TRABALHO DE CONCLUSÃO DE CURSO

SISTEMA DE RECONHECIMENTO DE PADRÕES EM TIPOS PARA ENSINO DE TIPOGRAFIA A DEFICIENTES VISUAIS

Fernanda Garcia Vilela

Brasília, agosto de 2017

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASÍLIA
Faculdade de Tecnologia

TRABALHO DE CONCLUSÃO DE CURSO

SISTEMA DE RECONHECIMENTO DE PADRÕES EM TIPOS PARA ENSINO DE TIPOGRAFIA A DEFICIENTES VISUAIS

Fernanda Garcia Vilela

Trabalho de Conclusão de Curso submetido ao Departamento de Engenharia Elétrica como requisito parcial para obtenção do grau de Engenheira Eletricista

Banca Examinadora

Prof. Mylène Christine Queiroz de Farias, PhD., EE/UnB _____
Orientadora

Prof. Alexandre Ricardo Romariz, PhD., ENE/UnB _____
Examinador interno

Prof. Cristiano Jacques Miosso Rodrigues Mendes, PhD., FGA/UnB _____
Examinador interno

FICHA CATALOGRÁFICA

VILELA, FERNANDA GARCIA
SISTEMA DE RECONHECIMENTO DE PADRÕES EM TIPOS PARA ENSINO DE TIPOGRAFIA A DEFICIENTES VISUAIS [Distrito Federal] 2017.
xvi, 57 p., 210 x 297 mm (ENE/FT/UnB, Engenheira, Engenharia Elétrica, 2017).
Trabalho de Conclusão de Curso - Universidade de Brasília, Faculdade de Tecnologia.
Departamento de Engenharia Elétrica

1. deficiente visual	2. machine learning
3. reconhecimento de padrões	4. tipografia
I. ENE/FT/UnB	II. Título (série)

REFERÊNCIA BIBLIOGRÁFICA

VILELA, F.G. (2017). *SISTEMA DE RECONHECIMENTO DE PADRÕES EM TIPOS PARA ENSINO DE TIPOGRAFIA A DEFICIENTES VISUAIS*. Trabalho de Conclusão de Curso, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 57 p.

CESSÃO DE DIREITOS

Fernanda Garcia Vilela
Depto. de Engenharia Elétrica (ENE) - FT
Universidade de Brasília (UnB)
Campus Darcy Ribeiro
CEP 70919-970 - Brasília - DF - Brasil

Dedicatória

"O que foi é o que há de ser; e o que se fez, isso se tornará a fazer; nada há, pois, novo debaixo do sol. Há alguma coisa de que se possa dizer: Vê, isto é novo? Não! Já foi nos séculos que foram antes de nós. Já não há lembrança das coisas que precederam; e das coisas posteriores também não haverá memória entre os que hão de vir depois delas. (...) Apliquei o coração a conhecer a sabedoria e a saber o que é loucura e o que é estultícia; e vim a saber que também isto é correr atrás do vento. Porque na muita sabedoria há muito enfado; e quem aumenta ciência aumenta tristeza." Eclesiastes 1.9-11;17-18 (ARA). Dedico este trabalho àquele que está acima do sol.

Fernanda Garcia Vilela

Agradecimentos

"Sê minha vida, sê minha visão". Esse é o trecho de uma das minhas músicas favoritas. É curioso como ela diz, em poucas palavras, o que a vida de um cristão é e qual deve ser o desejo mais profundo de seu ser. Nesse aspecto, digo que minha vida é Cristo e, por isso, Ele é minha visão em tudo aquilo que faço, mesmo que por vezes eu não reconheça isso. Não é diferente com este projeto. O Deus triúno é criador e mantenedor de todo o universo, portanto, de toda ciência e saber. Os padrões, tema central neste projeto, só existem por isso. A Ele, o mais profundo agradecimento pela vida, pela visão.

Em segundo lugar, agradeço à minha família. Só eles sabem o que é me aturar de verdade, coitados. Obrigada de coração por todos os dias de apoio amoroso, companhia e desventuras em série. Obrigada, em especial, pelo apoio nestes tempos de TCC, por todas as louças lavadas em meu lugar e tal. Sem vocês, realmente, seria quase impossível viver nesse mundo cinza, mãe, pai, menina, Dedé e Bruno. Um agradecimento agregado ao menino Guilhermão, que me ouviu chorar algumas vezes por causa dos códigos. Talvez não pareça, mas amo vocês.

À minha (outra) família, meus amados irmãos e melhores amigos: Lu, Vanessa, Kim, Pedrinho, Gigi e Péricles, meu gigante agradecimento. A benção de uma amizade profunda, alegre e edificante tenho em vocês. Obrigada por todos os momentos de broncas, partilhar do coração, diversão e chatice que vocês me proporcionaram durante meu curso e durante o desenvolvimento deste trabalho. Um agradecimento em especial à Lu, menina sonhadora que idealizou o projeto, me botou nessa e me ajudou em vários pontos do desenvolvimento. Obrigada, Vanessinha e Péricles, por me entenderem durante as crises engenheirísticas e por me ajudarem com o texto e tudo o mais. Obrigada, Gigi, por corrigir meu texto mil vezes, haja paciência.

Muito obrigada, prof. Mylène. Em primeiro lugar por aceitar me orientar mesmo sendo uma ideia meio fora do normal, mesmo não sendo um projeto seu. Obrigada por todo o apoio, paciência, ensinamentos e revisão. Mais do que isso, obrigada por ter sempre me tratado muito bem, deveriam existir mais professores assim.

Obrigada a todos os meus amigos da elétrica, guerreiros! Foi um prazer ter a amizade de vocês durante os anos de muita luta. É ótimo saber que, para alguns desses, a amizade é muito mais do que circunstancial. Wawa e Pedro, vocês são meus dois melhores amigos que fiz nessa época, fico feliz porque a parceria é sólida e prazerosa. Dani, Aline, Natasha, Helô, Raquel, Iago e Rodrigo, obrigada por todo o tempo juntos, por vários dias felizes, por outros tristes, por muito cálculo, café e parceria.

Um agradecimento final por todos aqueles que, de uma forma ou de outra, estiveram presentes em minha formação! Agradeço ao Marcos Mourthè, Calil, Pri, Bruno, Kukas, Donatos, Mona, Yam, família Cruz, primos e tios!

Fernanda Garcia Vilela

RESUMO

O deficiente visual vivencia exclusão em variados aspectos, tanto social, quanto em esferas educacionais e culturais. Isto pode ser exemplificado pelo fato de o deficiente visual encontrar várias barreiras em relação à ambientes físicos que não são adaptados a eles e também ao acesso restrito a livros e textos. Além disso, percebe-se uma distanciação dessas pessoas em relação às expressões artísticas variadas, como teatro, museus e filmes. Dentro deste contexto, insere-se também a tipografia. Além de se aproximar da escrita como uma forma de comunicação, a tipografia também comunica por meio de uma linguagem visual ligada à estética. Sendo assim, um produto de tecnologia assistiva para ensino de tipografia a deficientes visuais foi proposto, em trabalho anterior, como forma de diminuir a exclusão do deficiente visual com o campo da tipografia, aproximando-o também de aspectos culturais nos quais a tipografia é aplicada, como marcas de carros, de filmes, de bandas, entre outros. Desta forma, este trabalho descreve o desenvolvimento de uma parte deste produto de tecnologia assistiva, o sistema de reconhecimento de padrões em tipos, que comporá o software auxiliar ao deficiente visual. Este sistema classifica a imagem do caractere em relação à sua tipografia por meio do reconhecimento de padrões na imagem, podendo assumir qualquer uma das nove classes, que são as nove tipografias pertencentes ao projeto. Na etapa de desenvolvimento do algoritmo, foram utilizadas técnicas de aprendizado de máquina (*Machine Learning*), no qual foi aplicado o operador Padrão Binário Local (LBP, *Local Binary Pattern* em inglês) para extração de atributos e os classificadores Máquina de Vetor de Suporte (SVM, em inglês *Support Vector Machine*) e Floresta Aleatória (*Random Forest Classifier*). O problema neste projeto é caracterizado como classificação e, como resultado, obteve-se um índice de acerto da classificação de 84,91% no melhor caso, utilizando a Floresta Aleatória. Já no caso do emprego da Máquina de Vetor de Suporte, o melhor índice de acerto da classificação foi de 74,70%.

