

UNIVERSIDADE ESTADUAL PAULISTA "JÚLIO DE MESQUITA FILHO"
FACULDADE DE CIÊNCIAS - CAMPUS BAURU
DEPARTAMENTO DE COMPUTAÇÃO
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

JOÃO RENATO RIBEIRO MANESCO

**ADAPTAÇÃO DE DOMÍNIO PARA RECONHECIMENTO DE
PADRÕES VISUAIS**

BAURU
Dezembro/2020

JOÃO RENATO RIBEIRO MANESCO

ADAPTAÇÃO DE DOMÍNIO PARA RECONHECIMENTO DE PADRÕES VISUAIS

Trabalho de Conclusão de Curso do Curso
de Bacharelado em Ciência da Computação
da Universidade Estadual Paulista “Júlio
de Mesquita Filho”, Faculdade de Ciências,
Campus Bauru.

Orientador: Prof. Associado Aparecido Nilceu
Marana

BAURU
Dezembro/2020

João Renato Ribeiro Manesco Adaptação de domínio para reconhecimento
de padrões visuais/ João Renato Ribeiro Manesco. – Bauru, Dezembro/2020-
60 p. : il. (algumas color.) ; 30 cm.
Orientador: Prof. Associado Aparecido Nilceu Marana
Trabalho de Conclusão de Curso – Universidade Estadual Paulista “Júlio de
Mesquita Filho”
Faculdade de Ciências
Bacharelado em Ciência da Computação, Dezembro/2020.

João Renato Ribeiro Manesco

Adaptação de domínio para reconhecimento de padrões visuais

Trabalho de Conclusão de Curso do Curso de Bacharelado em Ciência da Computação da Universidade Estadual Paulista "Júlio de Mesquita Filho", Faculdade de Ciências, Campus Bauru.

Banca Examinadora

Prof. Associado Aparecido Nilceu Marana

Orientador

Departamento de Computação

Faculdade de Ciências

Universidade Estadual Paulista "Júlio de Mesquita Filho"

Profª. Drª. Simone das Graças

Domingues Prado

Departamento de Computação

Faculdade de Ciências

Universidade Estadual Paulista "Júlio de Mesquita Filho"

Prof. Dr. Kelton Augusto Pontara da Costa

Departamento de Computação

Faculdade de Ciências

Universidade Estadual Paulista "Júlio de Mesquita Filho"

Bauru, 14 de dezembro de 2020.

*Dedico esta monografia à minha família e aos meus amigos por todo apoio que me permitiu
chegar até aqui.*

Agradecimentos

Agradeço primeiramente à minha família, que me apoiou e me permitiu chegar até aqui e por todos os ensinamentos durante todos estes anos de vida.

Agradeço também aos professores que tive, em especial a meus orientadores Prof. Assoc. Aparecido Nilceu Marana e Prof. Dr. Fabiano Borges da Silva, por todo o aprendizado que me proporcionaram durante a graduação.

Agradeço ao ex-aluno Gustavo Rosa pela construção do modelo de TCC em Latex.

Por fim, agradeço a todos os meus amigos e colegas de curso, pela companhia, apoio e por todas as horas que passamos juntos.

*Science is what we understand well enough
to explain to a computer. Art is everything else.*

Donald Ervin Knuth

Resumo

Sistemas de Reconhecimento de Padrões estão suscetíveis ao problema de deslocamento de domínio, que ocorre quando as distribuições dos domínios fonte e alvo são diferentes, causando problemas na acurácia do sistema. Técnicas de Adaptação de Domínio surgem como forma de lidar com o problema de deslocamento de domínio. Neste trabalho, quatro técnicas de adaptação de domínio foram aplicadas aos problemas de Reconhecimento de Objetos e Reconhecimento Facial a fim de observar e analisar o impacto da adaptação de domínio em problemas com presença do deslocamento de domínio. Os resultados obtidos mostram que, em geral, técnicas de adaptação de domínio melhoram o desempenho destes sistemas em termos de acurácia e separabilidade. Por fim, foi elaborado um aplicativo para dispositivos móveis que atua na tarefa de autenticação facial, comparando similaridades com e sem adaptação de domínio.

Palavras-chave: Adaptação de Domínio, Reconhecimento de Padrões, Reconhecimento de Objetos, Reconhecimento Facial.

Abstract

Pattern Recognition Systems are susceptible to the domain shift problem, which happens when the distributions of the source and target domains are different, resulting in problems in the accuracy of the system. Domain adaptation techniques emerge as a way to deal with the problem caused by domain shift. In this work, four domain adaptation techniques were applied to the Object Recognition and Facial Recognition problems in order to analyze the impact of domain adaptation on problems with the presence of domain shift. The obtained results show that, in general, domain adaptation techniques improve the performance of accuracy and separability metrics on these systems. Finally, a mobile application was developed approaching the task of facial authentication, comparing face similarities with and without domain adaptation.

Keywords: Domain Adaptation, Pattern Recognition, Object Recognition, Facial Recognition.

Listas de figuras

| | |
|---|----|
| Figura 1 – Exemplos de imagens de dígitos escritos a mão. | 15 |
| Figura 2 – Exemplos de imagens de dígitos obtidos de casas utilizando a ferramenta <i>Google Street View</i> . | 16 |
| Figura 3 – Exemplo de função de classificador em um espaço linearmente separável. | 18 |
| Figura 4 – Exemplo de espaço de características que não consegue ser separado linearmente. | 19 |
| Figura 5 – Função de classificação, com <i>overfitting</i> . | 19 |
| Figura 6 – Diagrama de Voronoi ilustrando o método 1NN, de modo que a região em que a observação se encontra determina sua classe, por meio do vizinho mais próximo. | 21 |
| Figura 7 – Construção do Classificador de Máxima Margem, dividindo os objetos azuis e verdes em um espaço com duas características. É possível perceber que os elementos das extremidades servem como vetores de suporte para definir as margens. | 22 |
| Figura 8 – Exemplo dos filtros de convolução L_{yy} (esquerda) e L_{xy} (direita) usados para o cálculo da matriz Hessiana no método SURF. | 25 |
| Figura 9 – Arquitetura da rede convolucional usada como base para a seleção de características e treino do DeCAF. | 26 |
| Figura 10 – <i>Clusters</i> de dados utilizados durante o protocolo DeCAF para auxiliar na escolha das características. De modo que os dados presentes à esquerda, extraídos da primeira camada da rede convolucional apresentada na Figura 9, enquanto os dados presentes à direita foram extraídos da sexta camada da rede. | 27 |
| Figura 11 – Contrastos de Haar utilizados para a detecção de faces, no método proposto por Viola e Jones (2004). | 28 |
| Figura 12 – <i>Pipeline</i> do método de detecção de faces MTCNN. | 29 |
| Figura 13 – Arquitetura da rede VGG Face. | 30 |
| Figura 14 – Exemplos de imagens de faces obtidas de quatro pessoas no domínio caracterizado pelas fotos de documentos de identidade (linha superior) e obtidas no domínio caracterizado pelas <i>selfies</i> (linha inferior). | 31 |
| Figura 15 – Exemplificação de um problema de deslocamento de domínio. No primeiro gráfico estão presentes os dados no domínio fonte, com seu classificador traçado em azul. No segundo gráfico, os dados do domínio alvo, com seu classificador traçado em vermelho. No terceiro quadro, os dados de ambos os domínios após a adaptação de domínio, com um classificador treinado no domínio comum. | 31 |

| | |
|--|----|
| Figura 16 – Exemplos de imagens de <i>notebooks</i> , presentes nos quatro tipos de domínios da base Office-Caltech. | 39 |
| Figura 17 – Diferentes tipos de variação presentes na base de dados ARFace, incluindo, variações de expressão facial, iluminação e oclusões na região ocular e na região da boca. | 40 |
| Figura 18 – Exemplos de imagens presentes na base de dados FRGC. As duas primeiras colunas representam imagens capturadas em ambiente controlado, enquanto que as últimas três, apresentam imagens obtidas em ambiente não controlado. | 41 |
| Figura 19 – Métricas de separabilidade obtidas na base de dados Office-Caltech, utilizando o descriptor SURF. | 49 |
| Figura 20 – Métricas de separabilidade obtidas na base de dados Office-Caltech, utilizando o descriptor DeCAF. | 50 |
| Figura 21 – Métricas de separabilidade obtidas nas bases de dados FRGC e ARFace. . . | 51 |
| Figura 22 – Arquitetura de funcionamento do aplicativo. | 52 |
| Figura 23 – Telas elaboradas para o aplicativo. | 53 |
| Figura 24 – Captura da imagem. | 54 |
| Figura 25 – Exibição dos resultados. | 55 |

Lista de tabelas

| | |
|---|----|
| Tabela 1 – Acurárias obtidas na tarefa de classificação com e sem adaptação de domínio no problema de Reconhecimento de Objetos, utilizando o descritor SURF, sem adaptação de domínio (duas primeiras linhas) e com adaptação de domínio (cinco últimas linhas), considerando os quatro primeiros problemas de adaptação de domínio, propostos para a base de dados Office-Caltech. | 46 |
| Tabela 2 – Acurárias obtidas na tarefa de classificação com e sem adaptação de domínio no problema de Reconhecimento de Objetos, utilizando o descritor SURF, sem adaptação de domínio (duas primeiras linhas) e com adaptação de domínio (cinco últimas linhas), considerando os quatro problemas de adaptação de domínio restantes, propostos para a base de dados Office-Caltech. | 46 |
| Tabela 3 – Acurárias obtidas na tarefa de classificação com e sem adaptação de domínio no problema de Reconhecimento de Objetos, utilizando o descritor DeCAF, sem adaptação de domínio (duas primeiras linhas) e com adaptação de domínio (cinco últimas linhas), considerando os quatro primeiros problemas de adaptação de domínio, propostos para a base de dados Office-Caltech. | 47 |
| Tabela 4 – Acurárias obtidas na tarefa de classificação com e sem adaptação de domínio no problema de Reconhecimento de Objetos, utilizando o descritor DeCAF, sem adaptação de domínio (duas primeiras linhas) e com adaptação de domínio (cinco últimas linhas), considerando os quatro problemas de adaptação de domínio restantes, propostos para a base de dados Office-Caltech. | 47 |
| Tabela 5 – Acurárias obtidas no problema de Reconhecimento Facial, na tarefa de Identificação, com e sem a utilização de técnicas de Adaptação de Domínio. | 48 |

Lista de abreviaturas e siglas

| | |
|-------|---|
| 1NN | <i>1-Nearest Neighbour</i> |
| AUC | <i>Área sob a curva</i> |
| CORAL | <i>Correlation Alignment</i> |
| DeCAF | <i>Deep Convolutional Activation Features</i> |
| FRGC | <i>Face Recognition Grand Challenge</i> |
| JDA | <i>Joint Distribution Adaptation</i> |
| KNN | <i>K-Nearest Neighbour</i> |
| MMD | <i>Maximum Mean Discrepancy</i> |
| MTCNN | <i>Multi-Task Cascaded Convolutional Neural Network</i> |
| SURF | <i>Speeded-Up Robust Features</i> |
| SVHN | <i>Street View House Numbers</i> |
| SVM | <i>Support Vector Machine</i> |
| TCA | <i>Transfer Component Analysis</i> |
| TKL | <i>Transfer Kernel Learning</i> |

Sumário

| | | |
|------------|--|-----------|
| 1 | INTRODUÇÃO | 15 |
| 1.1 | Problema | 16 |
| 1.2 | Justificativa | 17 |
| 1.3 | Objetivos | 17 |
| 2 | FUNDAMENTAÇÃO TEÓRICA | 18 |
| 2.1 | Reconhecimento de Padrões | 18 |
| 2.1.1 | Métodos de Classificação | 20 |
| 2.1.1.1 | <i>K-Nearest Neighbour</i> | 20 |
| 2.1.1.2 | <i>Support Vector Machine</i> | 21 |
| 2.1.1.3 | Aprendizado de <i>Kernel</i> | 23 |
| 2.2 | Reconhecimento de Padrões Visuais | 24 |
| 2.2.1 | Reconhecimento de Objetos | 24 |
| 2.2.1.1 | Descriptor SURF | 24 |
| 2.2.1.2 | Descriptor DeCAF | 26 |
| 2.2.2 | Reconhecimento Facial | 27 |
| 2.2.2.1 | Detecção de Faces | 28 |
| 2.2.2.2 | Extração de Características | 29 |
| 2.3 | Adaptação de Domínio | 30 |
| 2.3.1 | Correlation Alignment (CORAL) | 33 |
| 2.3.2 | Transfer Kernel Learning (TKL) | 33 |
| 2.3.3 | Transfer Component Analysis (TCA) | 35 |
| 2.3.4 | Joint Distribution Adaptation (JDA) | 36 |
| 3 | METODOLOGIA | 38 |
| 3.1 | Bases de Dados | 38 |
| 3.1.1 | Reconhecimento de Objetos | 38 |
| 3.1.2 | Reconhecimento Facial | 39 |
| 3.1.2.1 | ARFace | 40 |
| 3.1.2.2 | FRGC | 41 |
| 3.2 | Experimentos | 41 |
| 4 | DESENVOLVIMENTO | 43 |
| 4.1 | Experimentos | 43 |
| 4.2 | Resultados e Discussão | 45 |
| 4.2.1 | Reconhecimento de Objetos | 45 |

| | | |
|------------|------------------------------------|-----------|
| 4.2.2 | Reconhecimento Facial | 48 |
| 5 | APLICATIVO | 52 |
| 6 | CONCLUSÃO | 56 |
| 6.1 | Trabalhos Futuros | 56 |
| | REFERÊNCIAS | 57 |

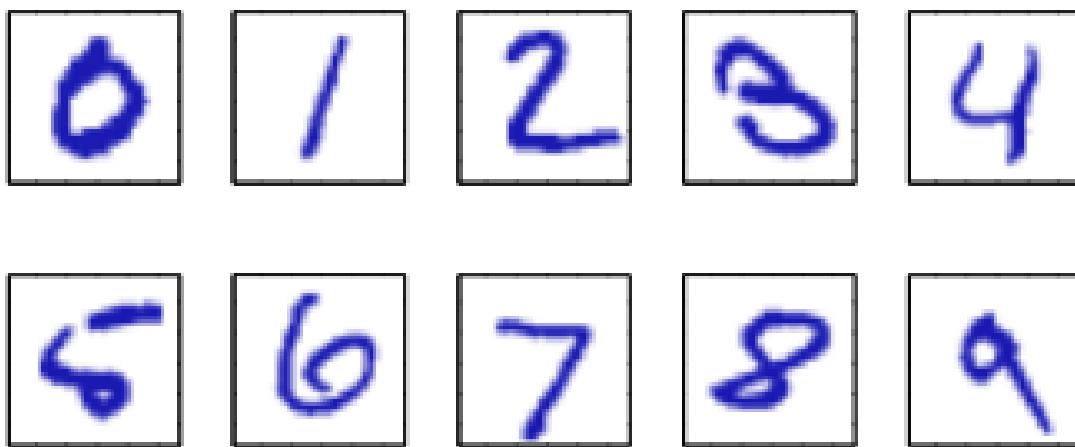
1 Introdução

Padrões estão presentes durante toda a vida de seres humanos, seja em tarefas cotidianas, como reconhecer um indivíduo através de sua face, ou de sua voz, identificar caracteres e conjuntos de palavras para realizar a leitura de um livro, ou em tarefas mais complexas, como na identificação de padrões em problemas matemáticos para se provar um teorema. Muito embora a utilização de padrões seja natural para seres humanos, a tarefa de se reconhecer padrões de maneira automática, por meio de computadores, acaba se tornando um problema complexo ([DUDA; HART; STORK, 2012](#)).

A área de reconhecimento de padrões tem o objetivo de descobrir padrões automaticamente em conjuntos de dados para que seja possível atuar sobre estes dados, classificando-os em suas categorias específicas ou predizendo valores para conjuntos de dados que ainda não foram vistos ([BISHOP, 2006](#)). Dentro desta área, tem-se o problema de reconhecimento de padrões visuais, em que, o conjunto de dados é composto por imagens. É possível citar o reconhecimento de objetos e o reconhecimento facial como problemas desta categoria.

Um exemplo de problema de reconhecimento de padrões visuais pode ser conferido na Figura 1, que contém imagens de dígitos escritos a mão, obtidas do serviço postal dos Estados Unidos. Neste caso, um sistema de reconhecimento de padrões visuais receberia uma imagem como entrada e tentaria prever a qual dígito se refere a imagem.

Figura 1 – Exemplos de imagens de dígitos escritos a mão.



Fonte: [Bishop \(2006\)](#).

Embora soluções para problemas de reconhecimento de padrões visuais não sejam triviais, a área avançou muito nos últimos tempos. O problema de reconhecimento de dígitos comentado anteriormente, por exemplo, já obtém taxas de acurácia de 99.82% ([KOWSARI et al., 2018](#)).

Mesmo com o recente desenvolvimento da área, esses resultados acabam não sendo reproduzíveis em situações reais. Como exemplo deste problema, será considerado um cenário em que um algoritmo detector de dígitos é treinado com imagens da Figura 2 e testado com imagens da Figura 1. Neste cenário, a acurácia do classificador de dígitos estaria próxima de 67.10% (LEE et al., 2019), uma acurácia bem menor em relação ao caso citado anteriormente. Caso o cenário de treino e teste estivesse invertido, essa perda de performance refletida na acurácia também estaria presente.

Figura 2 – Exemplos de imagens de dígitos obtidos de casas utilizando a ferramenta *Google Street View*.



