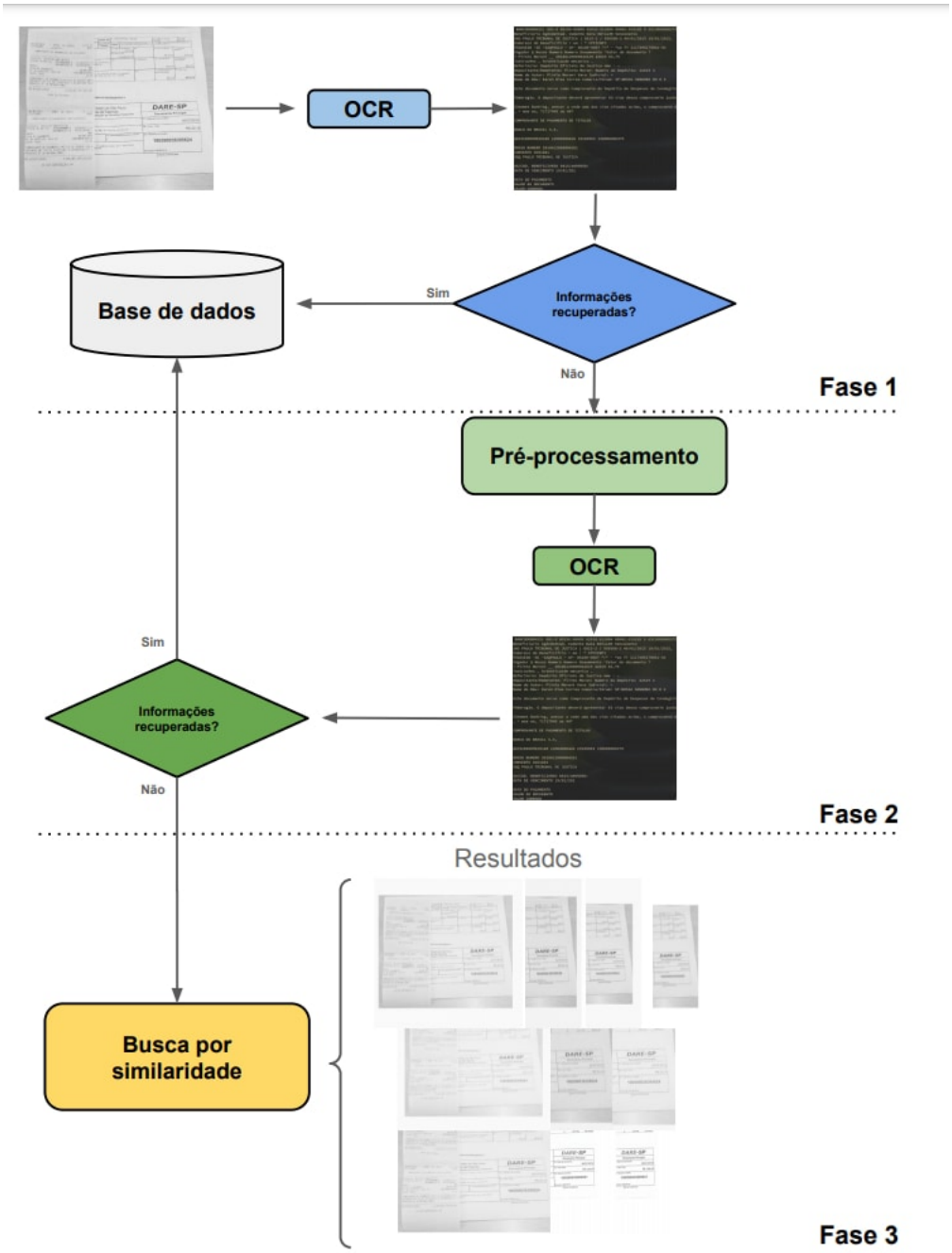
O projeto apresentado neste documento faz parte de um projeto maior realizado pela USP e pelo TJSP. Este projeto maior consiste na identificação de fraudes no pagamento de guias de custas emitidas pelo TJSP que foram utilizadas para comprovar o pagamento de várias taxas processuais. O TJSP possui o conjunto de todos os documentos apresentados pelas partes e seus advogados e que estão sendo totalmente digitalizados, nos permitindo identificar os reuso de documentos a partir de técnicas computacionais, como as que serão detalhadas a seguir.

O projeto completo possui três fases, como pode ser visto na Figure 4. As duas primeiras fases buscam a captura de informações dos documentos, detectando possíveis fraudes a partir do cruzamento de informações. A terceira fase lida com documentos que não puderem ter suas informações extraídas e convertidas em textos, usando, portanto, uma abordagem de comparação e cruzamento visual dos documentos.

Apesar de ter basicamente o mesmo funcionamento, as fases 1 e 2 se diferenciam em pré-processamento que são feitas nas imagens. A fase 1 não realiza operações complexas, enquanto que a fase 2 faz uso de processamentos como correção de iluminação, correção de rotação, remoção de linhas, entre outros, que exigem um maior poder computacional e mais tempo para processamento.