ABSTRACT

The visual impaired experiences exclusion in various aspects, both social, as well as in educational and cultural spheres. This can be exemplified by the fact that the visual impaired finds various barriers to physical environments that are not adapted to them and also restricted access to books and texts. In addition, there is a gap between them and varied artistic expressions, such as theater, museums and films. In this context, typography is also included. Besides a form of communication as writing is, typography also communicates through a visual language linked to aesthetics. Thus, an assistive technology product for teaching typography to the visually impaired was proposed, in a previous work, as a way to reduce the exclusion of the visually impaired towards the

field of typography, also approaching cultural aspects in which the typography is applied, such as logotypes of cars, films, bands and others. In this way, this work describes the development of a part of this assistive technology product, the pattern recognition system in types, that will compose the auxiliary software for the visually impaired. This system classifies the typeface image in relation to its typography by means of the pattern recognition in the image, being able to assume any of the nine classes, which are the nine typographies included in this project. In the algorithm development stage, Machine Learning techniques were used, applying the Local Binary Pattern (LBP) operator for feature extraction and the classifiers Support Vector Machine (SVM) and Random Forest Classifier. The problem in this project is characterized as a classification problem and the best result was 84.91% as classification index using the Random Forest. In the case of Support Vector Machine, the best classification index was 45.17%.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

OCR	*Optical Character Recognition* (Reconhecimento Óptico de Caracteres)
OFR	*Optical Font Recognition* (Reconhecimento Óptico de Fonte)
SVM	*Support Vector Machine* (Máquina de Vetor de Suporte)
LBP	*Local Binary Pattern* (Padrão Binário Local)
HOG	*Histogram of Oriented Gradient*
DAG	*Directed Acyclic Graph* (Gráfico Acíclico Dirigido)
ECOC	*Error Corrected Output Coding* (Código de Correção de Erro de Saída)
OvO	*One vs. One* (Um Contra Um)
OvA	*One vs. All* (Um Contra Todos)

1 INTRODUÇÃO

1.1 DESCRIÇÃO DO PROBLEMA

1.1.1 O Deficiente Visual e as Tecnologias Assistivas

Atualmente, um grande desafio na área de tecnologia é promover a inclusão de pessoas com deficiências na sociedade, sejam elas quais forem. Tomando o caso dos deficientes visuais como foco, vários são os desafios enfrentados diariamente por eles, desde a locomoção segura até o processo de aprendizado, ou mesmo o acesso à informação. Como uma forma de amenizar essas dificuldades, garantindo igualdade e promovendo a inclusão social e a cidadania daqueles que possuem alguma deficiência, o Congresso Nacional sancionou algumas leis nesse escopo.

Um exemplo é a Lei 13.146, sancionada em 2015, que instituiu o Estatuto da Pessoa com Deficiência. Em seu texto, esta lei descreve acessibilidade em termos de possibilitar, dentre variados direitos, o acesso à informação e a garantia de autonomia para os deficientes. Além disso, a lei trata também de tecnologia assistiva, que pode ser definida, resumidamente, como produtos, serviços ou estratégias que possibilitam ao deficiente a participação em determinada atividade, proporcionando-lhe independência, qualidade de vida e inclusão social [8].

Nessa perspectiva, Bersch [9] também define tecnologia assistiva como o conjunto de recursos que amplia as habilidades daqueles que possuem alguma deficiência. Porém, Bersch [9] vai além e assinala o que seria Tecnologia Assistiva em um contexto educacional ou informacional. Em um cenário no qual, antes, a participação ativa do estudante no processo de aprendizagem seria restrita ou praticamente inexistente, a tecnologia assistiva permite que a pessoa com deficiência rompa barreiras que limitam ou impedem o seu acesso à informação ou ao registro dela. Desta forma, permitindo a inclusão do estudante invidente em projetos, por meio da disponibilização de tecnologias que permitam o seu acesso aos objetos de estudo. Vale salientar que, de acordo com Radabaugh [10], como segue na citação: “Para as pessoas sem deficiência, a tecnologia torna as coisas mais fáceis. Para as pessoas com deficiência, a tecnologia torna as coisas possíveis.”

No que tange à construção do conhecimento, segundo Levy [11], seu processo está fundamentado principalmente na oralidade e no desenvolvimento da linha de raciocínio durante a escrita do indivíduo. A elaboração do raciocínio, por sua vez, é composta e baseada nas informações acessadas e sentidas pelo sujeito em seu universo. Portanto, no caso do invidente, a informação e a conexão entre os diferentes conceitos aprendidos se dá de forma distinta dos videntes, uma vez que esse processo encontra-se sempre dependente da capacidade do indivíduo de construir significado por meio dos sentidos [12]. Sendo assim, é fácil concluir que são pessoas que necessitam de uma abordagem cuidadosa quanto ao consumo de informação.

Um exemplo claro da necessidade de adequação de material didático para uso dos deficientes visuais é o caso do campo de estudos denominado STEM, acrônimo em inglês para Ciências,

Tecnologia, Engenharia e Matemática. Os problemas enfrentados pelos deficientes visuais são vários, começando com a dificuldade presente no próprio ambiente de estudo, já que, para que adaptações em infraestrutura sejam feitas, os gastos para a universidade são consideráveis. Consequentemente, esses ajustes necessários para o bom processo de aprendizagem dos deficientes visuais são constantemente negligenciados.

Além disso, muitos professores e colegas não são familiarizados quanto ao proceder em relação aos estudantes invidentes. Porém, provavelmente, o problema com maior impacto é a falta de material em si que seja acessível, contendo informações gráficas importantes e que muitas vezes são imprescindíveis ao ensino nessa área. Uma alternativa seria, por exemplo, utilizar técnicas multissensoriais para experimentos científicos, tecnologias com *design* apropriado, como modelos 3D ao invés de figuras em livros, e também softwares que transformam imagens em conteúdo audível [13].

Graças a alguns estudos e projetos que vem sendo desenvolvidos, percebe-se que, ao combinar recursos hápticos com instruções ou acompanhamento sonoro, os deficientes visuais totais ou parciais apresentam uma grande melhora no processo de aprendizagem. No caso, por exemplo, do sistema apresentado por Plimmer [14], os estudantes são ensinados a assinarem o próprio nome por meio de instruções de áudio, enquanto sentem por meio do tato a sua própria assinatura em alto relevo e em tempo real. Desta forma, obteve-se um resultado mais significativo do que o obtido no processo tradicional de ensino, ou seja, sem o auxílio combinatório desses dois recursos.

Além desse trabalho, há também o ensino de desenho à deficientes visuais em cujo processo são empregados sons e diferentes sensações táteis, como diferenças de relevo e textura. Ballestero-Alvarez enfatiza como esses e outros componentes sensoriais, como por exemplo a temperatura, afetam o deficiente visual, fazendo-o perceber, de maneira indireta, aspectos distintos de um ambiente [15].

Vale salientar que as imagens mentais que os invidentes constroem do mundo que os rodeia são semelhantes àquelas criadas pelo indivíduo vidente, apesar de serem formadas de maneiras distintas [16]. Na ausência de um dos sentidos, a percepção do ambiente e dos objetos vem a partir dos demais sentidos. Por isso, para o deficiente visual, é essencial que variadas estimulações sensoriais estejam conectadas durante o processo de aprendizagem, de forma que possa-se extrair mais informações visuais sobre o objeto de estudo por meio da multissensorialidade [15].

Sendo assim, é de suma importância que, para o estudo de determinado objeto, a adaptação da informação visual seja feita de acordo com o sentido mais apropriado para cada caso. Esse conceito é muito bem exemplificado por Ballestero-Alvarez [15], quando o autor discute em como a torre *Eiffel* deve ser apresentada a estudantes invidentes em ambientes de estudo. Nesse caso, a opção mais interessante seria, além de disponibilizar-lhes uma maquete, descrever-lhes também a estrutura da torre e seus detalhes, para que assim a experiência de aprendizado seja mais completa, harmonizando os sentidos e proporcionando-lhes uma imagem mental mais completa.

Outro ponto que suporta essa argumentação é que, em muitos casos, a imagem mental formada por videntes não é composta somente por aspectos visuais, mas envolve simultanamente

outros sentidos. Logo, o processo de construção do conhecimento de algum objeto para os deficientes visuais, apesar de poder ser em um ritmo distinto do que o é para os videntes, não é, de forma alguma, incompleto ou inferior, podendo ser melhor desenvolvido quando as adaptações necessárias são feitas [15].