Fonte: Netzer et al. (2011).

Esta diminuição na performance é visível porque algoritmos de classificação exigem que os dados de treino e teste sejam extraídos de uma mesma distribuição, quando isso não acontece, ocorre uma situação chamada de deslocamento de domínio (KOUW; LOOG, 2018). Uma maneira de se lidar com este problema é por meio da utilização de técnicas de adaptação de domínio, que visam utilizar um conjunto de dados diferente do domínio fonte (conjunto de dados em que o classificador é treinado), para melhorar a acurácia da classificação em um domínio alvo (conjunto de dados em que o classificador é testado) (CSURKA, 2017).

Neste Trabalho de Conclusão de Curso, foram utilizadas técnicas de Adaptação de Domínio para lidar com o problema do deslocamento de domínio em duas diferentes tarefas: Reconhecimento de Objetos e Reconhecimento Facial.

1.1 Problema

Tarefas de reconhecimento de padrões ainda apresentam problemas em aplicações reais, especialmente quando existem diferenças temporais e relativas à forma de obtenção da informação. Estes problemas ocorrem porque classificadores convencionais partem do pressuposto que a tarefa de classificação se mantém estacionária, ou seja, a classificação ocorrerá no mesmo domínio de treino (PAN; YANG, 2010).

Técnicas de adaptação de domínio surgem como uma forma de contornar o problema e tentar obter resultados de classificação mais precisos, visando melhorar o desempenho em

aplicações reais, como reconhecimento de objetos e de características biométricas (PATEL et al., 2015).

1.2 Justificativa

Tarefas de reconhecimento e categorização de padrões ganham cada vez mais relevância na sociedade atualmente, seja por motivos de automação de tarefas, aumentando a praticidade diária, ou por motivos de segurança, como por exemplo em sistemas biométricos.

Visto que esses tipos de tarefa exigem maior nível de versatilidade e uma constante melhora de desempenho em ambientes reais, o uso de técnicas de adaptação de domínio aparece como forma de contornar essa variação de domínio nesses tipos de sistema (KOUW; LOOG, 2018).

1.3 Objetivos

Objetivo Geral: Analisar o desempenho de técnicas de adaptação de domínio propostas na literatura para o problema de reconhecimento de padrões visuais.

Objetivos Específicos:

- a) Estudar técnicas de adaptação de domínio;
- b) Implementar algoritmos selecionados de reconhecimento de padrão com adaptação de domínio;
- c) Comparar o desempenho dos métodos de adaptação de domínio em tarefas de reconhecimento de padrões visuais;
- d) Desenvolver um aplicativo para dispositivos móveis para averiguação do desempenho das técnicas em ambiente real.

2 Fundamentação Teórica

A execução deste trabalho exige conhecimento de conceitos relacionados a reconhecimento de padrões e adaptação de domínio, portanto, nesta seção estão apresentados conceitos-chave referentes às áreas de reconhecimento de padrões e adaptação de domínio, incluindo informações acerca das técnicas utilizadas para as tarefas de reconhecimento de padrões visuais, bem como os métodos de adaptação de domínio utilizados no trabalho.

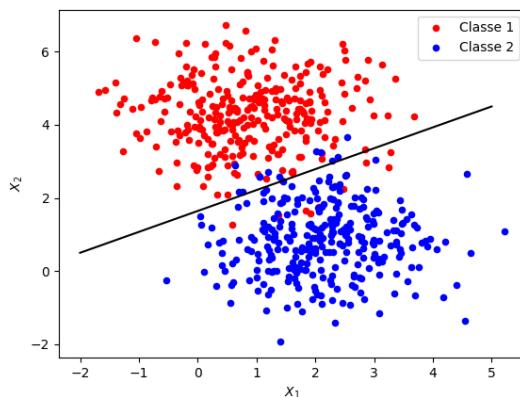
2.1 Reconhecimento de Padrões

Reconhecimento de padrões se refere ao ato de encontrar padrões de maneira automática por meio de algoritmos computacionais, de modo que seja possível atuar nos dados por meio destes padrões, assim como em um problema de classificação, em um problema visual, os dados utilizados para o reconhecimento de padrões são imagens (BISHOP, 2006).

Desta forma, o resultado de um problema de classificação pode ser expressado por meio de uma função $y(x)$, em que x representa uma imagem, ou um conjunto de características extraídos de uma imagem. Esta função de classificação é aprendida durante uma fase de treino, em que amostras de imagens são utilizadas para determinar a função $y(x)$ que consegue melhor prever a classe das imagens fornecidas como entrada (BISHOP, 2006).

Esta tarefa pode ser exemplificada por meio de um problema em que é necessário dividir um objeto com duas características em duas classes, no caso em que o espaço de características é linearmente separável, é possível separar duas classes por meio de uma função $y(x) = Ax + B$, de modo que a classe dos objetos é definida pelo posicionamento do objeto em relação à função de classificação, se está acima ou abaixo da reta (DUDA; HART; STORK, 2012). Na Figura 3, é possível ver a atuação de uma função de classificação em um espaço linearmente separável.

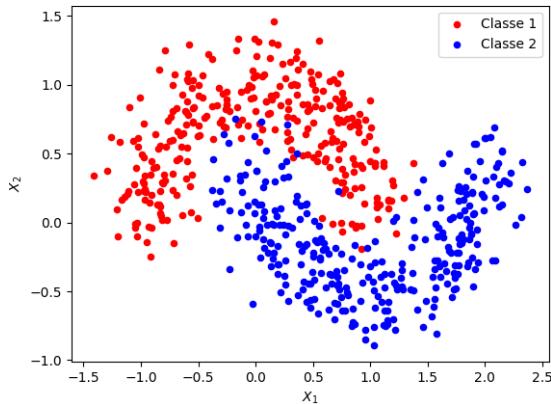
Figura 3 – Exemplo de função de classificador em um espaço linearmente separável.



Fonte: Elaborada pelo autor.

Muito embora este tipo de função seja adequada para este problema alguns outros problemas de classificação podem exigir funções mais complexas ou algum tipo de transformação que permita que o espaço passe a ser linearmente separável, como é possível perceber na Figura 4, neste caso os dados exigem que a função de classificação seja mais complexa do que uma reta (DUDA; HART; STORK, 2012).

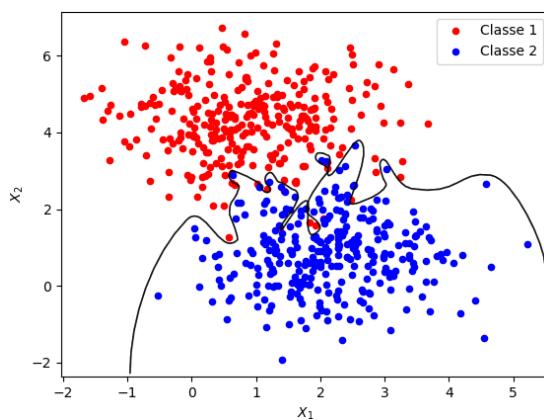
Figura 4 – Exemplo de espaço de características que não consegue ser separado linearmente.



Fonte: Elaborada pelo autor.

No que diz respeito a aplicações práticas, é importante que a função seja capaz de generalizar adequadamente para predizer amostras com características diferentes das utilizadas no treino (DUDA; HART; STORK, 2012). Se a performance do classificador funcionar apenas com os dados utilizados para o treino, existe a possibilidade de que uma situação chamada de *overfitting* tenha ocorrido, cuja função de classificação aprendida se ajusta de maneira extremamente semelhante à distribuição dos dados, como é possível observar na Figura 5, que retrata o mesmo problema de classificação da Figura 3, porém, com uma função ajustada apenas para este conjunto de dados. Neste caso, a predição de amostras que não seguem estritamente esta distribuição pode ser prejudicada, diminuindo o desempenho do sistema.

Figura 5 – Função de classificação, com *overfitting*.



Fonte: Elaborada pelo autor.

2.1.1 Métodos de Classificação

Para que os dados sejam classificados adequadamente, é necessário encontrar a função de classificação $y(x)$. Visto que muitas vezes estes sistemas trabalham com grandes quantias de dados e que os espaços de características podem atingir altas dimensões, é necessário que estas funções sejam aprendidas de maneira automática, por meio de métodos de classificação.

Neste trabalho foram utilizadas duas técnicas para a etapa de reconhecimento de padrões, *K-Nearest Neighbour* (KNN) e *Support Vector Machines* (SVM).

2.1.1.1 *K-Nearest Neighbour*

K-Nearest Neighbour (COVER; HART, 1967) é um método de classificação que, assim como descrito em seu nome, tenta classificar um objeto baseado nos k vizinhos mais próximos de sua posição no espaço de características, sendo desta forma um método baseado em métricas, cuja distância euclidiana define a classe do objeto. A função $y(x)$, portanto, é definida por:

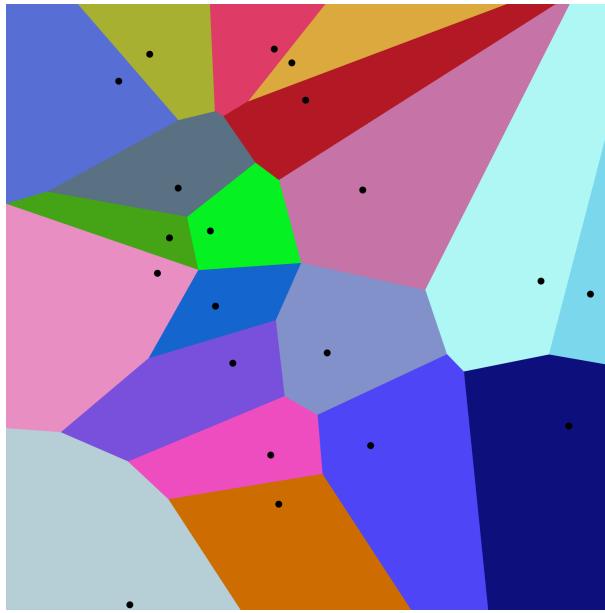
$$\hat{Y}(x) = \frac{1}{k} \sum_{x_i \in N_k(x)} y_i, \quad (2.1)$$

onde y_i se refere ao rótulo da amostra x_i no conjunto de treino, k ao número de vizinhos e $x_i \in N_k(x)$ se refere às amostras x_i no conjunto de treino, pertencentes à vizinhança, $N_k(x)$, que contém k vizinhos ao redor do dado de entrada x (HASTIE; TIBSHIRANI; FRIEDMAN, 2009). Pode-se perceber também que embora exista uma fórmula para se atribuir a classe da amostra, a função de classificação não é uma função fixa, sendo estabelecida no momento da atribuição do rótulo, já que as amostras presentes na vizinhança variam (MITCHELL, 1997).

A maneira mais simples de se abordar o método KNN é localizando somente o vizinho mais próximo, de modo que $k = 1$, neste caso, o método costuma ser chamado de 1NN. Nesta situação, o espaço de características pode ser representado pelo diagrama de Voronoi (MITCHELL, 1997), como é possível perceber na Figura 6, em que cada cor representa a região mais próxima do ponto estabelecido. No caso de um sistema de classificação baseado em 1NN, a classe de um objeto seria a mesma da amostra no conjunto de treino que define aquela região.

Classificadores baseados no KNN funcionam muito bem e partem de um conceito simples, no entanto, podem apresentar problemas em determinados cenários, visto que a atribuição de classes considera todas as características com o mesmo critério. Desta forma, um conjunto de dados contendo 5 características discriminatórias dentre um total de 50 características, pode acabar tendo seu resultado influenciado pelo ruído das outras 45 características, que podem posicionar o objeto a ser classificado próximo das amostras de outras classes (MURPHY, 2012; MITCHELL, 1997).

Figura 6 – Diagrama de Voronoi ilustrando o método 1NN, de modo que a região em que a observação se encontra determina sua classe, por meio do vizinho mais próximo.



Fonte: [Mysid e Cyp \(2015\)](#).

2.1.1.2 Support Vector Machine

Support Vector Machine ([CORTES; VAPNIK, 1995](#)) se refere a um método de classificação linear binário, cujo objetivo é encontrar um hiperplano, capaz de separar as duas classes, maximizando a distância entre a margem das mesmas.

O método parte do princípio do classificador de máxima margem, a forma mais simples de se abordar um problema de SVM. Neste caso, o objetivo é aprender as margens de cada conjunto de dados, juntamente com o hiperplano que maximiza essas margens. Sabendo que um hiperplano pode ser representado por:

$$w \cdot x + b = 0, \quad (2.2)$$

sendo w o vetor normal ao hiperplano, x um dado pertencente ao hiperplano e $b \in \mathbb{R}$. É possível escolher dois vetores de suporte, que vão dar apoio às margens que dividem os dois conjuntos de dados, de modo que:

$$\begin{aligned} w \cdot x^+ - b &= 1 \\ w \cdot x^- - b &= -1, \end{aligned} \quad (2.3)$$

onde x^+ e x^- representam os vetores de suporte das respectivas classes, ou seja, as amostras que estão na extremidade do conjunto de dados.

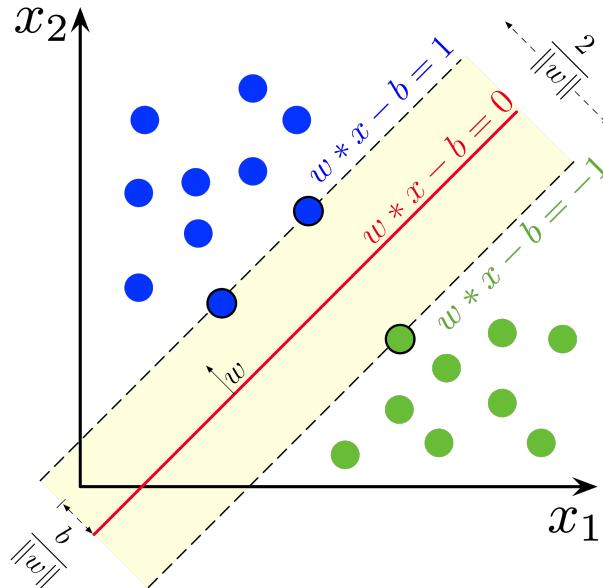
No caso em que o vetor normal ao plano está normalizado, é possível deduzir que o

tamanho máximo de margem γ equivale a $\gamma = \frac{1}{\|w\|_2}$. Desta forma, a função de classificação que maximiza a margem, com x_i representando cada uma das l amostras e y_i representando o valor de y para cada vetor de suporte (sendo 1 para valores acima do hiperplano e -1 para valores abaixo do hiperplano), passa a ser determinada por um problema que visa minimizar o tamanho da margem (CRISTIANINI; SHAWE-TAYLOR, 2000), representado por:

$$\begin{aligned} & \min_{w,b} \quad w \cdot w \\ & \text{s.a.} \quad y_i(w \cdot x_i - b) \geq 1, \\ & \quad i = 1, \dots, l. \end{aligned} \tag{2.4}$$

Na Figura 7, é possível observar uma construção do Classificador de Máxima Margem, em um espaço com duas características, utilizado para a separação de objetos em duas classes.

Figura 7 – Construção do Classificador de Máxima Margem, dividindo os objetos azuis e verdes em um espaço com duas características. É possível perceber que os elementos das extremidades servem como vetores de suporte para definir as margens.



Fonte: Larhamam (2018).

Este problema lida com uma situação chamada de *Hard-Margin*, em que o espaço de características é linearmente separável e é possível delimitar as margens sem que exista sobreposição entre os dados, no entanto, isto nem sempre é possível na prática, exigindo uma flexibilização na obtenção das margens e dos vetores de suporte, com uma variação do classificador chamada de *Soft-Margin*, na qual a SVM é construída.

Esta flexibilização aparece com a introdução de um conjunto de variáveis de folga ξ no problema de minimização, que vão relaxar a presença de ruídos no momento em que a margem é encontrada, mas que exige a presença de um fator de regularização C , para diminuir o erro

causado pela introdução de ξ . O problema de minimização passa então a ser:

$$\begin{aligned} \min_{w,b,\xi} \quad & w \cdot w + C \sum_{i=1}^l \xi_i^2 \\ \text{s.a.} \quad & y_i(w \cdot x_i - b) \geq 1 - \xi_i, \\ & i = 1, \dots, l, \\ & \xi_i \geq 0 \end{aligned} \tag{2.5}$$

A escolha do fator de regularização C que diminui o erro causado por ξ acaba introduzindo um novo problema de minimização, o que muitas vezes acaba sendo custoso, portanto, na prática, acaba-se variando um conjunto de valores para encontrar o melhor valor para um determinado cenário.

Desta forma, após a obtenção dos parâmetros w e b , é possível prever a classe de um dado x_i , observando se o resultado obtido por $w \cdot x_i - b$ é negativo ou positivo.

2.1.1.3 Aprendizado de *Kernel*

Classificadores baseados em SVM são ferramentas poderosas para lidar com espaços linearmente separáveis, no entanto, como já discutido, nem sempre é possível encontrar um hiperplano que consiga separar adequadamente um determinado conjunto de dados. Para lidar com esta situação, métodos de *Kernel* foram desenvolvidos.

Neste caso, uma função de transformação é aplicada no conjunto de dados, fazendo com que os dados sejam posicionados em outro espaço de característica, em que o conjunto de dados é linearmente separável, permitindo a aplicação de técnicas como a SVM ([SHawe-Taylor; Cristianini et al., 2004](#)).