A terceira fase, da qual este trabalho se trata, lida com as imagens que não tiveram

Figura 4 – Fases do projeto entre a USP e o TJSP

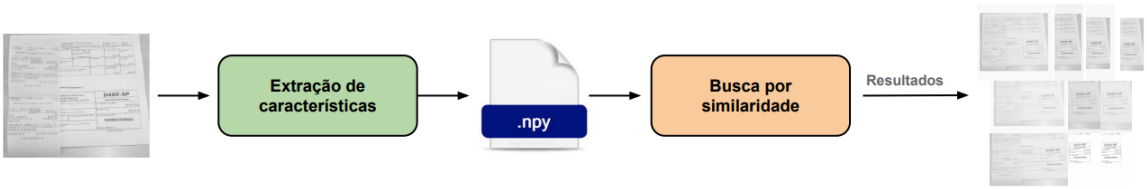


Fonte: Elaborada pelo autor.

sucesso na extração das informações nas fases anteriores, sendo a solução encontrada para estes casos a comparação visual dos documentos. Esta fase apresenta componentes internos que irão permitir a extração de características das imagens, armazená-las em uma estrutura de dados e permitir a busca de um conjunto de imagens que possuem maior similaridade. A Figure 5 apresenta um diagrama destes componentes como eles se relacionam.

A primeira etapa é um pré-processamento que prepara as imagens para a extração de características. Como trabalhamos com diferentes algoritmos, eles exigem diferentes em

Figura 5 – Estrutura interna da fase 3



Fonte: Elaborada pelo autor.

diferentes formatos de entrada, como diferentes dimensões e diferentes codificações de cores, sendo essencial para a boa extração de características.

Em seguida, são extraídas as características da imagem utilizando os algoritmos ORB, HOG e MobileNet. Estes algoritmos buscam pontos de grande importância e bordas nas imagens e os compila em um vetor descritor que é usado na comparação das imagens. Este vetor é escrito em um arquivo ".npy" que é utilizado pela estrutura de dados.

Na última etapa é feita a leitura dos arquivos ".npy" que contêm os vetores para cada imagem e os armazena em uma *kd-tree* para buscas futuras. Este método garante operações mais simples para encontrar os vetores mais próximos, assim como agilidade e economia de recursos computacionais.

3.2.2 Base de dados

Os dados que trabalhamos fazem parte do acervo do TJSP que está sendo totalmente digitalizado, sendo esta parte de teste e desenvolvimento aplicada, inicialmente, nos processos da 4ª Vara Cível do Foro Regional XII - Nossa Senhora do Ó, da cidade de São Paulo.

Deste Foro, trabalhamos em 139.603 processos que possuem 416.316 arquivos, distribuídos entre guias e comprovantes de pagamento. Estes aquivos passam pelas fases descritas acima, sendo que a quantidade de documentos para se processar reduz a cada fase, restando muito pouco para a fase 3. No momento da escrita deste documento não tinhamos todos os documentos processados, mas tinhamos a estimativa do quanto cada fase processa, sendo os valores indicados na tabela Table 1.

Tabela 1 – Quantidade de dados processados por cada fase do projeto TJSPxUSP

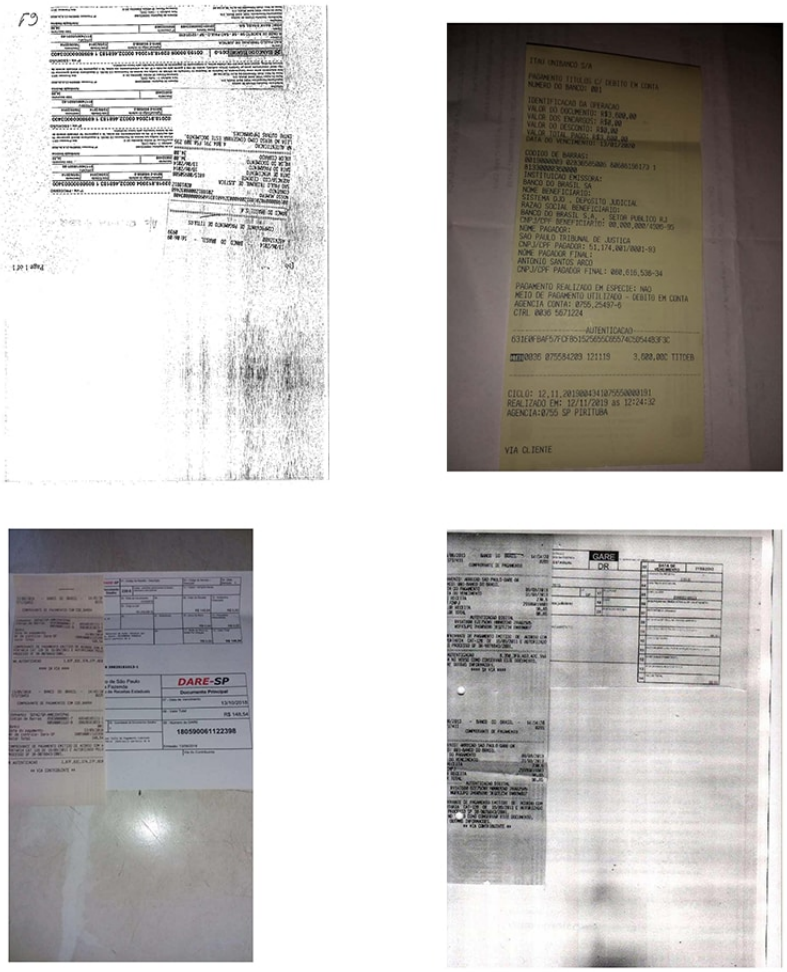
Fase	Quantidade processada (%)	Quantidade processada (absoluto)
Fase 1	78,56	327.058
Fase 2	20,14	83.846
Fase 3	1,3	5.412
Total	**100,0**	**416.316**