Portanto, nota-se a importância de as plataformas, produtos, serviços ou metodologias se adaptarem à realidade do deficiente visual, de forma a proporcionar o envolvimento dos invidentes com áreas do conhecimento inexploradas por eles ou que permitam a sua capacitação para exercer funções ou realizar tarefas antes impossíveis. Atualmente, ainda que de forma limitada, há esforços para mudar a realidade de exclusão dos deficientes.

Como exemplos, nesse sentido, podemos citar alguns trabalhos em desenvolvimento (ou já desenvolvidos) com propósito de aumentar acessibilidade. No Museu de Arte da Universidade Federal do Paraná (MusA), foi realizada uma exposição voltada para o público deficiente visual com a transposição de imagens em peças táteis, utilizando materiais para realçar linhas de contorno. Nesse projeto, foram desenvolvidos materiais, em braille e ampliados, de apoio didático sobre as obras de arte e sobre o museu, além de uma sessão de treinamento para os deficientes visuais. Esta sessão estava disponível antes da visita, com o objetivo de melhor ambientar os invidentes e possibilitar a formação de uma imagem mental, potencializando assim a experiência durante a exposição [17].

Uma outro exemplo é a plataforma *DOSVOX*, desenvolvido pela Universidade Federal do Rio de Janeiro [18]. O *DOSVOX* consiste em um sistema operacional que possibilita ao deficiente visual o uso do computador para realização de variadas tarefas. Esse sistema se tornou uma ferramenta muito importante para a inclusão dos invidentes, utilizada em todo o Brasil, destacando-se por ser uma ferramenta gratuita. O *DOSVOX* inclui jogos didáticos, tradução do Braille para a escrita latina e vice-versa, leitura de textos por meio de síntese de voz, e a composição e impressão de partituras, entre outras funcionalidades [19].

Outras tecnologias assistivas que se destacam são o *Jaws* e o *Virtual Vision*, sendo ambos softwares leitores de tela para uso em computador [20] [21]. Nesse mesmo escopo, existe o software *OpenBook* que escaneia documentos impressos, digitaliza-os e, por meio do Reconhecimento Óptico de Caracteres (OCR) e da fala sintetizada, torna os textos acessíveis a deficientes visuais, fornecendo instruções em áudio para a operação da ferramenta [22]. Também foi desenvolvido um dispositivo chamado Linha Braille ou *Display* Braille que, quando conectado ao computador, exibe dinamicamente a informação da tela em braille [23]. O dispositivo que possui também botões adicionais para controle dos softwares leitores de tela, unindo a informação sonora à leitura tátil [23]. Outra ferramenta interessante é o *TactileView for Tactile Graphics*, que consiste em um software para impressão de figuras e gráficos com pontos em alto relevo [24] [19].

1.1.2 O Deficiente Visual e a Tipografia

Apesar de toda essa variedade de ferramentas desenvolvidas até hoje, pouco material foi disponibilizado para promover maior aproximação do deficiente visual com a escrita latina e, de forma ainda mais acentuada, com a tipografia. O trabalho que talvez se aproxime mais dessa proposta é o sistema de Plimmer [14], que tem como objetivo ensinar de maneira mais eficaz os deficientes visuais a escrita à mão, começando com a assinatura de seu próprio nome.

Um outro exemplo que envolve essa área é o projeto de tipografia ajustável. O sistema possibilita aos deficientes visuais com baixa visão personalizar parâmetros da fonte utilizada no documento para melhorar sua legibilidade, de acordo com a necessidade de cada indivíduo [25]. Entretanto, a tipografia é envolvida nesses casos apenas como uma ferramenta para possibilitar a leitura, não sendo o foco dos projetos.

A tipografia, assim como a escrita, é utilizada para comunicação. No entanto, assim como em várias formas de arte, a tipografia comunica também por meio de formas, criando uma linguagem visual e estética. Ao associar o significado das palavras à essa linguagem visual, a interpretação é gerada. Se o deficiente visual é privado desse viés visual da informação transmitida por meio da tipografia, ele é também privado de parte da mensagem transmitida, caracterizando uma comunicação que não é bem sucedida em seu todo [26].

Sensações também podem ser geradas por meio da tipografia. Nesse caso, apesar do fim principal ser a comunicação, a questão visual e estética pode ser mais importante do que o conteúdo das palavras em si. Este aspecto que aproxima a tipografia das artes é exemplificado na Figura 1.1, na qual a tipografia é empregada tendo grande ênfase em sua forma e aspectos estésticos para compor o cartaz, em detrimento da ênfase na mensagem escrita vinculada. Dessa maneira, o deficiente visual é prejudicado quando privado dos detalhes da tipografia, já que grande parte da mensagem não é transmitido primariamente pelas palavras escritas na imagem.

Por gerar sensações distintas, uma das aplicações mais importantes da tipografia é diferenciar estilos, produtos e marcas. A tipografia desempenha papel importante no cenário atual, presente em boa parte dos artigos do cotidiano. A cultura popular se vale da tipografia em conteúdos como filmes, séries, propagandas comerciais, capas de álbuns musicais, cartazes diversos, marcas de empresas, mídias sociais, entre outros. Portanto, existe uma miríade de contextos culturais e de informação com qual o deficiente visual não tem contato, contribuindo para a sua exclusão[26].

Além disso, a tipografia também tem relevância no campo histórico, no qual influencia e é influenciada. Um exemplo é o evento da invenção da imprensa. Antes de sua criação, a comunicação e o acesso à informação eram exclusivistas uma vez que apenas poucas pessoas sabiam escrever e tinham acesso a esse tipo de material (e.g. monges). Porém, com o advento da imprensa e, consequentemente, da tipografia, houve um crescimento exponencial da comunicação escrita e da distribuição do conhecimento. Inclusive, vale ressaltar que a Reforma Protestante valeu-se da imprensa para disseminação de seus ideais [27]. Desta forma, a tipografia foi essencial para a propagação do conhecimento humano nos últimos cinco séculos.

Figura 1.1: Exemplo de tipografia aplicada como comunicação majoritariamente estética. Cartaz de Bill Graham feito para banda de *Blues* de Chicago [1].

Naturalmente, a tipografia também sofreu influências e foi moldada, tanto por limitações físicas como por correntes filosóficas adotadas pela sociedade. As limitações físicas consistem no processo de fabricação dos tipos: metal, madeira, fotocomposição ou digital, sendo que cada um desses processos possui suas características próprias.

1.2 PROPOSTA DE PROJETO

A proposta para esse projeto foi idealizada primeiramente por uma estudante de *Design* da Universidade de Brasília, Luciana Eller Cruz, com a qual esse projeto foi desenvolvido em parceria. O projeto consiste em um sistema auxiliar para ensino de tipografia a deficientes visuais [28]. O sistema, denominado Tipo Tátil, trata-se de um material tátil, com a qual o deficiente visual pode interagir e um material escrito, que traz as informações necessárias para auxiliar o ensino dessa área.

A parte tátil do sistema consiste de placas tridimensionais com as letras inseridas em alto relevo e réplicas avulsas, como ilustrado na Figura 1.2. Como prospota inicial, escolheram-se nove tipografias para compor o projeto, escolha baseada em uma classificação tipográfica específica (ver Capítulo 2). Na Figura 1.3, apresentam-se as nove tipografias listadas, exemplificadas pelo caractere “a”.

Figura 1.2: Conjunto de placas apresentando as nove tipografias [2]

Esse sistema permite que o deficiente visual seja capaz de diferenciar as características anatômicas e estilísticas de variados tipos. O sistema consiste em um alfabeto completo para cada tipografia e em um sistema pedagógico (material escrito) para que os deficientes visuais possam compreender as nuances e sutilezas da variedade de fontes tipográficas da escrita latina, entendendo seus significados formais e históricos [28] [2]. Além disso, o projeto foi expandido para abarcar um exemplo de escrita cursiva, também apresentando placas com letras em alto relevo, bem como caracteres especiais de acentuação e operações matemáticas.