Funções de *Kernel* partem do conceito de espaços de Hilbert, espaços vetoriais construídos de modo que, as operações de produto interno estão bem definidas, generalizando o conceito de produto escalar ([Hofmann; Scholkopf; Smola, 2008](#)). Sabendo disso, [Shawe-Taylor, Cristianini et al. \(2004\)](#) definem uma função de *Kernel* como uma função do tipo $k : X \times X \rightarrow \mathcal{H}$.

A área de aprendizado de *Kernel* também introduziu o conceito de *Kernel Trick*, que afirma que é possível calcular o *Kernel* de um conjunto de dados baseado no produto interno de suas transformações, ao invés de se calcular sua posição no espaço, o que diminui o custo computacional da operação. Desta forma, a função de transformação passaria a ser $k(x_i, x_j) = \phi(x_i) \cdot \phi(x_j)$, sendo $\phi : X \rightarrow \mathcal{H}$ ([Goodfellow; Bengio; Courville, 2016](#)).

Esta transformação permite que seja possível representar todo o conjunto de dados por meio de uma matriz de *Kernel*, de modo que cada elemento K_{ij} da matriz seja obtido pela equação $K_{ij} = k(x_i, x_j)$. Além disso, por mais que o conjunto de dados X tenha uma distribuição não linear, a função de classificação $y = w \cdot k(x, x_i) - b$, ainda é regida por

uma relação linear, e com os dados transformados pela função k em um espaço linearmente separável, podemos classificar até mesmo espaços de características complicados, aumentando o poder de classificação de métodos como a SVM.

2.2 Reconhecimento de Padrões Visuais

A área de reconhecimento de padrões lida com variados tipos de problemas, podendo ser usada para detecção de doenças, análise de créditos ou processamento de sinais advindos de diversas fontes, como sinais de áudio ou sinais de movimentação obtidos de um sensor.

Um desses tipos é o reconhecimento de padrões visuais, que lida com padrões visuais, ou seja, imagens. Neste trabalho, esta subcategoria foi utilizada para análise, com foco em dois sub-problemas: Reconhecimento de Objetos e Reconhecimento Facial, descritos a seguir.

2.2.1 Reconhecimento de Objetos

De acordo com [Biederman \(1995\)](#), o ser humano é capaz de reconhecer e categorizar mais de dez mil tipos de objetos. Com o desenvolvimento de sensores de fácil acesso, como câmeras em celulares, o número de imagens de objetos começou a crescer, levando ao surgimento de novas bases de dados e do desenvolvimento da tarefa de classificar objetos automaticamente.

Recentemente, novas aplicações que se beneficiam do reconhecimento de objetos foram surgindo, um exemplo de aplicação desta tarefa, que vem ganhando popularidade é no desenvolvimento de sistemas de carro autônomos ([REDMON et al., 2016](#)).

A categorização de objetos, no entanto, não é uma tarefa fácil, em aplicações reais, é necessário detectar o objeto em tempo real, além disso, com o objeto detectado, são necessários métodos para processar as imagens e realizar definitivamente a categorização dos objetos ([FERGUS, 2005](#)).

Muitas vezes, um conjunto de características é extraído das imagens, encontrando uma representação mais adequada para os dados e diminuindo o número de informações necessárias nesta representação, exigindo menos espaço de armazenamento para a base de dados. Esta etapa é chamada de extração de características. Durante este trabalho, duas características foram consideradas para a tarefa de reconhecimento de objetos, *Speeded-Up Robust Features* (SURF) e *Deep Convolutional Activation Features* (DeCAF).

2.2.1.1 Descritor SURF

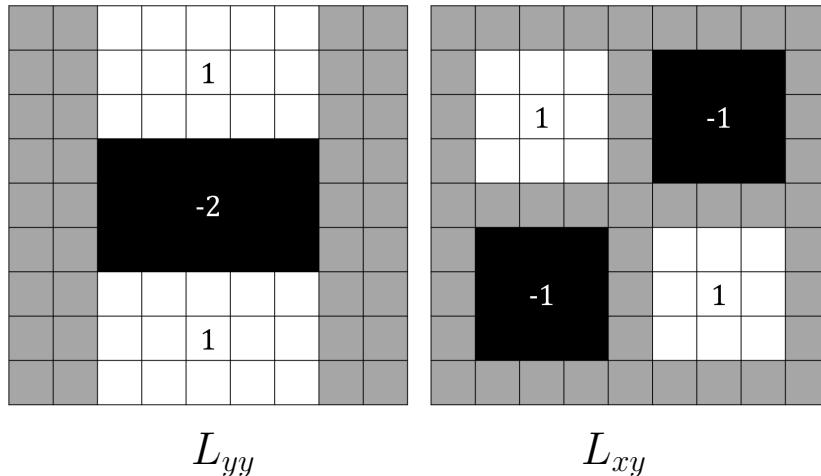
SURF ([BAY et al., 2008](#)) é um descritor de imagens conhecido por ser rápido o suficiente para trabalhar em aplicações de tempo real e ser variante tanto a escala quanto a rotação. Para atingir este objetivo, o descritor passa por duas etapas: detecção dos pontos-chave e descrição dos pontos chave.

Para a detecção dos pontos chave, o método utiliza como base o determinante da matriz Hessiana, de modo que, para cada ponto $P = (x, y)$ presente na imagem de entrada, a matriz Hessiana $\mathcal{H}(P, \sigma)$ é definida da seguinte forma:

$$\mathcal{H}(P, \sigma) = \begin{bmatrix} L_{xx}(P, \sigma) & L_{xy}(P, \sigma) \\ L_{xy}(P, \sigma) & L_{yy}(P, \sigma) \end{bmatrix}, \quad (2.6)$$

de modo que L_{ij} representa a convolução com filtros da derivada Gaussiana de segunda ordem na imagem, ao redor do ponto P , sendo i e j o eixo na qual a derivada é aplicada. Na Figura 8, pode-se observar exemplos dos filtros de convolução utilizados.

Figura 8 – Exemplo dos filtros de convolução L_{yy} (esquerda) e L_{xy} (direita) usados para o cálculo da matriz Hessiana no método SURF.



Fonte: Bay et al. (2008).

Em seguida, a análise de escala é feita variando a dimensão dos filtros L aplicados na Hessiana, partindo do tamanho 9×9 e incrementando em 6 unidades, até filtros de tamanho 27×27 .

O determinante da matriz Hessiana detectado passa a representar uma resposta da imagem ao redor do ponto P , desta forma, um conjunto de máximos locais é selecionado ao longo das diferentes escalas de filtros, este conjunto representa os pontos de interesse. Para garantir a invariância a rotação, a imagem é rotacionada na direção que possui o maior número de pontos de interesse (OYALLON; RABIN, 2015).

Para a criação do descritor, uma região retangular é selecionada ao redor dos pontos de apoio selecionados. Esta região é então sub-dividida em regiões menores de tamanho 4×4 , nas quais a transformada de Haar é aplicada, nos eixos x e y , considerando também os valores absolutos da transformada. O resultado é um vetor de 4 dimensões, contendo a soma de todas as transformadas, para cada uma das 16 regiões, resultando em 64 características por região.

O método também pode se beneficiar do conceito de *Bag of Visual Words* tentando encontrar uma representação mais compacta dos dados. Neste caso, é criado um vocabulário de

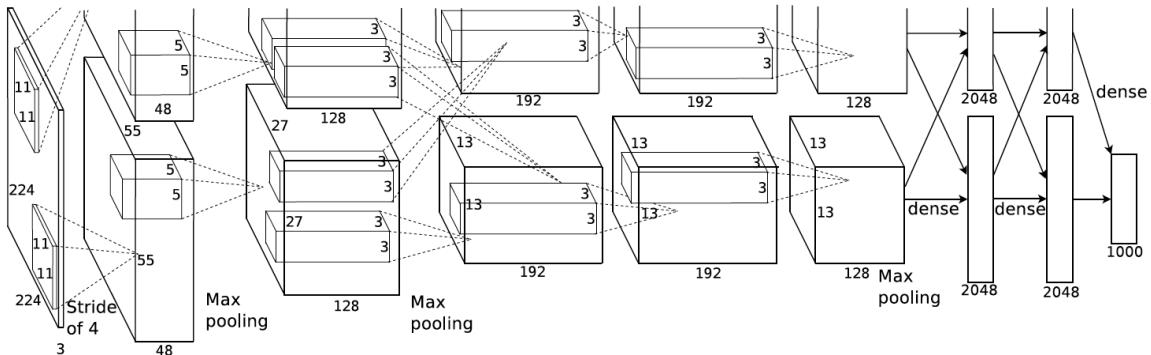
características, baseadas no descriptor SURF, de modo que, as imagens possam ser representadas pelo histograma deste vocabulário de características (FAROOQ, 2016; SAENKO et al., 2010).

2.2.1.2 Descritor DeCAF

Nas últimas décadas, além da extração de características de maneira tradicional, pela utilização de métodos como o SURF, métodos baseados em redes convolucionais vieram ganhando popularidade, através das características em profundidade, para o reconhecimento de objetos, um desses métodos é o DeCAF (DONAHUE et al., 2014).

Este método foi proposto como um *framework* de extração de características para redes convolucionais, aliando uma rede convolucional que apresentava bons resultados na literatura, por meio da arquitetura proposta por Krizhevsky, Sutskever e Hinton (2017) para o desafio *ImageNet*, com um protocolo de análise da discriminabilidade de características por meio de visualização de dados. Na Figura 9 pode-se observar a arquitetura utilizada no treino.

Figura 9 – Arquitetura da rede convolucional usada como base para a seleção de características e treino do DeCAF.

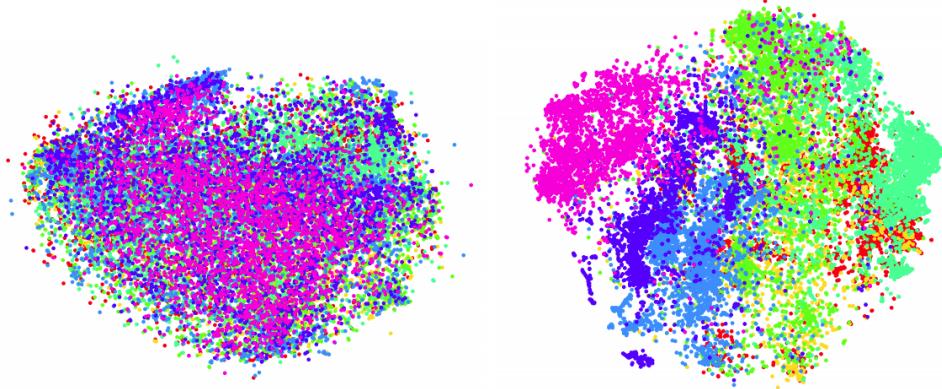


Fonte: Krizhevsky, Sutskever e Hinton (2017).

Desta forma, a extração de características por meio do DeCAF pode ser feita por meio do protocolo DeCAF, que consiste em realizar um *fine-tuning* da rede convolucional, observada na Figura 9, seguida da análise de quais camadas da rede oferecem melhor desempenho, através da observação de um *cluster* de dados, que permite escolher qual a melhor camada para atuar como extrator de características. Na Figura 10 estão presentes exemplos de *clusters* obtidos para análise da arquitetura, à esquerda, estão presentes dados extraídos da primeira camada da rede, enquanto à direita, os dados se referem à sexta camada da rede, mostrando uma maior discriminabilidade, facilitando a escolha da característica adequada.

A utilização do DeCAF como protocolo de treino se mostra eficaz para tarefas que exigem generalização do conjunto de dados, até mesmo em cenários complexos, em que a presença de deslocamento de domínio é observada.

Figura 10 – *Clusters* de dados utilizados durante o protocolo DeCAF para auxiliar na escolha das características. De modo que os dados presentes à esquerda, extraídos da primeira camada da rede convolucional apresentada na Figura 9, enquanto os dados presentes à direita foram extraídos da sexta camada da rede.



Fonte: [Donahue et al. \(2014\)](#).

2.2.2 Reconhecimento Facial

Reconhecimento facial pode ser definido como o processo de se estabelecer a identidade de um indivíduo baseando-se em suas características faciais, sendo uma das formas mais naturais de se identificar indivíduos utilizadas pelos seres humanos ([JAIN; LI, 2011](#)).

Este tipo de identificação é tão natural para seres humanos, que pesquisas na área neurocognitiva indicam que determinadas áreas do cérebro possuem um enfoque no reconhecimento de faces, esta facilidade de assimilação para a população, juntamente do fato de que sistemas de reconhecimento facial podem atuar de forma não intrusiva, fazem com que este formato de identificação biométrica seja um dos mais aceitos pela sociedade ([JAIN; ROSS; NANDAKUMAR, 2011](#)).

Assim como outras formas de reconhecimento biométrico, existem duas formas de se atuar em um sistema de reconhecimento facial, podendo este ser um sistema focado em **autenticação**, ou um sistema de **reconhecimento**.

A diferença entre os dois sistemas consiste no fato de que sistema de autenticação fazem uma comparação do tipo 1 : 1, ou seja, comparam duas imagens, ou características de indivíduos, e estabelecem se aquela pessoa é quem realmente diz ser, é comum que, quando estejamos lidando com um problema de autenticação, exista uma métrica de similaridade entre as imagens, que vai ditar o quanto parecidos são os dois indivíduos. Partindo desta similaridade, estabelece-se um limiar, métricas obtidas acima deste limiar são consideradas como comparações genuínas, enquanto que métricas que não atingem o limiar são comparações impostoras.

Por outro lado, sistemas de reconhecimento trabalham com comparações do tipo 1 : N , ou seja, seu objetivo passa a ser, baseado em uma imagem, ou característica, categorizar qual é o indivíduo desejado. Esta análise pode partir de um método de classificação comum, ou

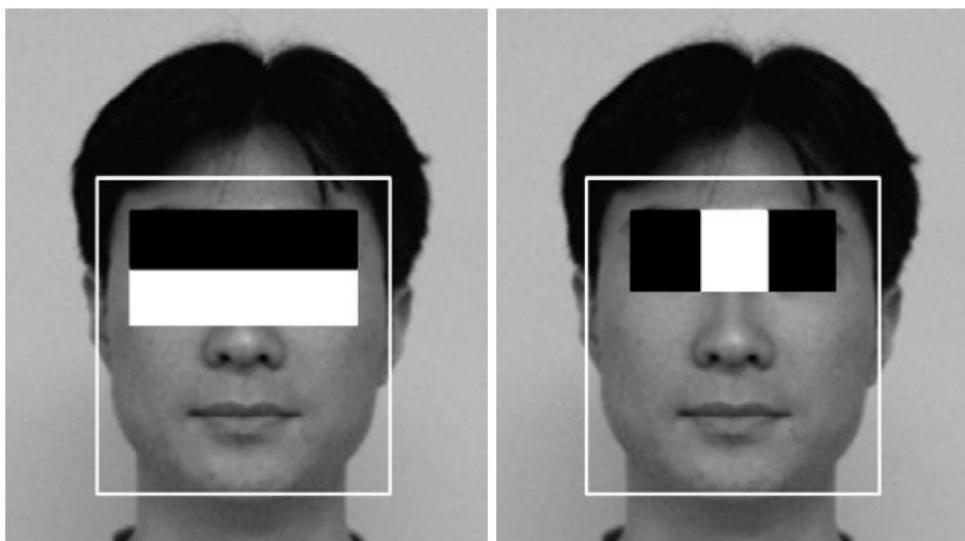
podem ser comparadas imagens entre todos os indivíduos de uma base de dados, escolhendo a imagem com maior métrica de similaridade.

2.2.2.1 Detecção de Faces

Para que um sistema de reconhecimento facial funcione adequadamente, é necessário que a região da face esteja delimitada para o início da análise, existem diversas formas de se fazer isso, sendo uma das mais comuns, por meio do método proposto por [Viola e Jones \(2004\)](#).

Este método utiliza um conjunto de contrastes orientados, baseados nas características de Haar, para encontrar possíveis regiões de face, por meio da análise em janelas de diferentes tamanhos ao longo da imagem. Na Figura 11 é possível observar a presença dos dois contrastes sobrepostos em uma face durante o processo de detecção facial.

Figura 11 – Contrastos de Haar utilizados para a detecção de faces, no método proposto por [Viola e Jones \(2004\)](#).



Fonte: [Jain, Ross e Nandakumar \(2011\)](#).

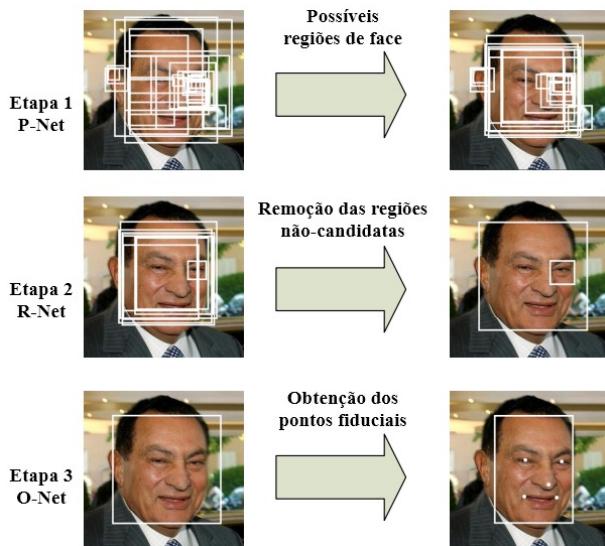
Embora o método de Viola-Jones apresente bons resultados para detecção de faces em cenários controlados, com o aumento da necessidade de aplicações de reconhecimento facial, novas soluções passaram a ser exigidas, de modo a lidar com problemas que surgem quando as imagens em que é preciso detectar a face possuem variações de iluminação, pose, oclusões na região da face, ou fundos muito complexos. No entanto, com avanços na área de redes convolucionais, surgiram métodos como o MTCNN (Multi-Task Cascaded Convolutional Neural Network) ([ZHANG et al., 2016](#)), que conseguem oferecer soluções mais robustas para a detecção de faces em ambientes pouco controlados.