Fonte: Dados da pesquisa.

Em geral, as imagens que chegam à fase 3 possuem uma baixa qualidade, contendo

muitos ruídos, baixa resolução, ou mesmo documentos amassados e mal posicionados, como pode ser visto nas amostras da Figure 6.

Figura 6 – Amostra de imagens que chegam à fase 3



Fonte: Elaborada pelo autor.

Devido ao grande volume, à complexidade e o bom desempenho das fases anteriores, tínhamos apenas 165 imagens que chegaram à fase 3. Para uma melhor análise dos algoritmos de extração de características, adicionamos 135 imagens que não obtiveram sucesso na captura de informações na fase 1.

3.2.3 Metodologia

Para garantir o bom desempenho da comparação visual das imagens, temos que garantir a eficácia do algoritmo para diferentes casos. Definimos, então, um conjunto de testes que nos permitiram avaliar a extração de características feita pelos algoritmos, encontrando quais são os seus pontos fortes e pontos fracos.

O ideal para um algoritmo de extração de características é que ele consiga capturar os pontos de maior importância nas imagens, não importando defeitos que elas podem ter. Entretanto, sabemos que pequenas alterações nas imagens podem afetar a captura de características, resultando em capturas mais precisas sob certas alterações do que outras. Para analisar o comportamento dos algoritmos com relação a cada circunstância, foram feitos testes dos algoritmos utilizando imagens com as seguintes alterações:

- Rotação;
- Transladação;
- Ampliação ou redução;
- Borrões;
- Iluminação;
- Ruídos.

Em cada critério de análise da lista acima, testamos diferentes intensidade e tipos de alterações, buscando analisar em quais pontos ou sob quais tipos de alterações a extração de características acaba não sendo eficiente.

A base de dados de teste foi povoada pelas imagens sem qualquer modificação e a busca foi feita a partir de uma *query* que consiste em uma imagem com uma das alterações analisadas. A resposta desta pesquisa é um conjunto de 10 imagens cujos vetores de características estão mais próximos da *query* de acordo com a distância euclidiana. A partir destes resultados, consideramos que o houve acerto se a imagem original está entre as imagens recuperadas na busca, nos permitindo calcular a acurácia do extrator de acordo com a Equação 3.1.

$$acuracia = \frac{n_{acertos}}{n_{imagens}} \tag{3.1}$$

Na equação 3.1, temos $n_{imagens}$ como o número de imagens na base de dados e $n_{acertos}$ o número de acertos.

A partir da relação acurácia, tipo de alteração e intensidade de alterações, fizemos as análises de como o extrator de características se comportam, quais são seus limites e como podemos aumentar a sua robustez.