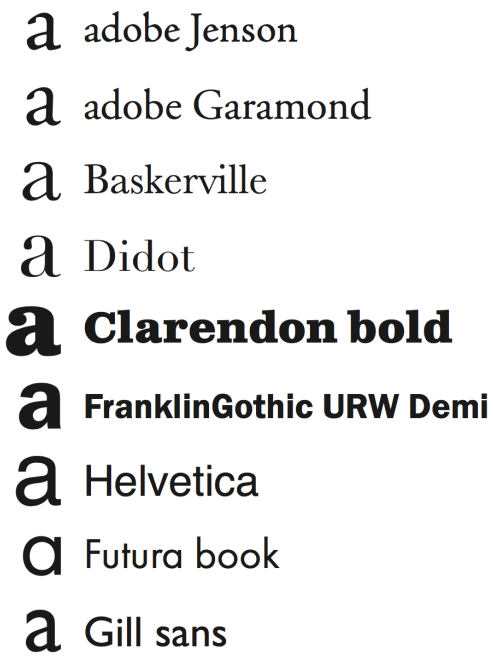

Figura 1.3: Tipografias que compõem o projeto.

De modo a tornar o projeto ainda mais completo, com o intuito de proporcionar ao deficiente a autonomia para estudar, de acordo com o conceito da tecnologia assistiva, e sabendo que maior eficácia é alcançada quando proporcionados estímulos multissensoriais, propõe-se a implementação de um sistema computacional auxiliar e interativo para guiar o deficiente visual durante o uso do produto de tecnologia assistiva [15].

O sistema funcionaria com o usuário inserindo, em uma área pré-determinada, a peça sobre a qual deseja maiores informações. Então, a partir de captura de imagem (propõe-se uma *webcam*) ocorreria o reconhecimento da tipografia e do caractere da peça e, por meio da síntese de voz, seriam providas ao usuário informações detalhadas sobre os conceitos acerca daquela tipografia, a saber, aplicações, especificações estilísticas e contexto histórico, conduzindo-o durante o processo de interação com a peça.

O escopo deste documento é o desenvolvimento e resultado da parte central do sistema: o algoritmo de reconhecimento de padrões que é utilizado para a classificação da tipografia da peça (tipo) contida em uma imagem de entrada (Figura 1.2). A imagem pode ser classificada em uma de nove classes, ou seja, em uma das nove tipografias que compõem o projeto.

Tendo em vista que o deficiente visual possui pouco contato com a área, abre-se a oportunidade de um novo tipo de leitura para o deficiente visual, um novo horizonte de conhecimento e interação dos invidentes com os caracteres latinos, usados em mais de um terço do mundo [29]. Ainda, a mitigação da exclusão cultural do deficiente visual, que é agravada pelo distanciamento desse grupo em relação ao elemento estético da tipografia.

Os deficientes visuais se encontravam em um "gueto cultural" antes do desenvolvimento de ferramentas como o *DOSVOX* que possibilitassem a tradução dos textos escritos em braille para os caracteres latinos. Suas produções ficavam restritas já que raros videntes sabem ler ou escrever em braille e o acesso à informação ficava comprometido [18]. Nesse sentido, o projeto aqui proposto contribui também para a eliminação deste "gueto cultural".

Como já enfatizado, Neto [12] fez um trabalho voltado para responder a questão sobre qual é a influência que o uso das tecnologias assistivas de informação e comunicação tem sobre a experiência do deficiente visual com a escrita e com a leitura por meio do braille. Por outro lado, a proposta aqui apresentada estende esse objetivo ao tentar inserir o deficiente visual em um contexto que diz respeito, não somente aos caracteres utilizados pela língua nativa do país onde vive, mas também ensinar aspectos a respeito à anatomia de cada letra, permitindo diferenciar os caracteres entre si e os seus estilos, aproximando-o da experiência que um vidente tem ao ler qualquer mídia.

O sistema proposto possui não só um caráter de aproximação do invidente com a escrita e com os diversos estilos tipográficos, mas também proporciona-lhe a opção de se aprofundar nos estudos da tipografia, incluindo conceitos, aplicações e contextos históricos. Sendo assim, esse projeto apresenta um material inédito, por ser uma tecnologia assistiva que proporciona ao invidente o contato com uma informação antes restrita a ele, apesar de inicialmente ainda ser uma tecnologia com opções limitadas.

Além do mais, como o envolvimento dos deficientes visuais com o mundo da informática está crescendo, há necessidade de cursos formativos para que o invidente possa conhecer melhor as possibilidades de leitura e escrita utilizando as tecnologias assistivas dessa natureza [12]. Sendo assim, um maior contato do deficiente com a tipografia é um fator predominante nesse contexto e que deve ser ensinado ao deficiente visual, sendo uma área ainda pouco explorada.

Focando, então, no desenvolvimento do sistema, para o reconhecimento de padrões, são empregadas técnicas de aprendizado de máquina (*Machine Learning*). Geralmente essa é a melhor abordagem para se tratar problemas de reconhecimento de padrões, apesar de também haver a possibilidade de serem aplicadas outras técnicas, como as heurísticas [30]. O aprendizado de máquina é uma técnica pertencente ao campo da Inteligência Artificial e que tem como objetivo projetar sistemas capazes de aprender de forma automática, sem terem sido explicitamente programados para tal [31] [32]. O processo de aprendizado do sistema ocorre por meio de experiências e de soluções bem-sucedidas de casos anteriores [33]. As técnicas de aprendizado de máquina tem sido aplicadas em diversos problemas com o objetivo de extrair conceitos a partir de um banco de dados (reconhendo padrões). Sendo assim, a constituição do banco de dados (suficientemente grande) é uma parte importante do projeto e compõe o conjunto de treinamento para o algoritmo de aprendizado de máquina.

O banco de dados, neste projeto, é formado por imagens (fotografias) contendo os diversos tipos. As categorias das imagens são previamente conhecidas e rotuladas, sendo os rótulos de cada imagem o nome de cada tipografia. Desta forma, o processo de aprendizado de máquina é denominado um problema de aprendizado supervisionado (*supervised learning*).

Em muitas aplicações, devido à grande variação dos dados de entrada, o algoritmo compreende apenas uma pequena porção de todo o conjunto que deseja-se classificar. Sendo assim, uma prática comum é passar os dados (neste caso, as imagens) por um estágio de pré-processamento, de forma a uniformizar as imagens de entrada. Um exemplo do processo adotado em um algoritmo de reconhecimento óptico de caracteres (OCR, *Optical Character Recognition*, em inglês) é apresentado na Figura 1.4. Está é uma abordagem comum para a solução desse tipo de problema, sendo uma abordagem semelhante utilizada neste trabalho [30] [34].

Os classificadores empregados neste projeto são a Máquina de Vetor de Suporte (SVM, em inglês *Support Vector Machine*) e a Floresta Aleatória (*Random Forest Classifier*). Máquina de Vetor de Suporte é um modelo classificador paramétrico, baseado na Teoria de Aprendizado Estatístico, podendo ser usada em casos de classificação ou regressão linear [4]. Já a Floresta Aleatória é uma extensão do modelo de Árvores de Regressão e de Classificação e sua predição baseia-se em dividir recursivamente o conjunto de dados, com base nos valores de preditores [35].

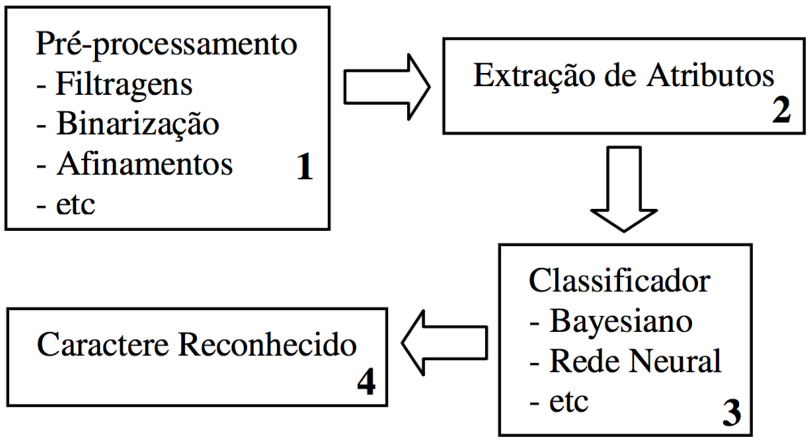

Figura 1.4: Etapas do Processo de Reconhecimento de Caracteres [3]

Vários dos softwares de tecnologia assistiva utilizam OCR em sua composição para leitura de tela ou de arquivos. Entre os exemplos, podem-se citar o *OpenBook*, o *Jaws* e o *Virtual Vision* [22] [20] [21]. No sistema de auxílio ao ensino da escrita à mão para deficientes visuais proposto por Plimmer [14], também utiliza-se OCR e uma abordagem semelhante à de Miranda [34]. Desta forma, percebe-se que esse campo de reconhecimento de caracteres favorece o desenvolvimento de várias tecnologias assistivas.