O método de detecção de faces proposto por [Zhang et al. \(2016\)](#) utiliza um conjunto de três redes neurais em cascata, para realizar a detecção da região da face em três diferentes

estágios, filtrar as possíveis regiões faciais e obter informações referentes ao alinhamento da face, por meio de pontos fiduciais, relativos aos olhos e a boca.

Para isso, a rede utiliza três redes convolucionais de modo que a saída de uma das redes é imediatamente utilizada como entrada para a outra. No primeiro estágio uma P-NET é utilizada para prever as regiões de face prováveis, após isso, uma R-NET é utilizada para filtrar as regiões não-candidatas. Por fim uma O-NET é utilizada para obter os pontos fiduciais da face. O *pipeline* do método pode ser conferido na Figura 12.

Figura 12 – *Pipeline* do método de detecção de faces MTCNN.



Fonte: [Zhang et al. \(2016\)](#).

2.2.2.2 Extração de Características

Após a detecção da região da face, é importante que uma etapa de extração de características seja realizada, a fim de reduzir a dimensionalidade dos dados e encontrar uma representação mais discriminativa para as faces.

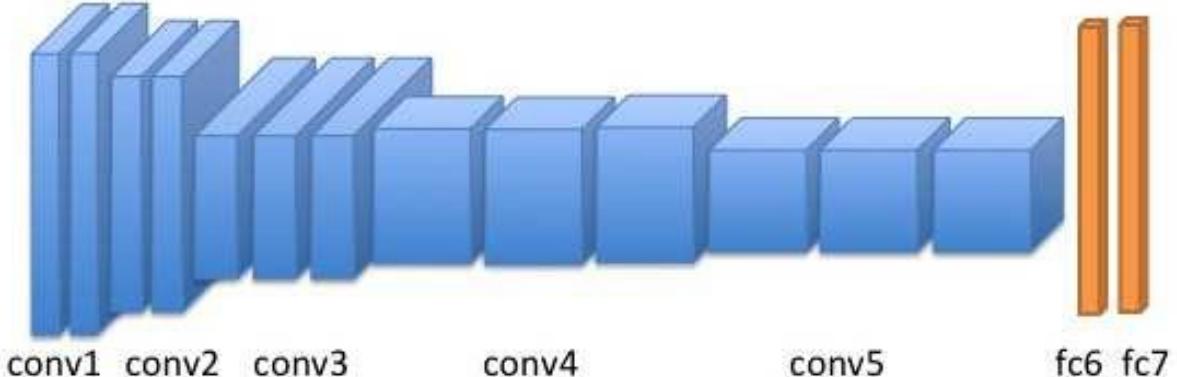
Da mesma forma que na etapa de detecção, a extração de características também se beneficiou do uso de redes convolucionais, por meio da utilização de características profundas.

Um exemplo de características profundas é o obtido através da rede convolucional VGG-Face ([PARKHI; VEDALDI; ZISSERMAN, 2015](#)), uma rede baseada na arquitetura VGG-16 ([SIMONYAN; ZISSERMAN, 2014](#)), que pode ser conferida na Figura 13.

Durante o desenvolvimento desta rede convolucional, uma base de dados foi criada para ser utilizada durante a etapa de treino, contendo mais de 2.6 milhões de imagens faciais, e um total de 2622 indivíduos.

Para melhor aproveitamento da rede, sua execução foi pensada na generalização da tarefa de reconhecimento facial, por meio da divulgação de modelos pré-treinados e da criação

Figura 13 – Arquitetura da rede VGG Face.



Fonte: [Nakada, Wang e Terzopoulos \(2017\)](#).

de um protocolo de *fine-tuning*, que facilita a utilização da rede em situações de *transfer learning*. Desta forma, para realizar a extração de características, basta introduzir a imagem na rede e obter a saída na camada totalmente conectada 7 (fc7), obtendo um total de 4096 características que podem ser utilizadas tanto em ambas as tarefas de identificação e autenticação.

2.3 Adaptação de Domínio

Assim como discutido na Seção 2.1, é importante que um sistema de classificação consiga generalizar bem em situações que não foram previstas, no entanto, classificadores podem ter problemas em lidar com isso quando as distribuições do conjunto de treino e de teste apresentam diferenças, isso ocorre devido a um problema chamado deslocamento de domínio, que acaba influenciando no desempenho de um classificador, fazendo com que imagens sejam classificadas incorretamente ([KOUW; LOOG, 2018](#)).

O deslocamento de domínio ocorre quando as distribuições do domínio fonte (utilizado para o treino) e do domínio alvo (utilizado para o teste) são diferentes. Uma forma de se lidar com este problema é por meio de técnicas de adaptação de domínio ([PATEL et al., 2015](#)). Na Figura 14 é possível observar uma situação em que há presença do deslocamento de domínio comparando imagens faciais de documentos de identidade e *selfies*. Neste caso, variações de pose, iluminação, sensor de captura e complexidade do fundo, podem acabar influenciando negativamente na acurácia de um sistema.

Adaptação de Domínio refere-se a uma sub-área da transferência de aprendizado, com o objetivo de utilizar dados de um domínio distinto do utilizado no treino, visando melhorar o desempenho do classificador quando aplicado a um outro conjunto de dados ([CSURKA, 2017](#)). Para que isso seja feito de maneira adequada, é necessário que ambos os domínios tratem do mesmo problema, ou seja, possuam o mesmo conjunto de rótulos.

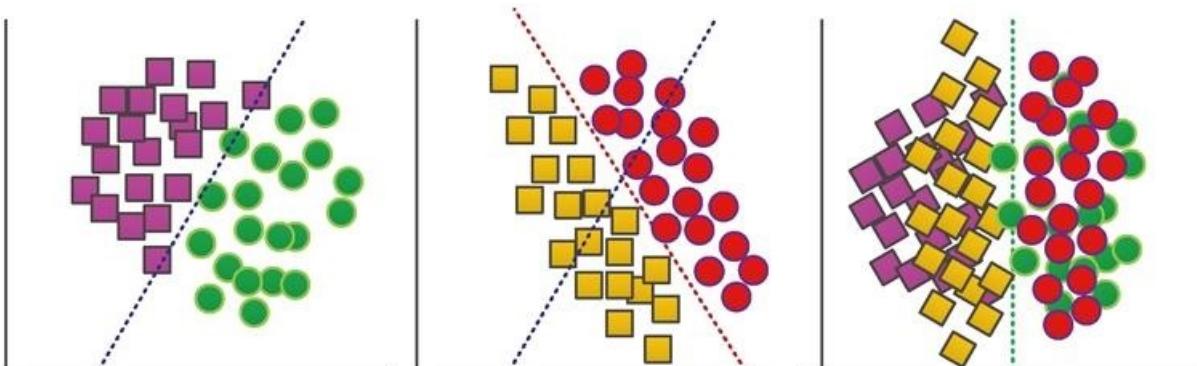
Figura 14 – Exemplos de imagens de faces obtidas de quatro pessoas no domínio caracterizado pelas fotos de documentos de identidade (linha superior) e obtidas no domínio caracterizado pelas *selfies* (linha inferior).



Fonte: Elaborada pelo autor.

Na Figura 15, é possível observar a importância da adaptação de domínio no problema de deslocamento de domínio, no segundo quadro, quando comparamos o classificador do domínio fonte (traçado em azul), com o classificador do domínio alvo (traçado em vermelho), podemos observar o efeito negativo de colocar dados de uma distribuição diferente para serem avaliados em um classificador treinado em outro domínio. Utilizando adaptação de domínio é possível lidar com este problema e encontrar um domínio comum entre os dois conjuntos de dados, com um classificador adequado.

Figura 15 – Exemplificação de um problema de deslocamento de domínio. No primeiro gráfico estão presentes os dados no domínio fonte, com seu classificador traçado em azul. No segundo gráfico, os dados do domínio alvo, com seu classificador traçado em vermelho. No terceiro quadro, os dados de ambos os domínios após a adaptação de domínio, com um classificador treinado no domínio comum.



Fonte: (CHAI et al., 2018)

As definições formais dos conceitos necessários para entender a adaptação de domínio, segundo Pan e Yang (2010) estão expostas a seguir.

Definição 2.3.1 (Domínio) Um domínio \mathcal{D} é composto por um espaço de características \mathcal{F} com d dimensões e uma função de distribuição de probabilidade marginal $P(x)$, ou seja, $\mathcal{D} = \{\mathcal{F}, P(x)\}$ com $x \in \mathcal{F}$.

Definição 2.3.2 (Tarefa) Dado um domínio \mathcal{D} , uma tarefa \mathcal{T} consiste de um conjunto de rótulos \mathcal{Y} e um classificador $f(x)$, ou seja, $\mathcal{T} = \{\mathcal{Y}, f(x)\}$, em que $y \in \mathcal{Y}$ e $f(x) = P(y|x)$.

Definição 2.3.3 (Adaptação de Domínio) Dado um domínio fonte \mathcal{D}_S e um domínio alvo \mathcal{D}_T e assumindo que $\mathcal{D}_S \neq \mathcal{D}_T$ no que se diz respeito a probabilidade marginal $P(X^S) \neq P(X^T)$, ou seja, há variação no domínio, e duas tarefas $\mathcal{T}_S \neq \mathcal{T}_T$, com distribuição condicional $P(Y^S|X^S) \approx P(Y^T|X^T)$. O objetivo da adaptação de domínio é melhorar a função de predição $f_T(\cdot)$ no domínio alvo \mathcal{D}_T , usando o conjunto de dados presente no domínio fonte \mathcal{D}_S .

Em linguagem natural, isso significa que um domínio é um conjunto de dados pertencentes a um espaço de características \mathcal{F} , distribuído por uma função $P(x)$, enquanto que uma tarefa é o ato de atribuir rótulos aos elementos deste domínio, por meio de uma função $f(x)$. Desta forma, o objetivo da adaptação de domínio passa a ser melhorar a característica preditiva de um domínio alvo, em um classificador treinado com dados do domínio fonte.

É possível categorizar os métodos de adaptação de domínio de duas formas, quanto ao tipo de característica utilizada na classificação e quanto à presença de rótulos no domínio alvo.

No que diz respeito ao tipo de características, pode-se obter os seguintes tipos de situações:

Adaptação de Domínio Homogênea: Quando o espaço de características é o mesmo no Domínio Fonte e no Domínio Alvo.

Adaptação de Domínio Heterogênea: Quando o espaço de características difere entre o Domínio Fonte e o Domínio Alvo.

Já quando as técnicas são classificadas quanto a presença de rótulos, as seguintes categorizações são obtidas:

Adaptação de Domínio Supervisionada: Neste caso, o domínio alvo possui todo o conjunto de dados rotulado.

Adaptação de Domínio Semi-Supervisionada: Utiliza dados rotulados e não rotulados do domínio alvo na tarefa de adaptação.

Adaptação de Domínio Não-Supervisionada: Não utiliza rótulos do domínio alvo na tarefa de adaptação.

Neste trabalho, foram abordados quatro técnicas de adaptação de domínio, sendo todas elas homogêneas e não-supervisionadas.

2.3.1 Correlation Alignment (CORAL)

Uma primeira abordagem para adaptação de domínio é oferecida pelo método CORAL ([SUN; SAENKO, 2016](#)), este método aparece de três diferentes formas, como substituto da análise de discriminante linear, como função de erro para redes em profundidade e, em sua forma mais simples, chamada de CORAL Linear, que foi utilizada neste trabalho.

O CORAL Linear é um método de adaptação de domínio chamado de "*frustratingly easy*", por ser elaborado partindo de um conceito simplista. O método aborda um sub-problema da adaptação de domínio, chamado de desvio de covariância, pressupondo que existe uma relação de dependência linear entre a covariância dos dois domínios.

Portanto, o objetivo do método é alinhar a covariância dos dados, de modo a minimizar a diferença entre a covariância dos dois domínios, para isso, transforma os dados do domínio fonte, para que possuam a mesma covariância dos dados do domínio alvo. Esta transformação ocorre em duas etapas, a primeira chamada de esbranquiçamento, que consiste em "descorrelacionar" os dados do domínio fonte, a partir da seguinte relação: $W_S = X_S \Sigma_S^{+\frac{1}{2}} + \lambda I$, sendo X_S o conjunto de dados do domínio fonte normalizado pela média e desvio padrão, I a matriz identidade, λ um parâmetro de regularização e Σ_S^+ a pseudo-inversa da matriz de covariância do domínio alvo.

Após o esbranquiçamento dos dados do domínio fonte, uma etapa cujo objetivo é "re-correlacionar" os dados do domínio fonte, com a covariância do domínio alvo é iniciada, por meio da expressão: $A_S = W_S \Sigma_T^{-\frac{1}{2}} + \lambda I$, sendo Σ_T a matriz de covariância do domínio alvo.

Embora este seja um método simples e fácil de ser calculado, apresenta problemas, pelo fato de que nem sempre a correlação dos dados é o fator determinante para o problema de deslocamento de domínio e, mesmo que exista influência da covariância nem sempre ela é satisfeita por uma relação linear. No entanto, esta versão do método serve para exemplificar a importância de relações estatísticas bem definidas para a área de adaptação de domínio.

2.3.2 Transfer Kernel Learning (TKL)

O método TKL ([LONG et al., 2015](#)), é um método que tem se mostrado promissor na área de adaptação de domínio em diferentes tarefas, com destaque nas tarefas de reconhecimento de padrão visual. De modo geral, o método busca encontrar um *kernel* de domínio invariante, ou seja, um *kernel* que minimiza a variação da distribuição entre os domínios, de modo que,

quando alimentado em uma máquina de *kernel*, consiga realizar a adaptação de domínio. No Problema 1 está contida a definição formal do método.

Problema 1 (Transfer Kernel Learning) *Dado um domínio fonte rotulado*

$\mathcal{Z} = \{(z_1, y_1), \dots, (z_m, y_m)\}$ e um domínio alvo não rotulado $\mathcal{X} = \{x_1, \dots, x_n\}$, com $\mathcal{F}_{\mathcal{Z}} = \mathcal{F}_{\mathcal{X}}$, $\mathcal{Y}_{\mathcal{Z}} = \mathcal{Y}_{\mathcal{X}}$, $P(z) \neq P(x)$ e $P(y|z) \neq P(y|x)$, precisamos aprender um kernel de domínio invariante $k(z, x) = \langle \phi(z), \phi(x) \rangle$, de modo que $P(\phi(z)) \approx P(\phi(x))$. Assumindo que $P(y|\phi(z)) \approx P(y|\phi(x))$ para que seja possível que os kernels treinados em \mathcal{Z} sejam generalizados adequadamente para \mathcal{X} .

O TKL é baseado no princípio de que muitas vezes, métricas de minimização como o MMD (*Maximum Mean Discrepancy*) são utilizadas somente como função de penalização, sem levar em conta a variação entre as distribuições, desta forma não conseguem sempre atingir um ponto de mínimo local adequado.

Para resolver o problema proposto, Long et al. (2015) utilizam o Teorema de Mercer (LONG et al., 2015), para extrapolar os autovetores do domínio fonte, por meio dos autovetores do domínio alvo, através da seguinte expressão:

$$\bar{\Phi}_Z \simeq K_{ZX}\Phi_X\Lambda_X^{-1}, \quad (2.7)$$

sendo $\bar{\Phi}_Z$ os autovetores extrapolados do domínio fonte, K_{ZX} a aplicação de uma função *kernel* nos dados domínio fonte em relação ao domínio alvo, Φ_X os autovetores calculados no *kernel* do domínio alvo e Λ_X , os autovalores, calculados no *kernel* do domínio alvo.

Em seguida, utiliza-se a aproximação pelo *Kernel* de Nyström (WILLIAMS; SEEGER, 2001) para obter um *kernel* \bar{K}_Z do domínio fonte aproximado pelos dados do domínio alvo, através da seguinte expressão:

$$\bar{K}_Z = \bar{\Phi}_Z\Lambda\bar{\Phi}_Z^T. \quad (2.8)$$

Este *kernel* é capaz de se aproveitar das estruturas internas do domínio alvo, através dos autovetores extrapolados, no entanto, para melhor representação, é necessário minimizar a distância entre o *kernel* original e o extrapolado. Para isso, os valores de Λ são relaxados em um problema de minimização, que visa minimizar a distância entre os dois *kernels*, elaborado de acordo com a seguinte expressão:

$$\begin{aligned} \min_{\Lambda} \|\bar{K}_Z - K_Z\|_F^2 &= \|\bar{\Phi}_Z\Lambda\bar{\Phi}_Z^T - K_Z\|_F^2 \\ \text{s.a.} \quad \lambda_i &\geq \zeta\lambda_{i+1}, i = 1, \dots, n-1, \\ \lambda_i &\geq 0, i = 1, \dots, n. \end{aligned} \quad (2.9)$$

A presença de ζ como um parâmetro passível de otimização é essencial para o desempenho do problema, que pode ser reduzido a um problema de minimização quadrático para melhora de performance. Após encontrar os valores de λ que minimizam a distância entre os domínios, um novo *kernel* de domínio invariante pode ser calculado por meio da seguinte equação:

$$\overline{K}_A = \begin{bmatrix} \overline{\Phi}_Z \Lambda \overline{\Phi}_Z^T & \overline{\Phi}_Z \Lambda \Phi_X^T \\ \Phi_X \Lambda \overline{\Phi}_Z^T & \Phi_X \Lambda \Phi_X^T \end{bmatrix}. \quad (2.10)$$

Desta forma, é possível treinar uma SVM alimentando os dados do domínio fonte ajustados, por meio da expressão: $\overline{\Phi}_Z \Lambda \overline{\Phi}_Z^T$, e avaliar o desempenho dos resultados por meio dos dados do domínio alvo, ajustados em relação ao domínio fonte extrapolado, por meio da expressão: $\Phi_X \Lambda \overline{\Phi}_Z^T$.