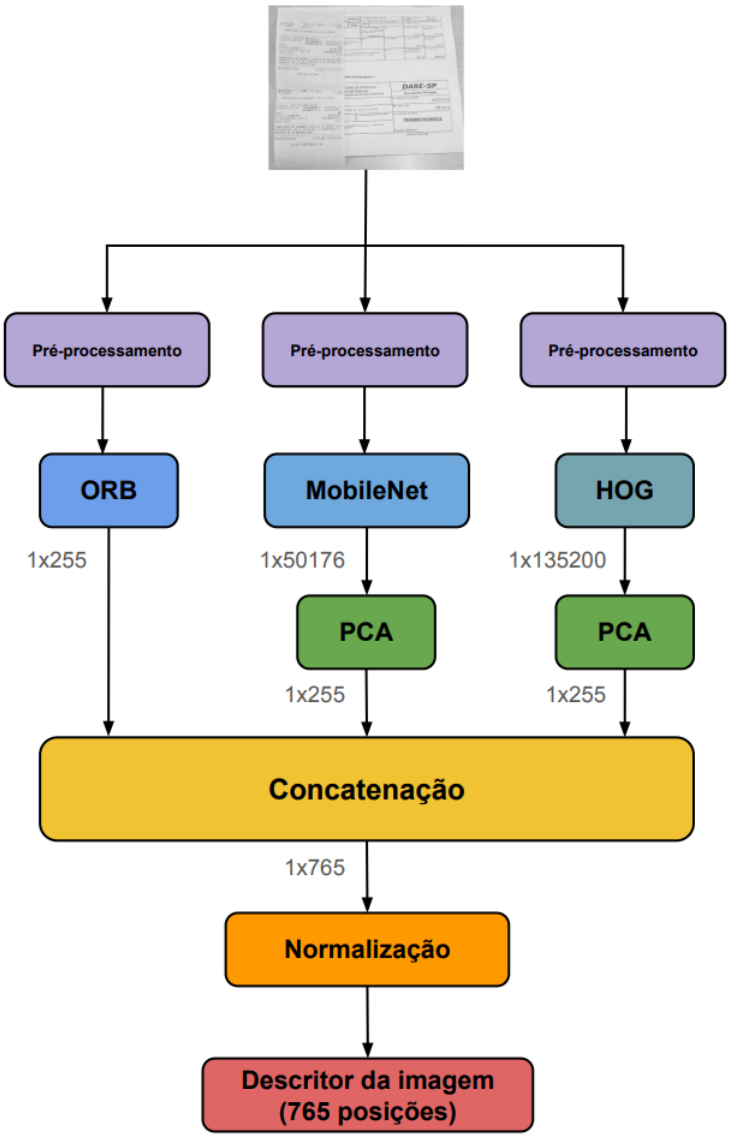
3.3 Descrição das Atividades Realizadas

Como mostrado anteriormente, o sistema descrito neste trabalho apresenta, principalmente duas partes, a extração de características e armazenamento, sendo a primeira parte

composta por um conjunto de algoritmos de extração de características, enquanto que o segundo consiste em uma estrutura de dados específica para busca eficiente por distância.

A extração de características é feita usando três algoritmos diferentes, o HOG, o ORB, e uma CNN, a MobileNet. O intuito do uso deste conjunto de algoritmos é criar um algoritmo híbrido que consiga combinar os pontos fortes de cada um, obtendo um algoritmo mais robusto a variadas alterações que o fraudador pode fazer nas imagens. Este algoritmo híbrido possui a estrutura interna como mostrada na Figure 7.

Figura 7 – Diagrama do extrator de características



Fonte: Elaborada pelo autor.

Cada algoritmo de extração de características possui um padrão de entrada diferente, o que exige um pré-processamento individual para cada um. O ORB e o HOG trabalham com imagens em escala de cinza, porém com tamanhos de imagens diferentes, o primeiro com entradas de 600x600 enquanto que o segundo trabalha com imagens de tamanho 400x400.

Estes dois algoritmos exigem um tamanho maior para poder capturar características menores, como pequenos números e ou letras. Já o processamento da MobileNet consiste na redução de dimensionalidade das imagens para um tamanho de 224x224, que é o tamanho exigido devido a transferência de conhecimento, e imagens coloridas. Em geral, CNNs exigem que as entradas tenham uma escala de valores específica, pois estas escalas levam a uma melhora no treinamento. O Keras permite abstrair esta parte a partir do uso de uma função que faz a transformação da escala para o formato necessário.

O ORB tem como saída um conjunto de descritores dos pontos-chave com 32 elementos de 8 bits. Para comparação, a biblioteca OpenCV busca os pontos com descritores similares, sendo que quanto mais pontos similares, maior é a similaridade entre as imagens, porém este tipo de comparação é muito custosa e poderia atrapalhar na velocidade de operação do nosso sistema.

Um modo para converter os descritores dos pontos em um único vetor de características é criando um histograma da saída do ORB. Isto se baseia no fato de que imagens similares terão muitos pontos cujos descritores são similares, e , logo, possuem uma distribuição de valores dos descritores semelhantes, porém imagens diferentes possuem poucos pontos em comum, logo possuem distribuições diferentes. A Figure 8 apresenta uma comparação de três imagens, sendo as figuras (a) e (b) muito semelhantes por serem da mesma classe de documentos, enquanto que a (c) difere das outras por ter uma digitalização de melhor qualidade e ser de outra classe.