Apesar de o reconhecimento de padrões ter raízes históricas na ciência, a sua abordagem com aprendizado de máquina encontra-se em um momento ímpar de crescimento. Graças ao veloz crescimento e popularização dos dispositivos com câmera, como smartphones e tablets, e também da internet e redes sociais, o volume de imagens e vídeos compartilhados na rede é imenso. Além disso, a capacidade de processamento dos computadores também apresenta um crescimento acelerado. Dessa forma, o aprendizado de máquina enfrenta um ótimo momento para seu desenvolvimento, já que está acessível um grande banco de dados de imagens para compor o conjunto de treinamento e o hardware necessário para o processamento também [36]. Sendo assim, várias são as aplicações nessa área, desde utilidades no campo das ciências sociais até o campo da criminalística [37] [38]. Para o reconhecimento de padrões na área de Visão Comuputacional, o aprendizado de máquina é muito importante, sendo aplicado em várias situações como, por exemplo, na localização de prédios, no reconhecimento facial e na edição inteligente de fotos [39].

Por outro lado, o reconhecimento de tipos (OFR, *Optical Font Recognition*, em inglês) con-

tinua como um campo que não é muito explorado pela comunidade de pesquisadores [40]. Geralmente, quando o problema é atacado, utiliza-se o reconhecimento de tipos apenas como forma de melhoria para o reconhecimento de caracteres, especialmente quando um documento possui variadas tipografias, ou ainda para uma possível classificação de documentos, que podem ser diferenciados de acordo com a tipografia ali empregada [41] [42]. Outra situação na qual o reconhecimento de fontes é utilizado é no reconhecimento de caracteres não-latinos, como chineses, arábicos ou farsi [43] [44] [45].

Nesse contexto, o trabalho de Zramdini [40] tem como diferencial um sistema que considera todas as variedades anatômicas e intrafamiliares das fontes e por também propor uma combinação entre o reconhecimento de fontes e de caracteres, o que faz parte do objetivo do sistema final proposto para o projeto apresentado neste documento. No entanto, Zramdini [40] apresenta uma abordagem distinta da aplicada nesse projeto para o reconhecimento das tipografias. Em seu trabalho, o autor utiliza as características dos caracteres, baseadas em propriedades locais das letras individuais, tais como a anatomia de cada letra e as peculiaridades de determinadas tipografias. Por exemplo, uma das características utilizadas é o estilo de serifa, para estimá-lo, computa-se a borda de cada caractere e analisa-se o comprimento da borda, para determinar o estilo da serifa. Uma das restrições deste trabalho [40] é que, para que possa ocorrer a classificação de tipografia baseada em comparações estruturais entre os tipos, deve-se ter um conhecimento prévio em relação à qual classe o caractere avaliado pertence. Sendo asim, o autor desenvolveu um sistema classificador distinto para cada caractere. Este sistema primeiramente aplica um algoritmo de OCR para identificar o caractere e, posteriormente, o modelo classificador para classificar a tipografia (OFR).

Já no caso do projeto apresentado neste trabalho, os caracteres foram tratados simplesmente como imagens, não sendo um requisito o seu reconhecimento prévio. Os seus atributos foram extraídos, também de forma local, por meio do Padrão Binário Local (LBP, *Local Binary Pattern* em inglês), um operador proposto em 1994 e utilizado em visão computacional, especialmente em classificação de texturas e reconhecimento de faces [46] [5].

1.3 ESTRUTURA DO TRABALHO

Este documento divide-se de forma a apresentar uma estrutura que facilite o entendimento do leitor em relação ao desenvolvimento do sistema de reconhecimento de padrões em tipos para classificação das tipografias presentes no projeto explicitado na seção anterior. No Capítulo 2, apresentam-se conceitos básicos de tipografia, relevantes para o projeto. No Capítulo 3 são explicados os fundamentos teóricos dos modelos de extração de atributos e de classificação, bem como a estrutura do algoritmo de Aprendizado de Máquina.

Já nos Capítulos seguintes, a saber, 4 e 5, é detalhado o desenvolvimento do trabalho especificamente, com todos os algoritmos implementados na metodologia e os resultados alcançados,

bem como sua análise. No Capítulo 6, apresentam-se as conclusões e discutem-se quais os objetivos do projeto já foram alcançados e quais serão os passos futuros.

2 SOBRE TIPOGRAFIA

O ensino de tipografia a deficientes visuais requer a transformação das formas e dos conceitos visuais em modelos táteis. Neste capítulo, definem-se quais conceitos serão trabalhados, identificando as partes estruturais das letras, que apresentam distinções de acordo com a tipografia a qual pertencem. Neste capítulo são apresentados esses conceitos e detalhados os fundamentos deste projeto.

2.1 CONCEITOS GERAIS

A tipografia é a comunicação escrita por meio de tipos [47].Surgiu na Alemanha na década de 1450, quando aliaram-se os tipos móveis de metal à prensa, resultando no primeiro livro impresso [48]. A diferença entre a letra escrita e a tipográfica são os método utilizados para gerar as letras [49]. Por exemplo, existem fontes de letra cursiva, que apesar da aparência de caligrafia, na realidade, são tipografia, uma vez que são geradas por uma definição de fonte de computador. Sendo assim, para considerar-se tipografia, o processo deve ser regulado por máquina. Na era digital, os tipos móveis foram adaptados para o que chamamos de fontes, apesar de o termo ser usado erroneamente. Fonte diz respeito ao software e já tipo, à configuração visual [49].

Dentro da tipografia, existem alguns parâmetros que determinam as configurações de um tipo. As linhas-guia, por exemplo, são linhas que determinam limites horizontais para as letras em dado tipo. Essas linhas-guia são classificadas como linha das ascendentes, linha mediana, linha de base e linha das ascendentes, conforme ilustrado na Figura 2.1. A linha de base serve para o alinhamento de todas as letras. Já a linha mediana indica a altura-x, ou seja, a altura das letras minúsculas. A linha das ascendentes delimita as letras que se projetam acima da linha de base e a linha das descendentes, das que se projetam abaixo da linha de base [50].

Figura 2.1: Conceitos básicos de tipografia: linhas-guia, eixo e altura-x das letras.

A relação entre a altura-x e altura das capitulares é o que vai fazer com que um tipo pareça

maior ou menor em uma mesma quantidade de pontos. Quanto menor for a diferença entre essas duas alturas, maior será a aparência do tipo e maior a sua legibilidade, principalmente em tamanhos pequenos. A unidade de medida usualmente utilizada para a medição de tipos é o ponto, que é equivalente a 0,35 mm. A contagem dos pontos em determinado tipo é feita desde a linha das ascendentes até a linha das descendentes [49].

Outros parâmetros moldam a aparência de um tipo, como por exemplo o contraste, a inclinação, o eixo e a largura. O contraste é definido como a diferença entre traços finos e grossos. Já o eixo é o ângulo no qual se constroem as letras e, por último, a largura é a extensão horizontal das letras.

Outro conceito importante no campo da tipografia é chamada anatomia da letra, que corresponde a termos técnicos que possuem o objetivo de especificar determinada parte de uma letra, como está ilustrado na Figura 2.2.

Figura 2.2: Partes principais da anatomia das letras.

2.2 CLASSIFICAÇÃO TIPOGRÁFICA

As classificações tipográficas surgiram com a intenção de criar uniformidade na definição e descrição dos tipos. Ainda que não exista uma versão oficial na área da tipografia, a classificação criada por Maximilien Vox, adaptada posteriormente pela Associação Tipográfica Internacional (ATypi), é uma das classificações mais reconhecidas e utilizadas atualmente. Esta classificação, conhecida como classificação Vox ATypi, une características históricas e estéticas e é dividida em grupos e subgrupos, conforme ilustrado na Figura 2.3 [51].