2.3.3 Transfer Component Analysis (TCA)

O TCA (PAN et al., 2010) é uma técnica de adaptação de domínio cujo objetivo é aprender uma transformação $\phi(\cdot)$ para um novo espaço de características onde a métrica de MMD entre os domínios fonte e alvo é minimizada.

A métrica de MMD é uma medida que visa obter uma aproximação da distância entre duas distribuições, sem que a amostragem da distribuição seja necessária, por meio da utilização de suas médias. Desta forma, o cálculo da métrica de MMD em um espaço de características H , após a aplicação da função $\phi(\cdot)$, em duas distribuições, X e Y passa a ser:

$$Dist_{MMD}(X, Y) = \left\| \frac{1}{n_1} \sum_{i=1}^{n_1} \phi(x_i) - \frac{1}{n_2} \sum_{i=1}^{n_2} \phi(y_i) \right\|_H. \quad (2.11)$$

Sendo K a matriz de *kernel* dos dados do domínio fonte e alvo, a expressão do cálculo da distância MMD pode ser reduzida a $Dist_{MMD}(X, Y) = tr(KL)$. De modo que L é uma matriz de MMD, calculada de modo que $L_{ij} = \frac{1}{n_1^2}$, se x_i e x_j pertencem ao domínio fonte; $L_{ij} = \frac{1}{n_2^2}$ se x_i e x_j pertencem ao domínio alvo; e $L_{ij} = -\frac{1}{n_1 n_2}$ do contrário.

Como o objetivo é encontrar uma transformação para os dados do domínio fonte, é necessário encontrar uma matriz W que mapeie os dados de K em novo espaço de características. Pelas propriedades de decomposição de matriz de *kernel*, o novo espaço de características pode ser encontrado por meio da expressão $\tilde{K} = KWW^T K$.

Desta forma, considerando a forma matricial da Equação 2.11, juntamente de um fator de regularização $tr(W^T W)$, o problema pode ser resolvido minimizando a seguinte expressão:

$$\begin{aligned} \min_W & tr(W^T W) + \mu tr(W^T K L K W) \\ \text{s.a.} & W^T K H K W = I, \end{aligned} \quad (2.12)$$

de modo que H é uma matriz de centralização, tal que $H = I - \frac{1}{n_s + n_t} \mathbf{1}\mathbf{1}^T$ e $\mathbf{1}$ representa o vetor coluna unitário.

Este problema de minimização pode ser reformulado, visando obter um conjunto dos autovetores com mais impacto, de maneira similar à análise dos componentes principais, através da seguinte expressão:

$$W = (KLK + \mu I)^{-1} KHK \quad (2.13)$$

Desta forma, é feita a seleção dos m autovetores com maiores autovalores, e utilizá-los para realizar a transformação do novo conjunto de dados $X^* = KW$. É importante notar que, por meio de manipulações algébricas na maneira como se encontra W , é possível utilizar o método sem a necessidade de uma transformação de *kernel* K , na chamada, forma primal do método (LONG et al., 2013).

2.3.4 Joint Distribution Adaptation (JDA)

O método JDA (LONG et al., 2013) surge como uma extensão do TCA, partindo do pressuposto que, embora o TCA consiga minimizar a distribuição marginal dos dados e encontrar um mínimo local entre os domínios, o método negligencia a distribuição condicional.

Desta forma o JDA opera da mesma forma que o TCA, com somente duas modificações na execução do método, afim de minimizar a diferença entre as distribuições condicionais: análise da distância entre as distribuições passa a ser feita por classe ao invés de maneira global, e acrescenta-se um processo iterativo de geração de pseudo-rótulos no domínio fonte, para deixar a análise por classes na primeira parte do processo mais precisa.

Inicialmente, o método realiza uma iteração da mesma forma que o TCA, obtendo um conjunto de dados ajustados, em seguida, utiliza um classificador auxiliar para gerar um conjunto de pseudo-rótulos baseado nos resultados da classificação. A partir deste rótulos, inicia-se a análise das distribuições condicionais por classe.

Para isso, todo o conjunto de dados D é dividido em subconjuntos $D^{(c)}$, compostos somente por elementos de D pertencentes à classe c . A partir desta divisão, o cálculo da distância MMD, na forma matricial passa a ser obtido por $tr(W^T KL_c KW)$, onde L_c representa um conjunto de matrizes de MMD, calculadas por classe, de modo que, $\frac{1}{n_s^{(c)} n_s^{(c)}}$ se $x_i, x_j \in D_s^{(c)}$; $\frac{1}{n_t^{(c)} n_t^{(c)}}$ se $x_i, x_j \in D_t^{(c)}$; $\frac{-1}{n_s^{(c)} n_t^{(c)}}$, caso x_i e x_j pertençam a domínios diferentes e 0 caso contrário.

Desta forma, o problema de minimização passa a ser:

$$\begin{aligned} \min_W \sum_{c=0}^C tr(W^T W) + \mu tr(W^T KL_c KW) \\ \text{s.a} \qquad \qquad \qquad W^T KHKW = I, \end{aligned} \quad (2.14)$$

De modo semelhante ao TCA, este problema pode ser reduzido a um problema de cálculo de autovetores, de modo que, é feita a escolha dos m principais componentes obtidos dos autovalores W , da seguinte equação: $(K \sum_{c=0}^C L_c K + \mu I)W = KH KW\Phi$

Após encontrar os valores de W , pode-se mapear os dados no novo conjunto de características $X^* = KW$, e recomeçar o processo iterativo, através da obtenção de um novo conjunto de rótulos, ajustados por meio desta nova aproximação.

3 Metodologia

Neste capítulo é apresentada a metodologia utilizada para a obtenção dos resultados nos diferentes experimentos de reconhecimento de padrões visuais baseados em adaptação de domínio, bem como as bases de dados utilizadas e o protocolo utilizado no decorrer dos experimentos.

Durante as explicações, a presença de uma seta para a direita indica a ocorrência de adaptação de domínio, sendo o objeto à esquerda da seta o domínio fonte, e o objeto à direita da seta o domínio alvo.

3.1 Bases de Dados

Durante este trabalho, os testes referentes à adaptação de domínio na tarefa de reconhecimento de padrões visuais foram executados em duas tarefas: Reconhecimento de Objetos e Reconhecimento Facial.

Nesta seção estão sendo apresentadas as bases de dados utilizadas em cada uma das tarefas, bem como os descritores e o processo de extração de características utilizados.

3.1.1 Reconhecimento de Objetos

Para a tarefa de reconhecimento de objetos os testes foram realizados na base Office-Caltech ([SAENKO et al., 2010](#)). Esta base de dados consiste na junção de outras quatro bases de dados para a criação de uma base específica para o estudo de técnicas de adaptação de domínio. As bases que compõe este conjunto são:

Caltech: Imagens extraídas da internet, presentes no dataset Caltech-256 ([GRIFFIN; HOLUB; PERONA, 2007](#)).

Amazon: Imagens extraídas de produtos do *website Amazon*.

Webcam: Imagens de objetos presentes em escritórios obtidas através de uma *webcam*.

DSLR: Imagens de objetos presentes em escritórios obtidas através de uma câmera DSLR.

O *dataset* é composto por imagens de 10 categorias de objetos frequentemente encontrados em escritórios, comuns entre os quatro conjuntos de dados originais, variando em termos de complexidade do fundo, posicionamento do objeto na imagem e iluminação do ambiente de captura da imagem. Na Figura [16](#) é possível observar exemplos das imagens contidas no *dataset*.

Figura 16 – Exemplos de imagens de *notebooks*, presentes nos quatro tipos de domínios da base Office-Caltech.



Fonte: Elaborada pelo autor.

Para a análise dos dados, ambos os descritores de características, SURF e DeCAF, descritos na subseção 2.2.1, foram utilizados, em ambos os casos os descritores foram disponibilizados de maneira pública pelo autor da base de dados, sem necessidade de processamento adicional em relação aos descritores. Foram extraídas 800 características SURF, com auxílio do modelo *Bag of Visual Words* para compactação das características e 4096 características do tipo DeCAF₆.

Neste problema, foram selecionados 8 problemas de adaptação de domínio dentre os domínios disponíveis, para realização dos experimentos e da análise e do efeito da adaptação de domínio nesta tarefa, sendo eles:

- Amazon → Caltech;
- Caltech → Amazon;
- DSLR → Webcam;
- Webcam → DSLR;
- Caltech → DSLR;
- Amazon → Webcam;
- DSLR → Caltech;
- Webcam → Amazon.

3.1.2 Reconhecimento Facial

Para a tarefa de reconhecimento facial, os testes foram realizados em dois tipos de bases de dados, ARFace e FRGC (*Face Recognition Grand Challenge*). Em ambos os casos, foi realizada uma etapa de recorte de faces antes da extração de características. Para a base ARFace o recorte das faces foi feito com o método sugerido por Viola e Jones (2004). Na base FRGC, o recorte de faces foi feito por meio do método MTCNN.

Após o recorte da região das faces, foi realizada a extração de características por meio do método VGGFace, extraíndo os dados da camada totalmente conectada 7 da rede, antes da extração de características, todas as imagens foram normalizadas em relação ao pixel médio fornecido pelo autor da rede VGGFace.

3.1.2.1 ARFace

A base de dados ARFace ([MARTINEZ; BENAVENTE, 1998](#)) contém imagens de 126 indivíduos em 13 diferentes cenários, ao longo de duas sessões diferentes, contando com diversos tipos de variações, como por exemplo, oclusão em determinadas regiões da face, condições de iluminação no ambiente de captura e diferenças de expressão facial, essas diferenças podem ser observadas na Figura 17.

Figura 17 – Diferentes tipos de variação presentes na base de dados ARFace, incluindo, variações de expressão facial, iluminação e oclusões na região ocular e na região da boca.



Fonte: [Zhou e Zhang \(2019\)](#).

A base foi dividida em quatro domínios para os testes posteriores que utilizam adaptação de domínio, além disso durante todos os testes, imagens com expressão neutra ou com alguma variação de expressão foram utilizadas no domínio fonte, estão apresentados os tipos de domínio estabelecidos:

- N:** Imagens faciais com expressão neutra ou alguma variação na expressão facial;
- O:** Imagens faciais com oclusão na região ocular por meio de óculos;
- C:** Imagens faciais com oclusão na região da boca por meio de cachecol;
- I:** Imagens faciais sem oclusão, com variações nas condições de iluminação.

3.1.2.2 FRGC

A base de dados FRGC (PHILLIPS et al., 2006) é uma base de dados, proposta por pesquisadores da Universidade de Notre Dame, elaborada com o objetivo de avançar e desenvolver pesquisas na área de reconhecimento facial.

A base contém imagens coloridas, coletadas ao longo de diferentes estações durante um período de três anos, oferecidas em dois tipos de contexto, ambientes controlados e ambientes não controlados.

Figura 18 – Exemplos de imagens presentes na base de dados FRGC. As duas primeiras colunas representam imagens capturadas em ambiente controlado, enquanto que as últimas três, apresentam imagens obtidas em ambiente não controlado.



Fonte: Elaborada pelo autor.

3.2 Experimentos

Todos os experimentos, tanto para o problema de reconhecimento de objetos, quanto para o problema de reconhecimento facial, foram realizados seguindo o mesmo protocolo. De modo que, os sistemas de classificação foram treinados unicamente com dados do domínio fonte e testado com dados do domínio alvo, sem que houvesse sobreposição entre os domínios.

Os diferentes problemas de adaptação de domínio foram avaliados nos quatro métodos descritos na Seção 2.3, além disso, para efeito de comparação, dois métodos de classificação foram escolhidos: 1NN e SVM.

Para a utilização da SVM, tanto como método de classificação, quanto como máquina de *kernel*, para o método TKL, o parâmetro de regularização utilizado foi $C = 10$. Além disso, para a SVM, bem como para a utilização do *kernel* inicial do TKL, a função de *kernel* de

base radial, com $\sigma = 1.0$ exposta abaixo foi escolhida. Para o TKL, parâmetro do fator de amortecimento foi estabelecido como $\zeta = 1.1$.

$$K(x, x') = \exp\left(\frac{-\|x - x'\|^2}{2\sigma^2}\right). \quad (3.1)$$

Para a avaliação do método CORAL, os dados foram normalizados por meio de suas médias e desvio-padrão, após a aplicação e ajuste dos dados do domínio alvo, o desempenho do método foi comparado aos dois classificadores, sendo avaliado por meio do 1NN, e do SVM, com os mesmos parâmetros utilizados para a classificação sem adaptação de domínio.

Já na execução dos métodos TCA e JDA, o valor de regularização escolhido foi $\mu = 1$, além disso, foram escolhidas os 20 componentes principais, reduzindo o número de características para 20. Em ambos os casos, assim como exemplificado por (LONG et al., 2013), não foram aplicadas transformações de *kernel*, utilizando a forma chamada primal, realizando as transformações diretamente no conjunto de dados e aplicando-as a um classificador 1NN. Para o JDA, foram escolhidas 3 iterações de geração de pseudo-rótulos, este valor foi escolhido empiricamente, após observar que após este ponto não havia impacto muito grande na acurácia e o tempo para realização dos experimentos diminuía consideravelmente.

A escolha dos parâmetros durante a execução dos experimentos foi baseada nos valores padrão, encontrados nas implementações dos métodos e utilizados pelos autores em seus respectivos artigos, podendo ser refinados com o objetivo de melhorar ainda mais a performance dos métodos em experimentos futuros.

Duas métricas foram utilizadas para se averiguar o desempenho da tarefa de adaptação de domínio, a acurácia, sendo o padrão em problemas de adaptação de domínio, e a métrica de separabilidade (TORRES; FALCAO; COSTA, 2004), que avalia o quanto separados estão os dados em um determinado espaço de características, por meio dos gráficos de separabilidade e da área sob a curva (AUC) destes gráficos.

A métrica de separabilidade foi aplicada em todos os métodos que alteram o espaço de características, desta forma, as características originais foram comparadas com as características no espaço de domínio invariante, por este motivo, o TKL, que retorna um *kernel* que foi utilizado por uma SVM, não teve sua separabilidade comparada.

4 Desenvolvimento

O desenvolvimento deste trabalho foi dividido em duas partes, na primeira foram realizados experimentos utilizando adaptação de domínio no problema de reconhecimento de padrões visuais. Já na segunda etapa, foi desenvolvido um aplicativo para dispositivos móveis que aborda a tarefa de reconhecimento facial por meio de técnicas de adaptação de domínio.

4.1 Experimentos

Durante a primeira etapa, foram realizados testes com as técnicas de adaptação de domínio seguindo o protocolo e os conjuntos de dados apresentados no Capítulo 3.

Para a implementação, a linguagem *Python* foi utilizada, por ser uma linguagem de fácil acesso, altamente documentada e que oferece um conjunto amplo de ferramentas de aprendizado de máquina, reconhecimento de padrões e computação científica, que são vantajosas para este trabalho.

Métodos de adaptação de domínio, assim como pode-se perceber na Seção 2.3, dependem de diversas operações baseadas em matrizes, além disso, se beneficiam de determinados métodos numéricos baseados em conceitos de álgebra linear. É importante que estas operações sejam realizadas da maneira mais rápida possível, por este motivo as bibliotecas *NumPy* (HARRIS et al., 2020) e *SciPy* (VIRTANEN et al., 2020) foram utilizadas.

A biblioteca *NumPy* é uma biblioteca que fornece acesso de alto nível a operações que envolvem matrizes e vetores, através de uma interface em linguagem *Python*, utilizando o máximo de operações paralelas por meio de técnicas de vetorização e *broadcasting*, fazendo com que as operações sejam realizadas de maneira mais rápida, e facilitando a leitura do código, de modo que se assemelha mais às expressões matemáticas a serem reproduzidas. Este pacote foi utilizado em conjunto com o pacote *ScyPy*, que aglomera um conjunto de ferramentas para computação científica, como vários métodos numéricos e técnicas de álgebra linear, que são essenciais para os métodos implementados.

Além disso, para facilitar a implementação dos métodos de classificação 1NN e SVM e da exibição dos resultados, os pacotes *Scikit-learn* (PEDREGOSA et al., 2011) e *Matplotlib* (HUNTER, 2007) foram utilizados. *Scikit-learn* é um pacote que oferece acesso a métodos de classificação supervisionados e não-supervisionados de maneira simples, facilitando a utilização de técnicas de aprendizado de máquina. O pacote também oferece ferramentas para análise de resultados e, pode ser integrado com o pacote *Matplotlib*, um pacote utilizado para a exibição de gráficos 2D.

Durante a primeira parte do processo, foi necessário realizar a extração de características

dos dados, assim como discutido no Capítulo 3, para o problema de reconhecimento de objetos, as características baseadas nos descritores SURF e DeCAF já foram fornecidas, e as características faciais foram extraídas por meio do método VGGFace, após a realização do recorte das faces.