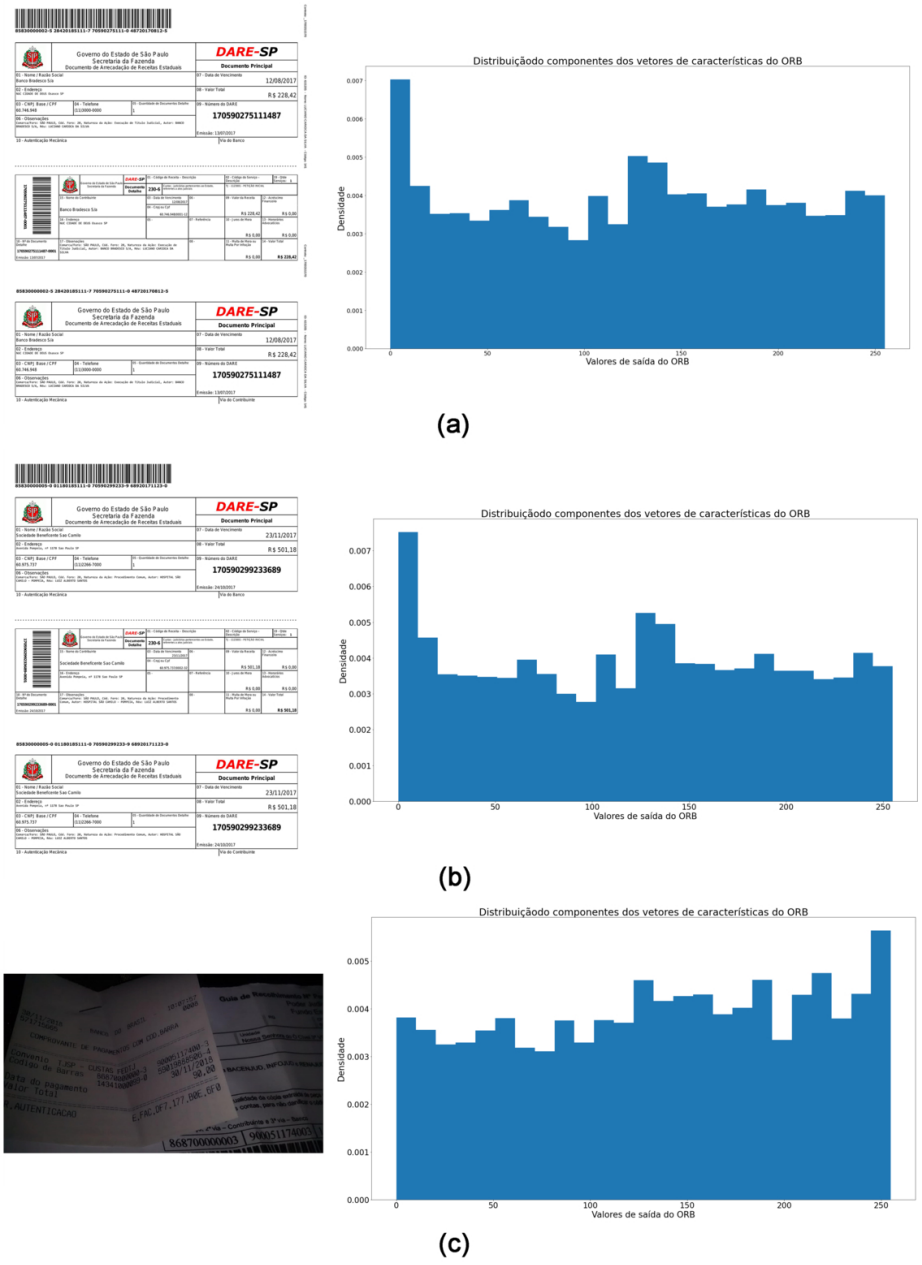
Os descritores do HOG e do MobileNet possuem muitas posições, sendo muito superior ao número de posições do ORB após a construção dos histogramas. Esta diferença no número de elementos de cada algoritmo pode causar um maior efeito dos algoritmos cujos descritores são maiores, enquanto atenua o efeito daqueles com menos posições. Por isso, é feita uma redução de dimensionalidades da saída do HOG e da MobileNet, utilizando PCA para que todos tenham uma descrição de um mesmo tamanho da imagem de entrada.

Após os pós-processamentos,os três vetores são concatenados, resultando em um vetor com 765 posições que é normalizado e então convertido em um arquivo ".npy".

A busca de similaridade é realizada a partir da estrutura de dados *kd-tree*, que irá receber os vetores no formato ".npy" e dispô-los em uma estrutura de árvore. Esta estrutura apresenta boas implementações na biblioteca Scikit-learn(PEDREGOSA *et al.*, 2011) para Python, reduzindo o esforço para o seu desenvolvimento.

Podem ocorrer dois tipos de busca na estrutura de dados, uma busca com o objetivo de encontrar documentos semelhantes no próprio conjunto de dados da estrutura, ou a partir de uma nova imagem. O primeiro tipo tem como resultado um relatório da relação de quais imagens apresentam maiores semelhanças entre si, enquanto que o segundo tipo apresenta um conjunto das 10 imagens mais semelhantes à *query*.

Figura 8 – Amostra de imagens e os seus histogramas gerados a partir dos descritores do ORB



Fonte: Elaborada pelo autor.