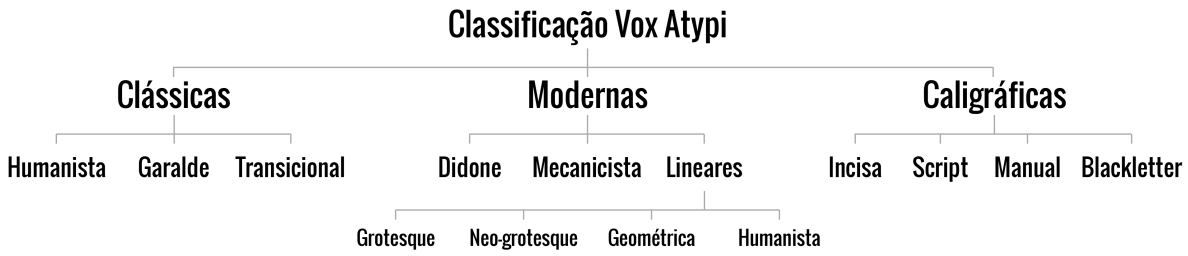

Figura 2.3: Sistema de classificação tipográfica Vox ATpyi [2]

O sistema divide-se em três grupos principais de tipografia: clássicas, modernas e caligráficas. Cada grupo é também dividido em alguns subgrupos, que são listados a seguir [2].

1. Caligráficas: incisa, *script*, manual e *blackletter*;
2. Clássicas: humanista, *garalde* e transicionais;
3. Modernas: *didone* e mecanicista;
4. Lineares: *grotesque*, *neo-grotesques*, geométrica e humanista.

Neste projeto, essa classificação é utilizada como modelo guia para a escolha das tipografias. A partir das categorias presentes na classificação citada, é possível definir uma estrutura que permite apresentar a tipografia por um panorâma histórico. Desta forma, é possível descrever como a tipografia evoluiu de acordo com as técnicas empregadas e com as correntes filosóficas contemporâneas, sendo esta uma abordagem interessante para o projeto.

Sendo assim, foi escolhida uma tipografia representante de cada categoria, excluindo-se a categoria caligráfica, por não pertencer ao escopo principal do projeto neste primeiro momento, que tem como intuito uma representação caligráfica e não tipográfica. As tipografias adotadas foram escolhidas devido à sua relevância histórica, estão listadas a seguir e são ilustradas na Figura 2.4.

1. Humanista: Adobe Jenson;
2. *Garalde*: Adobe Garamond;
3. Transicional: Baskerville;
4. Didone: Didot;
5. Mecanicista: Clarendon;
6. Grotesque: Franklin Gothic URW;
7. Neo-grotesque: Helvetica;
8. Geométrica: Futura;
9. Humanista: Gill Sans.

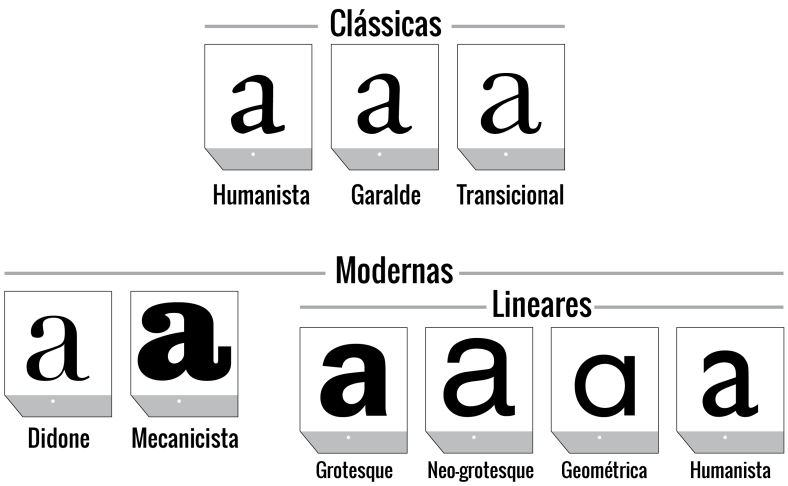

Figura 2.4: Categorias do sistema de classificação tipográfica Vox ATpyi que foram utilizadas no projeto [2]

2.3 APLICAÇÕES

Como já retratado no Capítulo 1, um dos objetivos do projeto é mitigar a exclusão que o deficiente visual sofre em relação à várias áreas do conhecimento humano, dentre elas, a cultura e as artes. Sendo assim, pretende-se apresentar a tipografia por meio de uma tecnologia assistiva, que fornece explicações sobre as características e os conceitos, exemplificando por meio de aplicações reais das tipografias do projeto. Alguns exemplos práticos do uso das tipografias contidas no projeto são apresentados na Figura 2.5. Observe que, dentre os exemplos, estão marcas de carros, equipamentos eletrônicos e roupas.

Figura 2.5: Exemplos de aplicação das tipografias presentes no projeto: a) Gill Sans, b) Helvetica, c) Didot, d) Garamond, e) Baskerville, f) Jenson, g) Clarendon, h) Franklin Gothic, i) Futura.

3 RECONHECIMENTO DE PADRÕES E APRENDIZADO DE MÁQUINA

Este capítulo é destinado à apresentação da fundamentação teórica dos conceitos gerais de reconhecimento de padrões e dos algoritmos de aprendizado de máquina utilizados neste projeto. Também são detalhados os processos utilizados no pré-processamento dos dados de treinamento (imagens) e na extração de atributos que estão relacionados ao desenvolvimento do sistema.

3.1 CONCEITOS GERAIS

O príncipio sobre o qual o Aprendizado de Máquina opera é a indução, na qual, a partir de um conjunto de casos (exemplos) que tipificam o problema, são feitas generalizações e, então, decisões são tomadas baseadas nessa regra construída. Para tal, necessita-se de um conjunto de dados de treinamento, que desempenham o papel dos exemplos para indução e um modelo de Aprendizado de Máquina, que irá ser ajustado conforme padrões apresentados nos dados de treinamento.

Como já descrito brevemente no Capítulo 1, o aprendizado indutivo da máquina pode ocorrer como aprendizado supervisionado (*supervised learning*) ou aprendizado não-supervisionado (*unsupervised learning*). O último caracteriza-se por utilizar um conjunto de dados cujos rótulos de classes não são utilizados no treinamento da máquina, sendo os domínios divididos naturalmente. Essa divisão é feita a partir da análise de padrões nos dados (imagens) como regiões com alta densidade relativa de pontos no espaço [52].

Já o aprendizado supervisionado, como no caso deste projeto, se dá com um conjunto de dados de treinamento que possuem classes já definidas e, portanto, tem-se uma estrutura na seguinte forma: entrada e saída desejada. As classes são apresentadas como rótulos de cada amostra do conjunto de dados e todo esse processo de aprendizado supervisionado é mostrado na Figura 3.1, na qual o conjunto de exemplos rotulados apresentam a forma $(\boldsymbol{x_i}, y_i)$, em que $\boldsymbol{x_i}$ é o vetor de uma amostra e y_i é seu rótulo. O classificador obtido por meio do algoritmo de aprendizado de máquina é denotado como f. Neste exemplo, tem-se um conjunto de n dados, onde cada dado $\boldsymbol{x_i}$ possui m atributos.

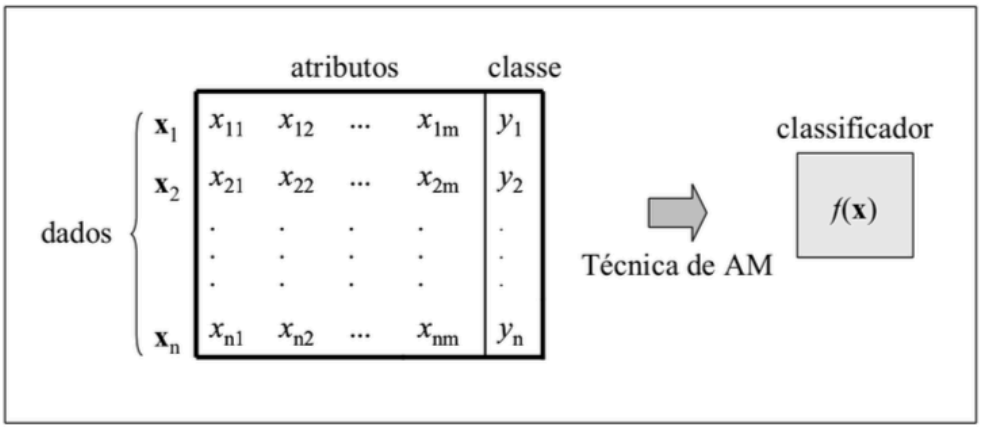

Figura 3.1: Processo de treinamento de modelo classificador com aprendizado supervisionado [4].