O recorte facial pelo método de Viola e Jones (2004) foi implementado por meio da função *CascadeClassifier*, obtida na biblioteca *OpenCV* (BRADSKI, 2000), com um conjunto de contrastes disponibilizado pela própria ferramenta. Já, para o recorte baseado em MTCNN, o pacote *mtcnn*¹, foi utilizado. Em ambos os casos, uma imagem é utilizada como entrada e um conjunto de pontos representando as coordenadas x e y de onde a região da facial inicia, juntamente com o tamanho de largura e altura da região facial, são devolvidos, para que o recorte possa ser realizado.

Após o recorte de faces, a extração de características por meio da rede VGGFace foi realizada, neste caso, foi utilizada uma implementação da rede feita por meio da biblioteca *Keras*. Os pesos pré-treinados da rede original, fornecidos publicamente pelos autores do modelo, foram disponibilizados para utilização por meio de um modelo *Caffe*, que foi eventualmente convertido para pesos de modelos *Keras*, permitindo que a rede fosse acessada por um pacote chamado *keras_vggface*².

Em seguida foi necessário realizar a implementação dos métodos de adaptação de domínio, o método TKL foi implementado em linguagem *Python*, com a utilização dos pacotes *NumPy* e *SciPy*, além disso, foi necessário utilizar o pacote *qpsolvers*³, que fornece acesso de alto nível a um conjunto de sistemas para resolver problemas de minimização quadráticos. De modo a facilitar o acesso às interfaces do método, a implementação do TKL foi transformada em um pacote chamado *tklpy*⁴.

O método CORAL foi implementado utilizando a biblioteca *NumPy*, devido a sua formulação simples, a implementação do método é bem direta, exigindo somente encontrar a matriz de covariância dos dados e calcular uma matriz pseudo-inversa elevada a meio, cálculo este, que pode ser feito simplesmente elevando uma matriz a $\frac{-1}{2}$.

Os métodos TCA e JDA possuíam suas implementações de maneira pública, em um repositório didático (WANG et al., 2017), no entanto, pela disponibilização ter sido feita com objetivos didáticos, foi necessário realizar um conjunto de alterações que faziam com que o método executasse de maneira onerosa, como a diminuição de estruturas de dados repetidas, que causava uso excessivo de memória nos problemas propostos, e o ajuste de algumas operações envolvendo matrizes. Em especial uma manipulação algébrica, que melhora o desempenho do método utilizado para encontrar os autovetores da matriz de transformação.

Pan et al. (2010) comenta que é possível calcular os autovetores W , manipulando a

¹ Disponível em: <<https://github.com/ipazc/mtcnn>>. Acesso em: 28 nov. 2020.

² Disponível em: <<https://github.com/rcmalli/keras-vggface>>. Acesso em: 28 nov. 2020.

³ Disponível em: <<https://github.com/stphane-caron/qpsolvers>>. Acesso em: 28 nov. 2020.

⁴ Disponível em: <<https://github.com/jrjoaoenato/tklpy/>>. Acesso em: 28 nov. 2020.

Equação 2.13, de modo que $(I + \mu K L K)W = K H K$, evitando uma operação de inversão de matrizes. Bibliotecas como *ScyPy*, oferecem soluções otimizadas para lidar com sistemas de autovetores como este, de modo que, fornecendo as matrizes à esquerda $(I + \mu K L K)$ e à direita $(K H K)$ de W , é possível resolver o problema de maneira mais rápida.

Após o desenvolvimento dos métodos de adaptação de domínio, os resultados foram avaliados em classificadores 1NN e SVM por meio da biblioteca *Scikit-learn*, de acordo com o protocolo estipulado no Capítulo 3. Para avaliação dos resultados, a acurácia foi obtida por meio das interfaces fornecidas pelos classificadores e a separabilidade foi implementada utilizando os pacotes *Numpy* e *Matplotlib*, gerando um pacote intitulado *SePyrability*⁵.

4.2 Resultados e Discussão

Nesta seção estão apresentados os resultados obtidos nos testes realizados ao longo do projeto. De modo geral, assim como será apresentado posteriormente, as técnicas de adaptação de domínio possuem impacto positivo em ambos os problemas, demonstrando o potencial de utilização destas técnicas para problemas em que existe a presença de deslocamento de domínio.

É importante notar que, as técnicas JDA e TCA foram consideravelmente mais lentas que todas as outras técnicas ao longo de todos os testes, mesmo com as otimizações discutidas anteriormente, isso se deve ao fato de que, o cálculo dos autovetores para grandes conjuntos de dados de uma vez só, é muito lento, mesmo sendo feito de maneira otimizada. Além disso, a complexidade do método JDA, por realizar a análise por classes e por passar por um processo iterativo, é bem maior que a dos outros métodos.

A técnica CORAL não sofre com este problema, visto que não precisa encontrar autovetores, enquanto que, o método TKL lida com o problema de encontrar autovetores realizando a análise por domínios e reduzindo o problema de minimização a um problema quadrático, o que resolve os problemas de processamento encontrados nos outros métodos.

4.2.1 Reconhecimento de Objetos

Os resultados de acurácia obtidos na tarefa de Reconhecimento de Objetos, na base de dados Office-caltech, considerando o descritor **SURF**, podem ser conferidos nas Tabelas 1 e 2. É possível perceber que de modo geral, o descritor foi beneficiado pela utilização de técnicas de Adaptação de Domínio, com destaque para o método TKL, que apresentou maior acurácia na maioria dos casos.

Também é importante notar que a utilização do CORAL, raramente surtiu efeito neste problema e neste classificador, inclusive com perda de desempenho em alguns casos. Assim

⁵ Disponível em: <<https://github.com/jrjoaorenato/SePYrability>>. Acesso em: 28 nov. 2020.

como comentado anteriormente, este resultado era esperado, visto que o CORAL é um método de adaptação de domínio simples. Desta forma, pode-se interpretar que a covariância não era um fator determinante neste caso, ou pelo menos que as variações de covariância não eram lineares.

Tabela 1 – Acurárias obtidas na tarefa de classificação com e sem adaptação de domínio no problema de Reconhecimento de Objetos, utilizando o descritor SURF, sem adaptação de domínio (duas primeiras linhas) e com adaptação de domínio (cinco últimas linhas), considerando os quatro primeiros problemas de adaptação de domínio, propostos para a base de dados Office-Caltech.

| | A → C | C → A | D → W | W → D |
|------------------|---------------|---------------|---------------|---------------|
| 1-NN | 27.66% | 21.88% | 72.00% | 60.12% |
| SVM | 43.90% | 51.98% | 62.37% | 84.71% |
| CORAL 1NN | 27.66% | 23.69% | 63.38% | 60.12% |
| CORAL SVM | 43.90% | 52.71% | 47.79% | 84.71% |
| TKL | 45.94% | 55.95% | 86.10% | 84.71% |
| TCA | 40.60% | 46.65% | 88.47% | 91.71% |
| JDA | 40.60% | 46.54% | 89.83% | 92.35% |

Fonte: Elaborada pelo autor.

Tabela 2 – Acurárias obtidas na tarefa de classificação com e sem adaptação de domínio no problema de Reconhecimento de Objetos, utilizando o descritor SURF, sem adaptação de domínio (duas primeiras linhas) e com adaptação de domínio (cinco últimas linhas), considerando os quatro problemas de adaptação de domínio restantes, propostos para a base de dados Office-Caltech.

| | C → D | A → W | D → C | W → A |
|------------------|---------------|---------------|---------------|---------------|
| 1-NN | 25.47% | 29.83% | 26.26% | 22.96% |
| SVM | 48.40% | 38.30% | 29.56% | 32.15% |
| CORAL 1NN | 25.47% | 29.83% | 26.27% | 22.96% |
| CORAL SVM | 48.40% | 38.30% | 29.56% | 32.15% |
| TKL | 45.22% | 42.03% | 39.01% | 37.99% |
| TCA | 47.77% | 37.28% | 32.14% | 31.62% |
| JDA | 47.13% | 38.98% | 32.05% | 32.85% |

Fonte: Elaborada pelo autor.

Nas Tabelas 3 e 4, podemos observar os resultados de acurácia obtidos na base de dados Office-Caltech, considerando o descritor **DeCAF**. Este descritor foi elaborado seguindo um framework que visa otimizar o desempenho da arquitetura *ImageNet* na tarefa de reconhecimento de objetos e já se mostrou eficaz para a tarefa de Adaptação de Domínio sem a necessidade de técnicas adicionais ([DONAHUE et al., 2014](#)).

Mesmo assim, podemos observar que a utilização de técnicas de adaptação de domínio foi benéfica para os resultados observados, desta vez com destaque para a técnica JDA. Novamente,

o método CORAL não ofereceu tanta melhora na rede, apresentando um comportamento similar ao que foi visto no descritor SURF.

Tabela 3 – Acurárias obtidas na tarefa de classificação com e sem adaptação de domínio no problema de Reconhecimento de Objetos, utilizando o descritor DeCAF, sem adaptação de domínio (duas primeiras linhas) e com adaptação de domínio (cinco últimas linhas), considerando os quatro primeiros problemas de adaptação de domínio, propostos para a base de dados Office-Caltech.

| | A → C | C → A | D → W | W → D |
|------------------|---------------|---------------|---------------|---------------|
| 1-NN | 71.28% | 88.54% | 99.66% | 87.50% |
| SVM | 83.43% | 91.54% | 98.30% | 98.73% |
| CORAL 1NN | 83.70% | 89.97% | 99.66% | 87.50% |
| CORAL SVM | 81.74% | 91.63% | 96.61% | 98.73% |
| TKL | 82.81% | 93.63% | 97.96% | 99.36% |
| TCA | 83.79% | 89.45% | 99.32% | 99.36% |
| JDA | 83.79% | 90.81% | 99.32% | 99.36% |

Fonte: Elaborada pelo autor.

Tabela 4 – Acurárias obtidas na tarefa de classificação com e sem adaptação de domínio no problema de Reconhecimento de Objetos, utilizando o descritor DeCAF, sem adaptação de domínio (duas primeiras linhas) e com adaptação de domínio (cinco últimas linhas), considerando os quatro problemas de adaptação de domínio restantes, propostos para a base de dados Office-Caltech.

| | C → D | A → W | D → C | W → A |
|------------------|---------------|---------------|---------------|---------------|
| 1-NN | 86.24% | 74.57% | 79.16% | 77.13% |
| SVM | 87.89% | 74.91% | 66.25% | 72.65% |
| CORAL 1NN | 86.24% | 74.57% | 79.16% | 77.13% |
| CORAL SVM | 87.89% | 74.91% | 58.05% | 72.86% |
| TKL | 87.89% | 76.61% | 74.44% | 75.57% |
| TCA | 85.98% | 74.91% | 81.30% | 80.58% |
| JDA | 87.89% | 76.94% | 83.52% | 85.17% |

Fonte: Elaborada pelo autor.

A análise das métricas de separabilidade do descritor SURF apresentada na Figura 19 mostra que a utilização das características obtidas nos métodos TCA e JDA apresenta maior discriminabilidade, e suas classes estão mais separadas no espaço de características, resultado refletido nas acurárias obtidas para este descritor. Além disso, a separabilidade obtida pelo CORAL se assemelha muito, à separabilidade do descritor sem adaptação de domínio.

Por outro lado, a análise das métricas de separabilidade para o descritor DeCAF mostra que a separabilidade dos descritores sem adaptação de domínio são mais discriminativas, mesmo com as técnicas de adaptação de domínio obtendo melhora acurácia.

4.2.2 Reconhecimento Facial

Os resultados obtidos para a tarefa de reconhecimento facial podem ser observados na Tabela 5. Começando a análise pela base de dados ARFace, de imediato pode-se perceber que houve melhora na acurácia da tarefa de reconhecimento facial.

Para o problema em que o domínio alvo é composto por oclusão na região ocular, o descritor sem adaptação de domínio atingiu apenas 67.29% de acurácia, indicando uma importância grande nesta região para a discriminabilidade do sistema. No entanto, a utilização de técnicas de adaptação de domínio permitiu que o problema fosse evitado, atingindo uma performance de 89.73% de acurácia no método TKL.

O problema de oclusão na região da boca, não teve seus resultados penalizados pela oclusão de maneira tão brusca como no caso anterior, mesmo assim, a utilização de técnicas de adaptação de domínio fez com que a acurácia atingisse a marca de 99,54%.

Já para o problema com variações de iluminação, o descritor já conseguia lidar muito bem com este caso, portanto, os valores variaram pouco em relação aos resultados obtidos sem adaptação de domínio.

Na base FRGC os melhores resultados foram obtidos utilizando os métodos TCA e JDA, neste caso, o processo iterativo do método JDA não acarretou melhora na acurácia em relação ao TCA. Mesmo assim, obteve-se melhora de 12.57% na acurácia com a utilização das técnicas de adaptação de domínio. É interessante notar também, que o método CORAL ofereceu certa melhora nos resultados, mesmo que pequena, indicando que havia pelo menos alguma relação linear no conjunto de dados, da qual o método pode se beneficiar.

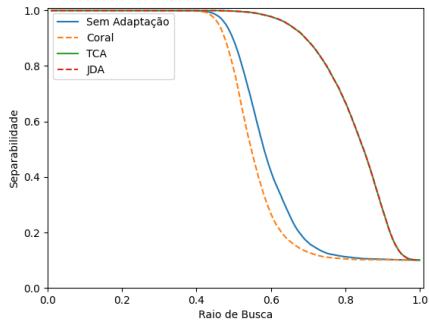
Tabela 5 – Acurárias obtidas no problema de Reconhecimento Facial, na tarefa de Identificação, com e sem a utilização de técnicas de Adaptação de Domínio.

| | FRGC | ARFace N → O | ARFace N → C | ARFace → I |
|------------------|---------------|---------------------|---------------------|-------------------|
| 1-NN | 75.33% | 67.29% | 95.89% | 99.62% |
| SVM | 76.35% | 64.81% | 94.52% | 100% |
| CORAL 1NN | 76.82% | 67.29% | 95.89% | 99.60% |
| CORAL SVM | 78.06% | 70.41% | 96.80% | 100% |
| TKL | 82.63% | 89.73% | 97.71% | 99.76% |
| TCA | 88.92% | 85.60% | 99.09% | 100% |
| JDA | 88.92% | 84.40% | 99.54% | 100% |

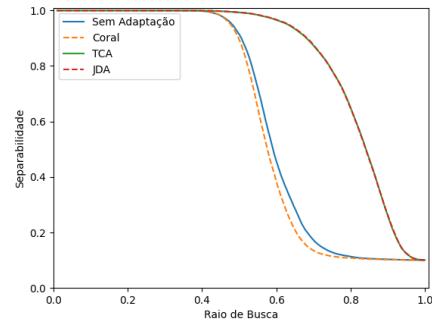
Fonte: Elaborada pelo autor.

Quando analisamos a separabilidade, por meio da Figura 21, podemos perceber que as técnicas de adaptação de domínio não só melhoraram a acurácia do método, mas também sua discriminabilidade, visto que as separabilidades dos métodos TCA e JDA foram melhores em todos os casos. É curioso notar que a separabilidade do método CORAL foi pior que a do descritor sem adaptação de domínio, mesmo que a acurácia não tenha sido tão influenciada por este aspecto.

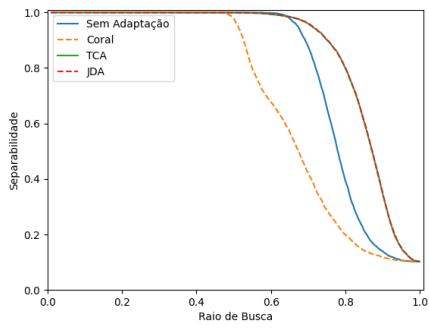
Figura 19 – Métricas de separabilidade obtidas na base de dados Office-Caltech, utilizando o descriptor SURF.



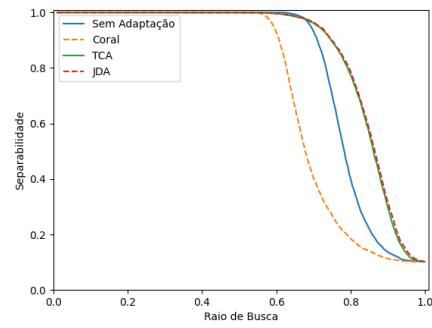
(a) Amazon → Caltech.



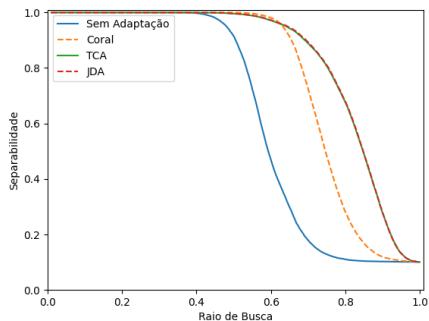
(b) Caltech → Amazon.



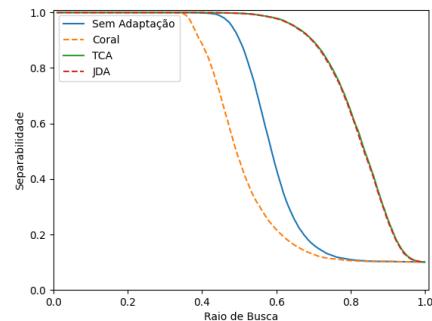
(c) DSLR → Webcam.