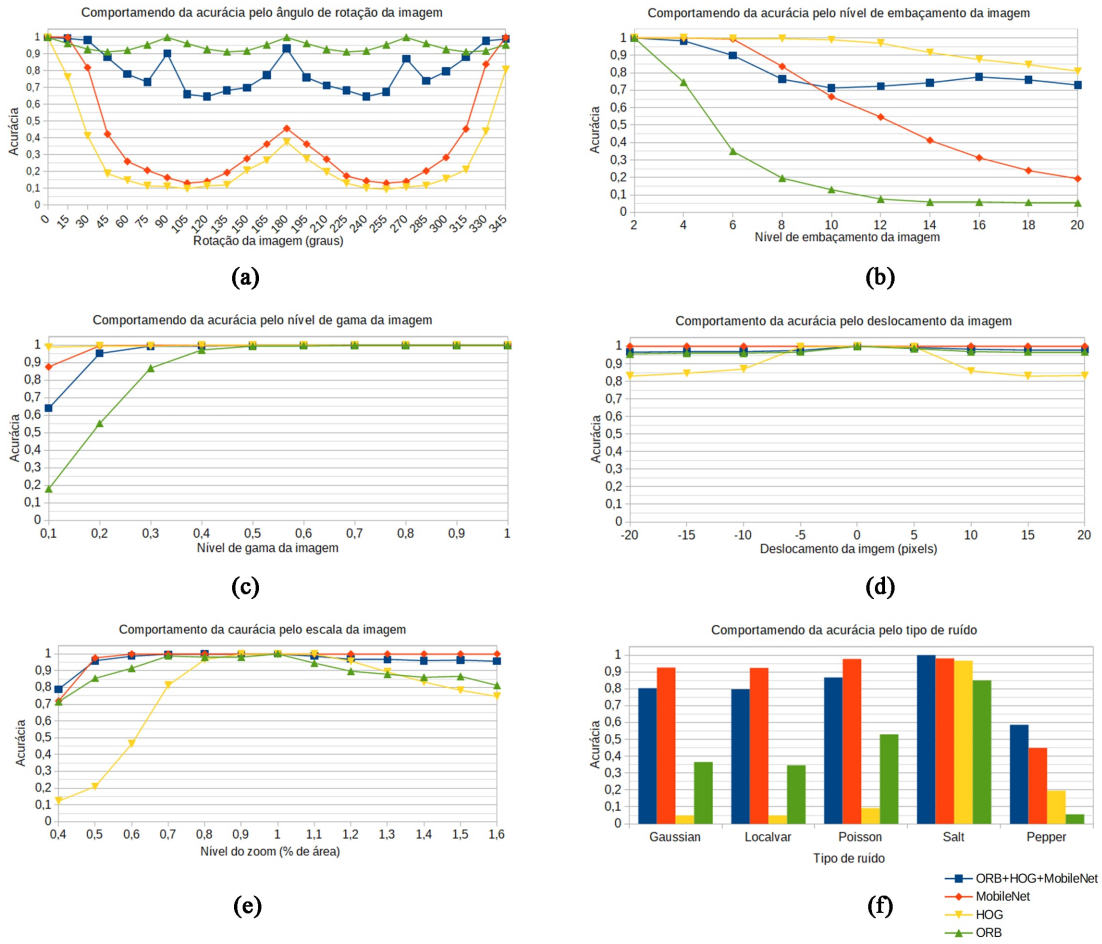
3.4 Resultados Obtidos

Os resultados que obtivemos foram calculados a partir de uma base de dados com 300 imagens, sendo 165 imagens das quais não foi possível capturar informações usando OCR, e 135 imagens selecionadas aleatoriamente entre as imagens que precisaram pré-processamento para que tivessem suas informações extraídas pelo OCR. Apesar de não ser composto por apenas dados que o sistema irá receber, podemos analisar se a busca reversa pelas imagens de fato funciona em imagens modificadas.

Os testes foram realizados povoando a estrutura de dados com imagens sem modificações,

enquanto que as *query* eram de imagens modificadas. As acurácias foram coletadas para cada nível ou tipo da alteração analisada. A Figure 9 apresenta os resultados para os quatro algoritmos analisados, o HOG, ORB, MobileNet o algoritmo proposto.

Figura 9 – Resultados obtidos nos testes dos algoritmos



Fonte: Elaborada pelo autor.

Na maioria dos resultados, o algoritmo proposto tem um desempenho satisfatório, existindo poucos casos que tenham uma acurácia menor de 60% usando-se uma recuperação de 10 imagens. O modelo proposto sempre apresentou um desempenho entre os melhores, sendo raros os momentos em que ele apresenta desempenho acima dos algoritmos puros, e não existindo momento algum em que ele seja o pior.

Destaca-se o bom desempenho do modelo proposto para as alterações de rotação no qual o modelo foi capaz de aumentar muito o desempenho com relação a MobileNet e ao HOG, mantendo o comportamento observado no ORB que é um maior desempenho nas orientações múltiplas de 90 graus. Nota-se também um grande aumento de desempenho nas imagens ruidosas ao se usar o modelo proposto, estando muito próximo do desempenho da MobileNet, a melhor neste quesito, chegando a ter um desempenho melhor para ruídos dos tipos Salt e Pepper.

De modo geral e para uma resposta com 10 imagens, o modelo proposto apresentou um

desempenho satisfatório e dentro do esperado, apresentando uma robustez a um maior número de alterações que os outros extratores analisados. Como ponto contrário, existe uma pequena perda de desempenho ao se fazer a junção dos algoritmos quando comparado com os outros, mas não é um valor que impossibilite o uso.