Os exemplos do conjunto de treinamento são representados como atributos em um vetor, sendo que cada atributo expressa uma característica do dado, a qual é utilizada no treinamento do modelo de aprendizado de máquina. Os tipos de atributos são o nominal e o contínuo. O primeiro tipo (nominal) gera um problema de classificação, no qual o objetivo é atribuir a cada vetor de entrada uma das categorias finitas (valores discretos). Este é o tipo de problema apresentado neste projeto. O tipo contínuo de atributo corresponde a um caso de regressão, em que à saída desejada é atribuída uma ou mais variáveis contínuas. Um exemplo é o problema de predição do preço de uma casa, que pode utilizar atributos como o tamanho e a localização da casa [30].

O classificador também utiliza uma função para descrever os dados. Sendo assim, é utilizada uma parte do conjunto de dados, denominada conjunto de teste, para mensurar o grau de desempenho do modelo de predição. Neste projeto, para avaliação do desempenho e da capacidade de generalização do modelo utiliza-se a técnica de Validação Cruzada (*Cross Validation*). Nesta técnica, divide-se o conjunto total de dados em vários subconjuntos, sendo alguns utilizados para estimação dos parâmetros do modelo (etapa de treinamento) e outros, para a validação [53]. Os três métodos mais populares de Validação Cruzada são: K-Pastas (*K-Fold*), Substituição da Amostra (*Holdout*) e Deixe-Um-De-Fora (*Leave-One-Out*), sendo o primeiro escolhido para a implementação do sistema aqui apresentado [54].

No método K-Pastas, o conjunto de dados é dividido aleatoriamente em k subconjuntos, sendo um subconjunto destinado para teste e $k-1$ subconjuntos, para treinamento. O processo de treinamento do modelo classificador e de avaliação de seu desempenho é realizado k vezes, sendo que, a cada vez que o processo é repetido, faz-se uma nova divisão do conjunto total em subconjuntos. Então, o desempenho final do classificador é obtido calculando-se a média de k avaliações [55].

3.2 EXTRAÇÃO DE ATRIBUTOS

Para o treinamento do algoritmo classificador são necessários atributos dos dados do conjunto de treinamento. A extração de atributos está relacionada com a redução da dimensionalidade de dados. Além de serem parâmetros descritivos do dado ao qual pertence, os atributos evitam que a quantidade de dados de entrada para o treinamento do modelo seja excessivamente grande, o que aumentaria o esforço computacional.

Desta forma, o problema de super-ajustamento (*overfitting*) às amostras no treinamento pode ser evitado com a correta seleção de atributos. O super-ajustamento pode ser causado por uma grande quantidade de variáveis no modelo, sendo definido como uma demasiada especialização nos dados de treinamento e, consequentemente, uma generalização ruim que gera uma baixa taxa de acerto para dados de entrada diferentes dos de treinamento [4].

Também pode ocorrer um caso de sub-ajustamento (*underfitting*), o que produz uma taxa de acerto que pode ser também considerada baixa para esta aplicação. O sub-ajustamento, por sua vez, é quando o modelo treinado não descreve bem o conjunto de dados de treinamento, apresentando baixa e insuficiente especialização em relação a esses dados. O sub-ajustamento em geral se deve à escolha de atributos pouco representativos para o tipo de análise pretendida.

Conclui-se, então, que a etapa de extração de atributos é importante para o bom desempenho do classificador. Neste projeto, a extração de atributos das imagens de entrada foi feita utilizando o operador Padrão Binário Local (LBP, *Local Binary Pattern* em inglês). Os vetores de atributos são formados concatenando-se os histogramas das imagens resultados da aplicação do LBP, o qual é descrito a seguir.

3.2.1 Padrão Binário Local (LBP)

O operador LBP se popularizou devido à sua simplicidade computacional, que possui pouca sensibilidade à variação de iluminação e invariância à rotação. Esse operador, quando aplicado em uma imagem, substitui cada pixel (central) por um valor binário, que é obtido por meio da comparação desse pixel com seus vizinhos. Mais especificamente, forma-se uma matriz de nove pixels, no qual o pixel central é comparado com todos os oito pixels vizinhos ao pixel central. Para os pixels com valor maior do que o pixel central, atribui-se "1", caso contrário, atribui-se "0". Após esse processo, um número binário é gerado, considerando que a matriz foi avaliada em sentido horário, com o primeiro pixel sendo o bit menos significativo. Uma ilustração da aplicação do operador LBP em um pixel de uma imagem é apresentada na Figura 3.2. Neste exemplo, qual o valor resultante gerado após a aplicação do LBP é 184 [5]. Em segundo momento na Figura 3.2, mostra-se a imagem resultante após a aplicação do LBP.

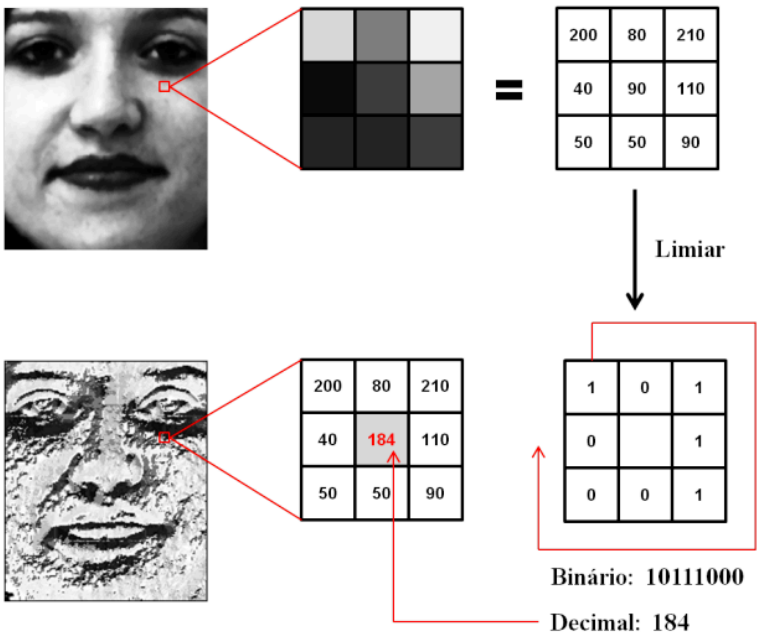

Figura 3.2: Processo do operador LBP em um pixel [5]

Naturalmente, o LBP pode ser computado em áreas maiores do que 3x3 pixels. Neste caso, cria-se uma circunferência de raio R, que pode ser ajustado. Também se determina o número de pontos amostrais P sobre a circunferência. A comparação se dá entre o pixel principal central e as posições amostradas no círculo com a imagem interpolada. Este procedimento é ilustrado ilustrado na Figura 3.3 para alguns valores de P e R.

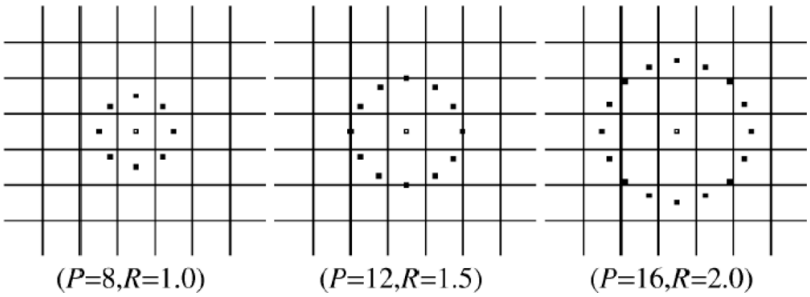

Figura 3.3: Exemplo de operador LBP com diferentes valores de P e R [6]

A seguinte equação define o operador LBP nos termos de R, P e os valores em escala de cinza dos pixels da imagem:

$$LBP_{P,R} \begin{cases} \sum_{p=0}^{P-1} S(g_p - g_c) & \text{se número de transições} \leq 2 \\ P+1 & \text{caso contrário} \end{cases} \tag{3.1}$$

sendo g_c o valor do pixel central e g_p, o do pixel que está sendo comparado. Assim, obtém-se a sequência de valor binário $T_p = S(g_0 - g_c), ..., S(g_{P-1} - g_c)$, em que $S(x) = 0$ se $x < 0$ e é

igual a 1 caso contrário [56].