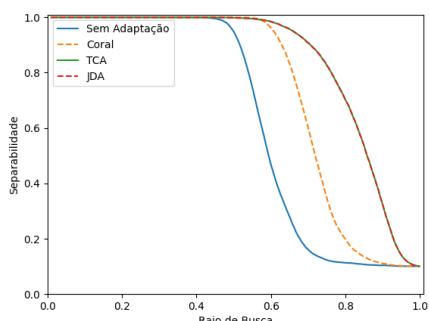
(d) Webcam → DSLR.



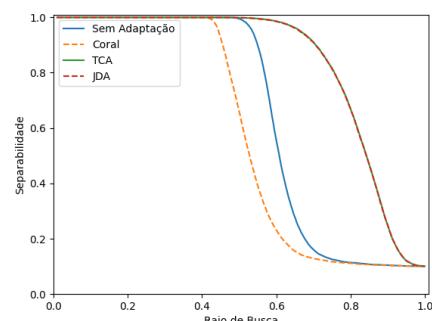
(e) Caltech → DSLR



(f) DSLR → Caltech.



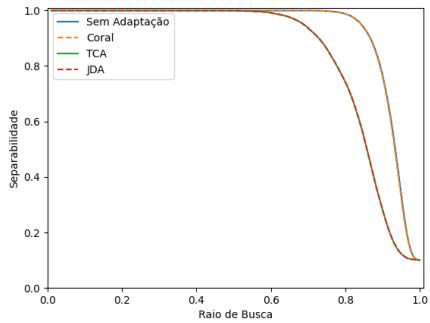
(g) Amazon → Webcam.



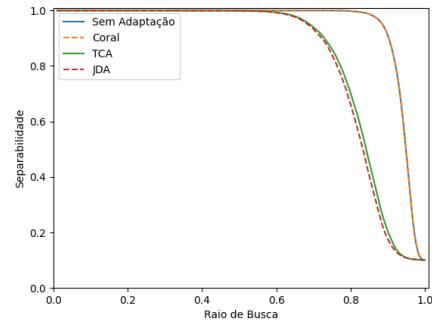
(h) Webcam → Amazon.

Fonte: Elaborada pelo autor.

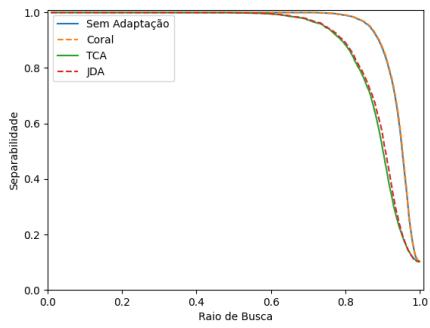
Figura 20 – Métricas de separabilidade obtidas na base de dados Office-Caltech, utilizando o descriptor DeCAF.



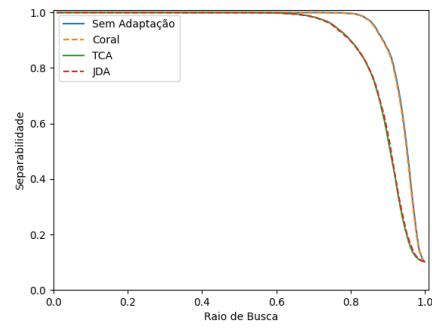
(a) Amazon → Caltech.



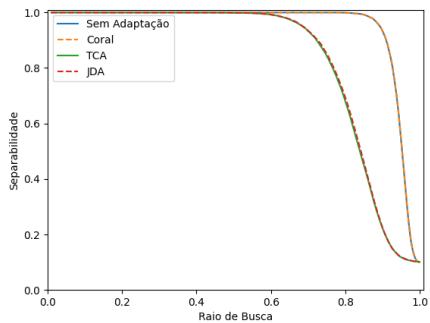
(b) Caltech → Amazon.



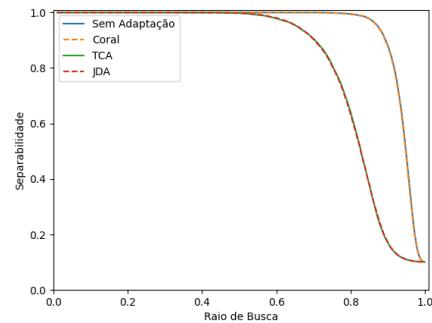
(c) DSLR → Webcam.



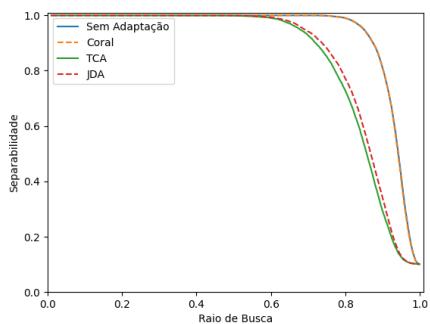
(d) Webcam → DSLR.



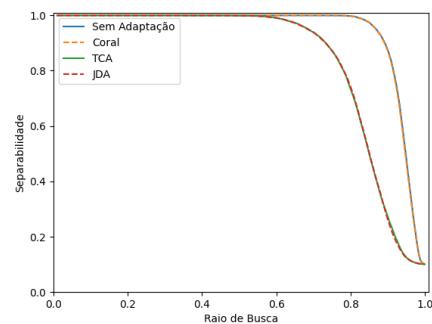
(c) Caltech → DSLR



(d) DSLR → Caltech.



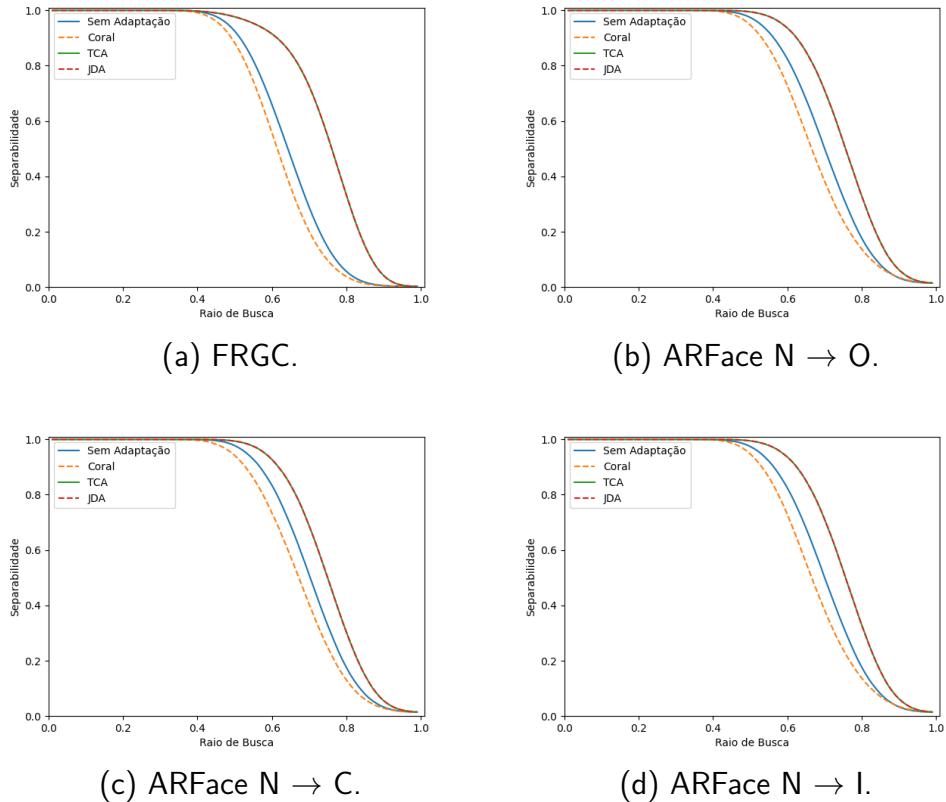
(c) Amazon → Webcam.



(d) Webcam → Amazon.

Fonte: Elaborada pelo autor.

Figura 21 – Métricas de separabilidade obtidas nas bases de dados FRGC e ARFace.



Fonte: Elaborada pelo autor.

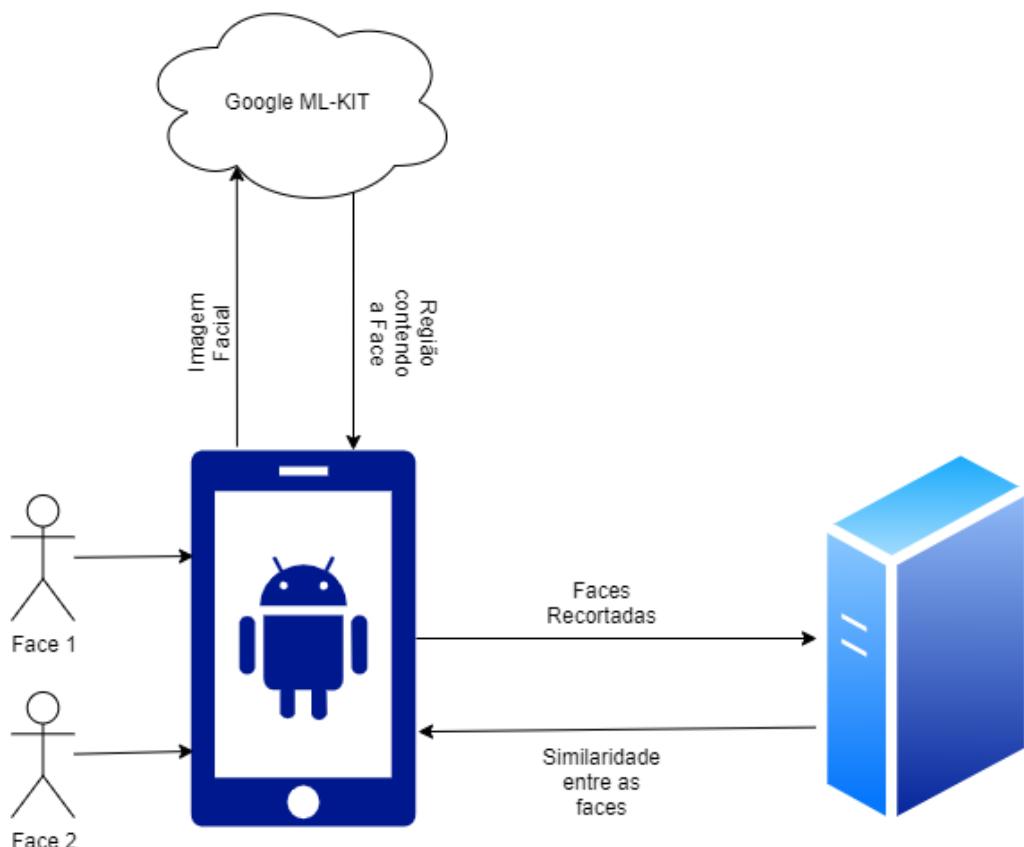
5 Aplicativo

Na segunda etapa do trabalho, foi desenvolvido um aplicativo para dispositivos móveis, que atua na tarefa de autenticação facial, comparando resultados obtidos e retornando a similaridade entre as faces e indicando se as imagens tratam do mesmo indivíduo ou não, de acordo com um valor limiar customizável.

O aplicativo foi desenvolvido utilizando a ferramenta *React Native*¹, uma ferramenta desenvolvida pelo *Facebook* para criar aplicativos para dispositivos móveis, multi-plataforma e com componentes nativos.

O aplicativo segue a arquitetura apresentada na Figura 22, como o extrator de características utilizado ao longo do projeto foi o VGGFace, foi necessária a presença de um servidor externo ao aplicativo, que realizasse esta extração de características, visto que, a arquitetura VGGFace devido ao seu tamanho, que mesmo em modo de avaliação ocupa uma grande quantia de memória, não seria adequada para execução em um dispositivo móvel.

Figura 22 – Arquitetura de funcionamento do aplicativo.



Fonte: Elaborada pelo autor.

¹ Disponível em: <<https://reactnative.dev/>>. Acesso em: 28 nov. 2020.

Além disso, foi necessária a utilização de um serviço gratuito fornecido pela *Google* por meio das APIs do *Google ML-Kit* para a realização do recorte das faces, visando dispensar o recorte manual das faces pelo usuário, que seria suscetível a erros, ou evitar sobrecarga do servidor que já está executando a rede VGGFace. O serviço utiliza uma rede convolucional baseada na arquitetura *MobileNet* para a detecção da região da face e retorna um arquivo *json* contendo as possíveis regiões faciais.

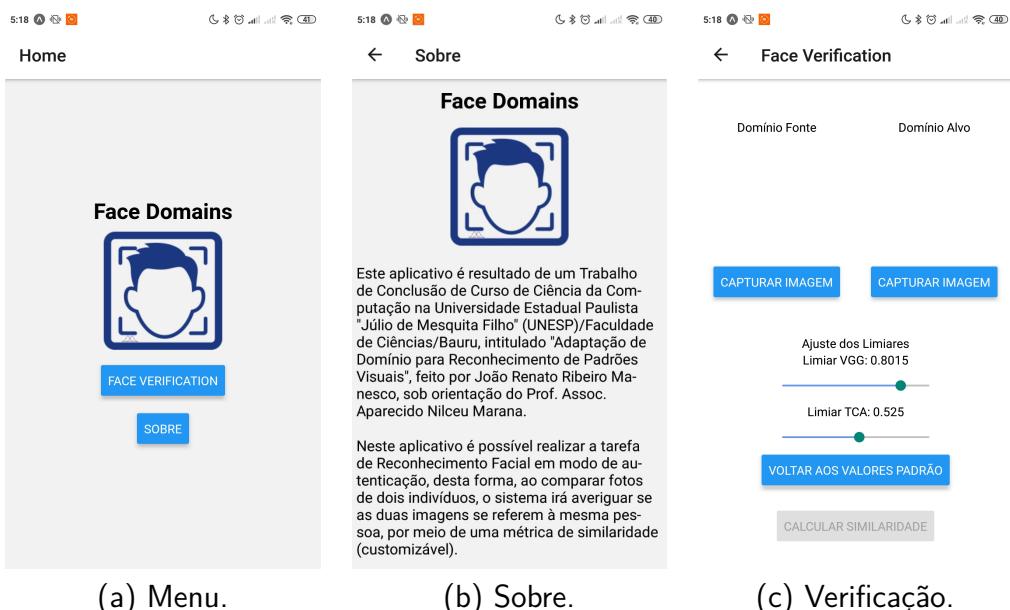
O desenvolvimento do aplicativo foi feito por meio da plataforma *Expo*, uma plataforma fornecida pelos desenvolvedores do *React Native* que facilita a integração com pacotes, simplifica os mecanismos de distribuição do aplicativo e permite observar em tempo real as alterações feitas no código.

Para que o aplicativo conseguisse lidar com mais de uma tela, a biblioteca *react-navigation* foi utilizada, por meio da interface *StackNavigation*, que permite que uma tela seja "empilhada" sobre a outra, salvando o estado anterior e permitindo voltar a este estado, removendo a tela atual da pilha.

Para a aquisição de faces, a biblioteca *expo-image-picker* foi utilizada, após a aquisição, a detecção de faces por meio do sistema *Google ML-Kit* foi feita com auxílio da biblioteca *expo-face-detector*, com a região das faces, foi possível utilizar a biblioteca *expo-image-manipulator* para realizar o recorte.

O aplicativo é formado por 3 telas, a tela do inicial, uma tela explicando a motivação do aplicativo e uma tela em que a tarefa de autenticação facial é executada. Na Figura 23 estão apresentadas as três telas do aplicativo.

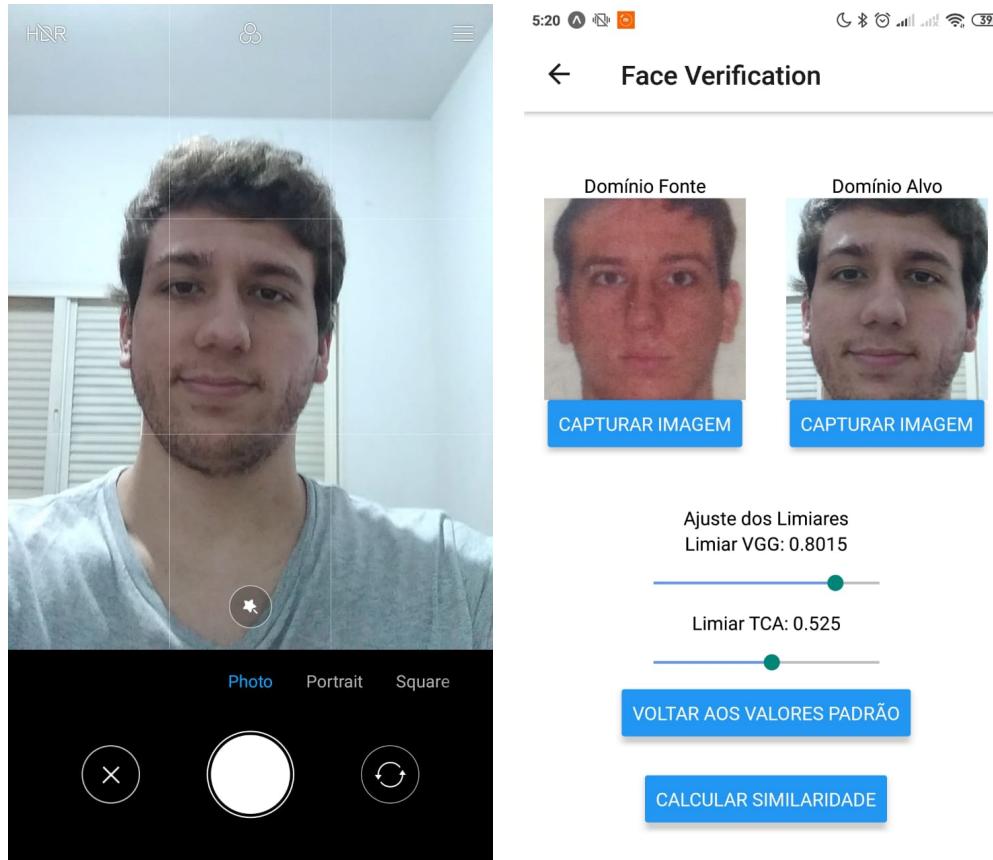
Figura 23 – Telas elaboradas para o aplicativo.