3.5 Dificuldades e Limitações

Este projeto surgiu com um foco diferente do seu final, teria Aprendizado Profundo e CNN como foco principal para extração das características, e utilizaria arquiteturas diferentes como Autoencoder. Porém, como já foi dito antes, o treinamento destes modelos são complexos, levando muito tempo para mesmo em computadores potentes. Devido aos problemas que o cluster Euler teve, estas opções foram removidas pois não seria viável fazer o treinamento dos modelos. Como lado positivo deste imprevisto, foi possível ver como de fato a junção de diferentes algoritmos para extração de características pode aumentar a robustez sem perder muita acurácia, sendo uma solução que pode de fato ser empregada e gerar trabalhos futuros ao se buscar combinações de outros algoritmos para diferentes aplicações.

Uma das limitações foi a validação do algoritmo devido o número reduzido de amostras disponíveis. Era possível utilizar mais imagens que não seriam aquelas que chegariam à fase 3, mas isso poderia gerar percepções falsas sobre o desempenho do algoritmo.

Um ponto que é de grande importância para o bom funcionamento deste projeto mas que não teve muito foco foi a estrutura de dados para armazenamento e busca das imagens similares. Esta parte acaba sendo um pouco difícil para se explorar e trabalhar em cima por ser uma área bem consolidada, com boas soluções já disponíveis. Seria possível ter explorado um pouco mais as soluções disponíveis para poder fazer uma comparação e uma seleção melhor.

3.6 Considerações Finais

O algoritmo proposto para extração de características de imagens se sustenta em diversos trabalhos que também buscam melhorar a qualidade dos vetores de características através da junção de diversos algoritmos. Ele busca utilizar os pontos fortes dos algoritmos HOG, ORB e MobileNet para poder aumentar a sua robustez e conseguir, em um só modelo, extrair características para uma grande gama de alterações que uma imagem pode sofrer.

Como resultado, temos um modelo que de fato apresenta uma maior robustez que os algoritmos testados, assim como apresenta uma boa acurácia na recuperação de imagens

Apesar de ter seguido caminhos diferentes da proposta inicial do projeto, ficamos contentes com o resultado obtido, abrindo novas possibilidades de novos trabalhos tanto para utilizar novas técnicas para extração de características como para o armazenamento de e busca das imagens similares.

Capítulo 4

CONCLUSÃO

4.1 Contribuições

Este trabalho faz parte de um projeto maior desenvolvido entre a USP e o TJSP, possuindo um possível aplicação no sistema final que está sendo desenvolvido. Ele permitirá fazer a identificação das fraudes para que ocorra a cobranças dos valores devidos o TJSP, além de que abrirá portas para tecnologias semelhantes dentro do tribunal.

Novos caminhos foram aberto para serem explorados futuramente a fim de buscar melhorias na extração de características utilizando novas construções para o vetor descritor das imagens, assim como a possibilidade de explorar diversos modos de busca e armazenamento pelas imagens similares.

Com relação a ganho pessoal, tive oportunidade de trabalhar em um grande projeto de pesquisa e desenvolvimento dentro da USP, com pessoas altamente capacitadas, com enorme experiência, que sempre contribuíram para a melhora dos resultados e incentivaram a busca por conhecimento que levaram ao avanço deste trabalho. Além disso, a relação com o TJSP durante o desenvolvimento deu um outro olhar dentro do projeto, possibilitando ter um contato com um cliente e poder desenvolver soluções para os seus problemas, uma experiência pouco comum dentro da universidade mas que dá muita experiência para o aluno como o mercado de trabalho pode ser.

Com a finalização deste trabalho, me sinto mais confiante para poder escolher qualquer caminho que queira após a graduação, pois tive dificuldades, vi que sou capaz de transpassá-las e sei que poderei fazer o mesmo no futuro independente onde esteja. A aplicação da metodologia científica durante o desenvolvimento deste projeto de forma mais criteriosa mostrou a mim sua eficácia ao obtermos um bom resultado e de acordo com o esperado inicialmente, reforçando que a ciência busca nos dar base e nos ajudar a evoluir, mesmo que tenhamos respostas que não são esperadas, são estas dificuldades que nos permitem evoluir ainda mais.

4.2 Trabalhos Futuros

Inicialmente, este projeto tinha um maior foco em Aprendizado Profundo e CNNs, buscando diferentes arquiteturas, tipos de aprendizado e treinamento para poder fazer a extração de

características das imagens, com muito foco na arquitetura de *Autoencoder*. Porém percebeu-se que seria complicado trabalhar com isso devido a limitações de *hardware* para o treinamento já que o cluster que seria utilizado apresentou problemas. Muitas pesquisas buscam a implementação desta arquitetura para aprendizado e extração de características de imagens, sendo um caminho que pode ser explorado no futuro.

Além disso, há algoritmos de extração de características sendo publicados com certa frequência, como o *Boosted Efficient Binary Local Image Descriptor* (BEBLID) que foi publicado no ano de 2020 mas que não foi explorado neste projeto. Estes novos algoritmos podem ajudar tanto no tempo de computação como também na qualidade do descritor gerado, precisando de testes para poder se ter o real desempenho.