Fonte: Elaborada pelo autor.

Ao clicar no botão "Capturar Imagem", o usuário é capaz de capturar a imagem referente ao domínio desejado, assim como exemplificado na Figura 24, após a captura e o recorte das imagens em ambos os domínios é possível realizar a comparação entre as imagens.

Figura 24 – Captura da imagem.



(a) Sistema de captura da imagem.

(b) Imagens capturadas.

Fonte: Elaborada pelo autor.

Ao pressionar o botão "Calcular Similaridades", as imagens são enviadas para um servidor escrito em linguagem *Python*, que recebe os dados por meio de uma API REST, utilizando o pacote *Flask*, que facilita a criação de rotas e o interfaceamento da API REST na linguagem *Python*. As imagens são submetidas então à rede VGGFace, ao obter as características da rede, a similaridade segundo o descriptor VGGFace é comparada por meio da similaridade cosseno, através da fórmula:

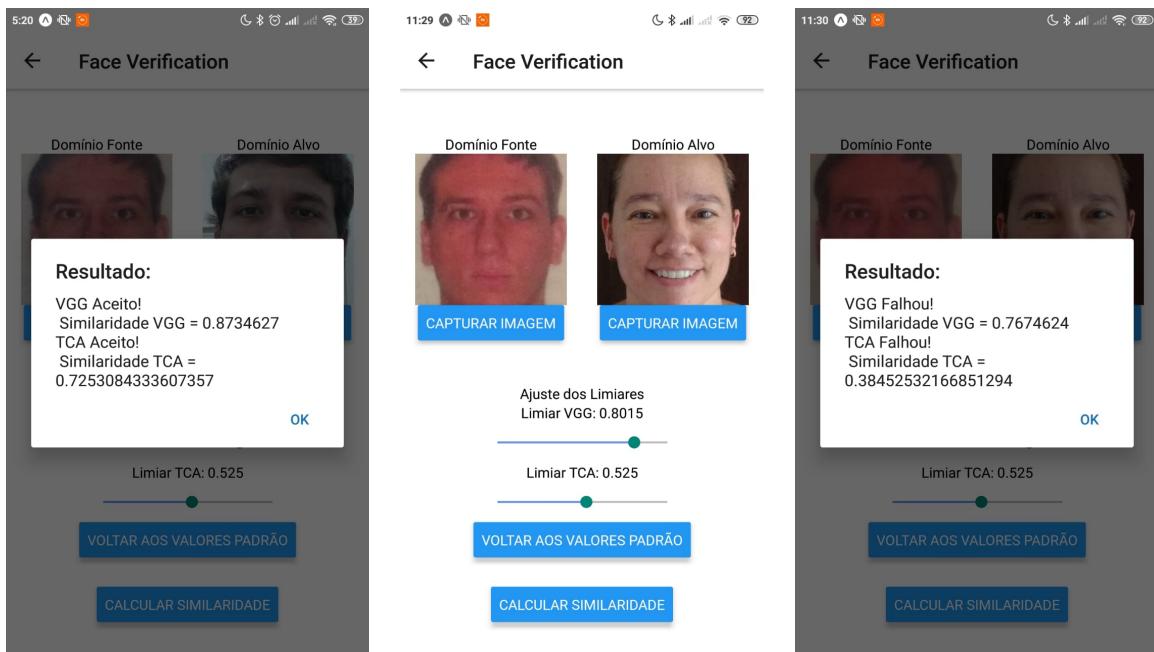
$$sim = \frac{A \cdot B}{\|A\| \|B\|}, \quad (5.1)$$

em seguida os dados são transformados para um espaço de características de domínio invariante, por meio de uma matriz de transformação *W*, obtida após a aplicação do método TCA, na base de dados FRGC, estes dados transformados para o espaço de características do método

TCA, são comparados novamente, por meio da similaridade cosseno. O servidor então retorna uma resposta contendo as duas similaridades calculadas para o aplicativo.

O aplicativo então utiliza os valores de limiar obtidos dos *sliders* da Figura 24(b), para comparar os resultados, caso a similaridade seja menor que o valor do limiar, a comparação é atribuída como falsa, do contrário, é atribuída como verdadeira, exibindo as mensagens presentes na Figura 25.

Figura 25 – Exibição dos resultados.



Fonte: Elaborada pelo autor.

Os valores padrão de limiar utilizados foram obtidos através do cálculo do *Equal Error Rate*, métrica obtida quando a taxa de erro das comparações impostoras e genuínas é a mesma, em dez mil comparações impostoras e dez mil genuínas, na base de dados FRGC, nos dois tipos de características.

6 Conclusão

Com o aumento da utilização de técnicas de reconhecimento de padrões na sociedade, exige-se cada vez mais uma melhora no desempenho dos sistemas de reconhecimento de padrões. Por conta disso, técnicas de adaptação de domínio acabam sendo necessárias.

Neste trabalho, foi possível constatar que métodos de adaptação de domínio conseguem impactar positivamente nas tarefas de reconhecimento facial e de objetos, causando aumento nas taxas de acurácia e, na maior parte dos descritores, melhor discriminabilidade, por meio de espaços mais separáveis.

Por meio da implementação de um aplicativo para dispositivos móveis, que compara similaridades obtidas entre os descritores VGGFace e TCA, é possível perceber que a utilização de técnicas de adaptação de domínio é viável não somente em ambientes experimentais, como também em sistemas reais, demonstrando a importância e praticidade de utilização deste conjunto de técnicas.

6.1 Trabalhos Futuros

Visto que a utilização de técnicas de adaptação de domínio aparenta ser vantajosa em ambientes de reconhecimento de padrões visuais, para trabalhos futuros, as técnicas de adaptação de domínio utilizadas neste trabalho, ou até mesmo um conjunto novo de técnicas, pode ser utilizado para explorar novos problemas de reconhecimento de padrões visuais, ou até mesmo, outras áreas do reconhecimento de padrões.

Além disso, com o recente desenvolvimento das técnicas de adaptação de domínio profundas no estado da arte, é possível explorar a utilização destas técnicas nas tarefas já analisadas e em tarefas futuras, permitindo também a realização de uma comparação entre técnicas baseadas em aprendizado profundo e técnicas tradicionais.

Referências

- BAY, H.; ESS, A.; TUYTELAARS, T.; Van Gool, L. Speeded-up robust features (surf). *Computer Vision and Image Understanding*, v. 110, n. 3, p. 346 – 359, 2008. ISSN 1077-3142. Similarity Matching in Computer Vision and Multimedia.
- BIEDERMAN, I. *An Invitation to Cognitive Science, Vol. 2: Visual Cognition*. Cambridge: MIT press, 1995.
- BISHOP, C. M. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2006. ISBN 0387310738.
- BRADSKI, G. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.
- CHAI, X.; WANG, Q.; ZHAO, Y.; LIU, X.; LIU, D.; BAI, O. Multi-subject subspace alignment for non-stationary eeg-based emotion recognition. *Technology and Health Care*, IOS Press, v. 26, n. S1, p. 327–335, 2018.
- CORTES, C.; VAPNIK, V. Support-vector networks. *Machine Learning*, v. 20, n. 3, p. 273–297, Sep 1995. ISSN 1573-0565.
- COVER, T.; HART, P. Nearest neighbor pattern classification. *IEEE transactions on information theory*, IEEE, v. 13, n. 1, p. 21–27, 1967.
- CRISTIANINI, N.; SHawe-Taylor, J. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge: Cambridge University Press, 2000.
- CSURKA, G. *Domain Adaptation for Visual Applications: A Comprehensive Survey*. 2017.
- DONAHUE, J.; JIA, Y.; VINYALS, O.; HOFFMAN, J.; ZHANG, N.; TZENG, E.; DARRELL, T. Decaf: A deep convolutional activation feature for generic visual recognition. In: *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*. Beijing, China: JMLR.org, 2014. (ICML'14), p. I-647–I-655.
- DUDA, R. O.; HART, P. E.; STORK, D. G. *Pattern Classification*. 2. ed. New York: Wiley-Interscience, 2012.
- FAROOQ, J. Object detection and identification using surf and bow model. In: IEEE. *2016 International Conference on Computing, Electronic and Electrical Engineering (ICE Cube)*. Quetta, Pakistan, 2016. p. 318–323.
- FERGUS, R. *Visual object category recognition*. Tese (Doutorado) — University of Oxford, 2005.
- GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. Cambridge: The MIT Press, 2016. ISBN 0262035618.
- GRIFFIN, G.; HOLUB, A.; PERONA, P. Caltech-256 object category dataset. California Institute of Technology, 2007.

- HARRIS, C. R.; MILLMAN, K. J.; WALT, S. J. van der; GOMMERS, R.; VIRTANEN, P.; COURNAPEAU, D.; WIESER, E.; TAYLOR, J.; BERG, S.; SMITH, N. J.; KERN, R.; PICUS, M.; HOYER, S.; KERKWIJK, M. H. van; BRETT, M.; HALDANE, A.; R'IO, J. F. del; WIEBE, M.; PETERSON, P.; G'eRARD-MARCHANT, P.; SHEPPARD, K.; REDDY, T.; WECKESSER, W.; ABBASI, H.; GOHLKE, C.; OLIPHANT, T. E. Array programming with NumPy. *Nature*, Springer Science and Business Media LLC, v. 585, n. 7825, p. 357–362, set. 2020.
- HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. *The Elements of Statistical Learning*. New York: Springer New York, 2009.
- HOFMANN, T.; SCHOLKOPF, B.; SMOLA, A. J. Kernel methods in machine learning. *Ann. Statist.*, The Institute of Mathematical Statistics, v. 36, n. 3, p. 1171–1220, 06 2008.
- HUNTER, J. D. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, IEEE COMPUTER SOC, v. 9, n. 3, p. 90–95, 2007.
- JAIN, A. K.; LI, S. Z. *Handbook of face recognition*. New York: Springer-Verlag London, 2011.
- JAIN, A. K.; ROSS, A. A.; NANDAKUMAR, K. *Introduction to biometrics*. New York: Springer Science & Business Media, 2011.
- KOUW, W. M.; LOOG, M. *An introduction to domain adaptation and transfer learning*. 2018.
- KOWSARI, K.; HEIDARYSAFA, M.; BROWN, D. E.; MEIMANDI, K. J.; BARNES, L. E. Rmdl: Random multimodel deep learning for classification. In: *Proceedings of the 2nd International Conference on Information System and Data Mining*. New York, NY, USA: Association for Computing Machinery, 2018. (ICISDM '18), p. 19–28. ISBN 9781450363549.
- KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, Association for Computing Machinery, New York, NY, USA, v. 60, n. 6, p. 84–90, maio 2017. ISSN 0001-0782.
- LARHMAM. *Maximum-margin hyperplane and margin for an SVM trained on two classes. Samples on margins are called support vectors*. 2018. Disponível em: <https://upload.wikimedia.org/wikipedia/commons/7/72/SVM_margin.png>. Acesso em: 08 out. 2020. CC BY-SA 4.0 <<https://creativecommons.org/licenses/by-sa/4.0/>>, via Wikimedia Commons.
- LEE, C.-Y.; BATRA, T.; BAIG, M. H.; ULBRICHT, D. Sliced wasserstein discrepancy for unsupervised domain adaptation. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Long Beach, CA, USA: IEEE, 2019. p. 10285–10295.
- LONG, M.; WANG, J.; DING, G.; SUN, J.; YU, P. S. Transfer feature learning with joint distribution adaptation. In: *Proceedings of the IEEE international conference on computer vision*. Sydney, NSW, Australia: IEEE, 2013. p. 2200–2207.
- LONG, M.; WANG, J.; SUN, J.; PHILIP, S. Y. Domain invariant transfer kernel learning. *IEEE Transactions on Knowledge and Data Engineering*, IEEE, v. 27, n. 6, p. 1519–1532, 2015.
- MARTINEZ, A.; BENAVENTE, R. *The AR Face Database*. Bellaterra, 1998. (CVC Technical Report #24).
- MITCHELL, T. M. *Machine Learning*. 1. ed. USA: McGraw-Hill, Inc., 1997. ISBN 0070428077.

- MURPHY, K. P. *Machine learning: a probabilistic perspective*. Cambridge: MIT press, 2012.
- MYSID; CYP. *Euclidean Voronoi Diagram*. 2015. Disponível em: <https://commons.wikimedia.org/wiki/File:Euclidean_Voronoi_diagram.svg>. Acesso em: 08 out. 2020. CC BY-SA 3.0 <<http://creativecommons.org/licenses/by-sa/3.0/>>, via Wikimedia Commons.
- NAKADA, M.; WANG, H.; TERZOPOULOS, D. Acfr: Active face recognition using convolutional neural networks. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. Honolulu, HI, USA: IEEE, 2017. p. 35–40.
- NETZER, Y.; WANG, T.; COATES, A.; BISSACCO, A.; WU, B.; NG, A. Y. Reading digits in natural images with unsupervised feature learning. 2011.
- OYALLON, E.; RABIN, J. An analysis of the surf method. *Image Processing On Line*, v. 5, p. 176–218, 2015.
- PAN, S. J.; TSANG, I. W.; KWOK, J. T.; YANG, Q. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, IEEE, v. 22, n. 2, p. 199–210, 2010.
- PAN, S. J.; YANG, Q. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, v. 22, n. 10, p. 1345–1359, Oct 2010. ISSN 1041-4347.
- PARKHI, O. M.; VEDALDI, A.; ZISSEMAN, A. Deep face recognition. British Machine Vision Association, 2015.
- PATEL, V. M.; GOPALAN, R.; LI, R.; CHELLAPPA, R. Visual domain adaptation: A survey of recent advances. *IEEE signal processing magazine*, IEEE, v. 32, n. 3, p. 53–69, 2015.
- 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.
- PHILLIPS, P. J.; FLYNN, P. J.; SCRUGGS, T.; BOWYER, K. W.; WOREK, W. Preliminary face recognition grand challenge results. In: *IEEE. 7th International Conference on Automatic Face and Gesture Recognition (FGR06)*. Southampton, UK, 2006. p. 15–24.
- REDMON, J.; DIVVALA, S.; GIRSHICK, R.; FARHADI, A. You only look once: Unified, real-time object detection. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV, USA: IEEE, 2016.
- SAENKO, K.; KULIS, B.; FRITZ, M.; DARRELL, T. Adapting visual category models to new domains. In: SPRINGER. *European conference on computer vision*. [S.I.], 2010. p. 213–226.
- SHAWE-TAYLOR, J.; CRISTIANINI, N. et al. *Kernel methods for pattern analysis*. Cambridge: Cambridge university press, 2004.
- SIMONYAN, K.; ZISSEMAN, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
- SUN, B.; SAENKO, K. Deep coral: Correlation alignment for deep domain adaptation. In: SPRINGER. *European conference on computer vision*. Amsterdam, The Netherlands, 2016. p. 443–450.

- TORRES, R. da S.; FALCAO, A.; COSTA, L. da F. A graph-based approach for multiscale shape analysis. *Pattern Recognition*, v. 37, n. 6, p. 1163 – 1174, 2004. ISSN 0031-3203.
- VIOLA, P.; JONES, M. Robust real-time face detection. *International Journal of Computer Vision*, v. 57, n. 2, p. 137–154, 2004.
- VIRTANEN, P.; GOMMERS, R.; OLIPHANT, T. E.; HABERLAND, M.; REDDY, T.; COURNAPEAU, D.; BUROVSKI, E.; PETERSON, P.; WECKESSER, W.; BRIGHT, J.; van der Walt, S. J.; BRETT, M.; WILSON, J.; MILLMAN, K. J.; MAYOROV, N.; NELSON, A. R. J.; JONES, E.; KERN, R.; LARSON, E.; CAREY, C. J.; POLAT, İ.; FENG, Y.; MOORE, E. W.; VanderPlas, J.; LAXALDE, D.; PERKTOLD, J.; CIMERMAN, R.; HENRIKSEN, I.; QUINTERO, E. A.; HARRIS, C. R.; ARCHIBALD, A. M.; RIBEIRO, A. H.; PEDREGOSA, F.; van Mulbregt, P.; SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, v. 17, p. 261–272, 2020.
- WANG, J. et al. *Everything about Transfer Learning and Domain Adaption*. 2017. Disponível em: <<http://transferlearning.xyz>>. Acesso em: 15 out. 2020.
- WILLIAMS, C.; SEEGER, M. Using the nyström method to speed up kernel machines. In: LEEN, T.; DIETTERICH, T.; TRESP, V. (Ed.). *Advances in Neural Information Processing Systems*. Cambridge: MIT Press, 2001. v. 13, p. 682–688.
- ZHANG, K.; ZHANG, Z.; LI, Z.; QIAO, Y. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, v. 23, n. 10, p. 1499–1503, Oct 2016. ISSN 1070-9908.
- ZHOU, J.; ZHANG, B. Collaborative representation using non-negative samples for image classification. *Sensors*, Multidisciplinary Digital Publishing Institute, v. 19, n. 11, p. 2609, 2019.