Por fim, seria possível fazer uma análise mais detalhada dos algoritmos de armazenamento e busca dos vetores mais próximos, utilizando métricas diferentes como a similaridade por cosseno, ou mesmo diferentes estruturas de dados.

REFERÊNCIAS

AHONEN, T.; HADID, A.; PIETIKAINEN, M. Face description with local binary patterns: Application to face recognition. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 28, n. 12, p. 2037–2041, 2006. Citado na página 21.

BAY, H.; ESS, A.; TUYTELAARS, T.; GOOL, L. V.; LEUVEN, B. K. U. **Speeded-Up Robust Features (SURF)**. 2008. Citado na página 18.

BENTLEY, J. Multidimensional binary search trees used for associative searching. **Commun. ACM**, v. 18, p. 509–517, 1975. Citado 2 vezes nas páginas 19 e 20.

CALONDER, M.; LEPETIT, V.; STRECHA, C.; FUA, P. Brief: Binary robust independent elementary features. In: DANIILIDIS, K.; MARAGOS, P.; PARAGIOS, N. (Ed.). **Computer Vision – ECCV 2010**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010. p. 778–792. ISBN 978-3-642-15561-1. Citado na página 19.

CHOLLET, F. *et al.* **Keras**. 2015. <https://keras.io>. Citado na página 18.

CINTRA, A. C. d. A.; GRINOVER, A. P.; DINAMARCO, C. R. **Teoria geral do processo**. [S.l.]: Malheiros, 1995. 301-317 p. Citado na página 13.

DALAL, N.; TRIGGS, B. Histograms of oriented gradients for human detection. In: **2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)**. [S.l.: s.n.], 2005. v. 1, p. 886–893 vol. 1. Citado na página 19.

DENG, J.; DONG, W.; SOCHER, R.; LI, L.-J.; LI, K.; FEI-FEI, L. Imagenet: A large-scale hierarchical image database. In: IEEE. **2009 IEEE conference on computer vision and pattern recognition**. [S.l.], 2009. p. 248–255. Citado na página 21.

HOWARD, A. G.; ZHU, M.; CHEN, B.; KALENICHENKO, D.; WANG, W.; WEYAND, T.; ANDREETTO, M.; ADAM, H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. **CoRR**, abs/1704.04861, 2017. Disponível em: <http://arxiv.org/abs/1704.04861>. Citado na página 17.

KARAMI, E.; PRASAD, S.; SHEHATA, M. **Image Matching Using SIFT, SURF, BRIEF and ORB: Performance Comparison for Distorted Images**. 2017. Citado na página 21.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. In: PEREIRA, F.; BURGES, C. J. C.; BOTTOU, L.; WEINBERGER, K. Q. (Ed.). **Advances in Neural Information Processing Systems**. Curran Associates, Inc., 2012. v. 25. Disponível em: <https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf>. Citado na página 21.

LIU, T.; XIE, S.; ZHANG, Y.; YU, J.; NIU, L.; SUN, W. Feature selection and thyroid nodule classification using transfer learning. In: **2017 IEEE 14th International Symposium on Biomedical Imaging (ISBI 2017)**. [S.l.: s.n.], 2017. p. 1096–1099. Citado na página 21.

LOWE, D. Object recognition from local scale-invariant features. In: **Proceedings of the Seventh IEEE International Conference on Computer Vision**. [S.l.: s.n.], 1999. v. 2, p. 1150–1157 vol.2. Citado na página 18.

PARASHIVAMURTHY, R.; NAVEENA, C.; KUMAR, Y. Sift and hog features for the retrieval of ancient kannada epigraphs. **IET Image Processing**, v. 14, 01 2021. Citado na página 21.

PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V.; VANDERPLAS, J.; PASSOS, A.; COURNAPEAU, D.; BRUCHER, M.; PERROT, M.; DUCHESNAY, E. Scikit-learn: Machine learning in Python. **Journal of Machine Learning Research**, v. 12, p. 2825–2830, 2011. Citado na página 29.

ROSTEN, E.; DRUMMOND, T. Machine learning for high-speed corner detection. In: LEONARDIS, A.; BISCHOF, H.; PINZ, A. (Ed.). **Computer Vision – ECCV 2006**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006. p. 430–443. ISBN 978-3-540-33833-8. Citado na página 19.

RUBLEE, E.; RABAUD, V.; KONOLIGE, K.; BRADSKI, G. R. Orb: An efficient alternative to sift or surf. In: **ICCV**. [s.n.], 2011. p. 2564–2571. Disponível em: <https://doi.org/10.1109/ICCV.2011.6126544>. Citado na página 19.

SHAKARAMI, A.; TARRAH, H. An efficient image descriptor for image classification and cbir. **Optik**, v. 214, p. 164833, 05 2020. Citado na página 21.

SIMONYAN, K.; ZISSERMAN, A. Very deep convolutional networks for large-scale image recognition. In: BENGIO, Y.; LECUN, Y. (Ed.). **3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings**. [s.n.], 2015. Disponível em: <http://arxiv.org/abs/1409.1556>. Citado na página 17.