Para tornar o operador LBP invariante à rotação, pode-se modificar a ordem com que coleta-se os bits, de forma a alterar a sua representação binária. Mais especificamente, escolhemos como a primeira posição entre os pontos amostrais aquela que gera o menor valor final. Isso faz com que a possibilidade de valores para o LBP seja de 36 variações no total [57].

Pode também haver uma predominância da ocorrência de valores binários considerados uniformes, ou seja, aqueles em que são poucas, ou inexistentes, as transições de 0 para 1. Desta forma, a quantidade de classes de texturas pode ser reduzida quando agrupam-se todos os LBPs não-uniformes em uma única classe.

De forma geral, após o cálculo do LBP para todos os pixels da imagem, calcula-se o histograma desta nova imagem. O histograma é um descritor com a principal função de apresentar de forma compacta as características da imagem. O histograma usado para o LBP, enquanto uma função discreta, pode ser descrito pela seguinte equação:

$$h(r_k) = n_k \tag{3.2}$$

onde r_k é o valor de uma das classes do LBP e n_k é a quantidade de pixels que apresentam o referido valor do LBP [58].

No entanto, é comum utilizar a versão normalizada do histograma que é descrita por:

$$p(r_k) = \frac{n_k}{MN} \tag{3.3}$$

em que M e N representam as dimensões da imagem.

Sendo assim, ao final do processo, todos os histogramas da imagem são concatenados ao final do processo, de forma a se tornarem um único vetor de atributos daquela imagem.

3.3 CLASSIFICADORES

Os atributos extraídos das imagens do conjunto formam vetores, um para cada imagem, de forma a caracterizá-las. Esses são os dados de entrada para o modelo classificador. As características destes dados de entrada são analisados na fase de treinamento do modelo para ajustar o classificador, de forma que ele consiga realizar a classificação correta de novos dados de entrada.

Neste projeto, os modelos escolhidos para classificação no sistema foram: Máquina de Vetor de Suporte (SVM, em inglês *Support Vector Machine*) em primeiro momento e, posteriormente, a Floresta Aleatória (*Random Forest Classifier*). Ambos são apresentados mais detalhadamente nas seções seguintes.

3.3.1 Máquina de Vetor de Suporte

As Máquinas de Vetor de Suporte são um modelo de aprendizado com grande popularidade na comunidade de Aprendizado de Máquina. A SVM é utilizada em variadas aplicações, desde bioinformática ao reconhecimento de imagens, como é o caso desse projeto [59] [60] [61].

Intrinsicamente, a SVM é um modelo de classificação binária em problemas de aprendizado supervisionado (*supervised learning*). Concedido um conjunto de dados para treinamento, todos especificados como pertencendo a uma das duas categorias, o treinamento do algoritmo SVM gera um modelo que irá classificar novos dados como sendo de uma categoria ou de outra.

A SVM baseia-se em um mapeamento de todos os dados de treinamento, representando-os como pontos em um espaço. Os exemplos para treinamento são divididos nesse espaço de acordo com sua categoria de tal forma que os dois conjuntos de pontos estejam distanciados pelo maior espaçamento possível (d), como ilustrado na Figura 3.4. Sendo assim, um novo dado, quando fornecido ao sistema, é mapeado nesse mesmo espaço e a predição de sua classe é feita de acordo com a sua localização no espaço, que foi gerado no treinamento do modelo.

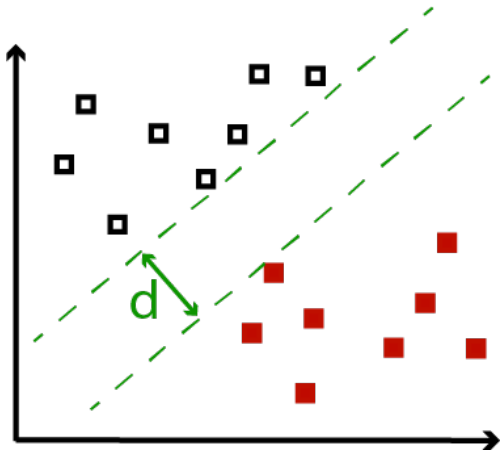

Figura 3.4: Ilustração da distância dos subconjuntos de dados de treinamento mapeados no espaço pelo modelo SVM.

Em sua versão linear, a SVM pode ser classificada como "Margens Rígidas"ou "Margens Suaves". Em ambos casos, definem-se fronteiras lineares (para aplicação do limiar de decisão) no conjunto de dados que aprensentam perfil linearmente separável. As margens são utilizadas na SVM para encontrar a solução de classificação para os subconjuntos de forma que o erro de generalização seja o menor possível. Desta forma, as margens são definidas como a menor distância entre o hiperplano (fronteira) de decisão e qualquer uma das amostras de treinamento [30].

No entanto, apesar de não se aplicar a este projeto, a SVM também pode ser empregada em problemas em que os dados de treinamento não são rotulados, ou seja, em caso de aprendizado não-supervisionado (*unsupervised learning*). Para tal, há uma versão distinta da SVM que funciona baseada no mapeamento dos dados em um espaço. No entanto, inicialmente realiza-se um processo para encontrar as conexões naturais entre os dados de entrada, formando grupos de

acordo com o grau de similaridade apresentado entre eles, para então distribuí-los no espaço. Essa técnica é um algoritmo de aglomeração (*clustering*) conhecido como *Support Vector Clustering* [62].

3.3.1.1 SVM com Margens Rígidas

Pode-se definir um conjunto de dados de treinamento *C*, composto de dados $x_i \in$ X, no qual *i = 1, ..., n* e X é o espaço de dados, cujos rótulos de categoria são $y_i \in$ Y, onde Y = $\{-1,+1\}$. Sendo assim, o conjunto *C* é dito linearmente separável caso possa-se criar um hiperplano tal que os subconjuntos de dados com valor -1 seja separado do subconjunto de dados $+1$ [4].

A seguinte equação, representa um hiperplano qualquer e divide o espaço dos dados de treinamento em duas regiões, uma para representar -1 e outra, $+1$,

$$f(x) = \boldsymbol{w} \cdot \boldsymbol{x} + b = 0 \tag{3.4}$$

Desta forma, define-se uma função $g(x)$ como a função sinal de $f(x)$ que irá ser o padrão para a classificação de novos dados de entrada, dada pela seguinte equação:

$$g(x) = sgn(f(x)) = \begin{cases} +1 & \text{se } w \cdot x + b > 0 \\ -1 & \text{se } w \cdot x + b < 0 \end{cases} \tag{3.5}$$

na qual $\boldsymbol{w}$ é o vetor normal ao hiperplano criado.

Para $f(x) = 0$, a distância entre um ponto no espaço (amostra) e o hiperplano definido é dada por $|f(x)|$ / $||w||$. Além disso, deve existir pelo menos um conjunto de parâmetros $\boldsymbol{w}$ e *b* que satisfaça *g(x)*, ou seja, tal que $y_i \cdot f(x_i) > 0$. Sendo assim, pode-se definir a distância entre uma amostra x_i qualquer e o hiperplano escolhido como na seguinte equação:

$$\frac{y_i \cdot f(x_i)}{||\boldsymbol{w}||} = \frac{y_i(\boldsymbol{w} \cdot x_i + b)}{||\boldsymbol{w}||}. \tag{3.6}$$

As margens (H_1 e H_2) são definidas, em primeiro momento, mediante a distância perpendicular entre a fronteira de decisão e aquele que é o ponto mais próximo de todas as amostras dos dados de treinamento, como mostra a parte *a* da Figura 3.5. Posteriormente, as margens são maximizadas escolhendo-se a sua localização, tomando como base as amostras que se encontram mais próximas do hiperplano separador, como destacado na parte *b* da Figura 3.5. Essas amostras são denominadas Vetores de Suporte (*support vectors*), dando nome à técnica de aprendizado. Para maximizar a distância entre os vetores de suporte e o hiperplano separador ($f(x) = 0$), deve-se otimizar os parâmetros $\boldsymbol{w}$ e *b*, que são ajustados durante o treinamento do modelo [63].

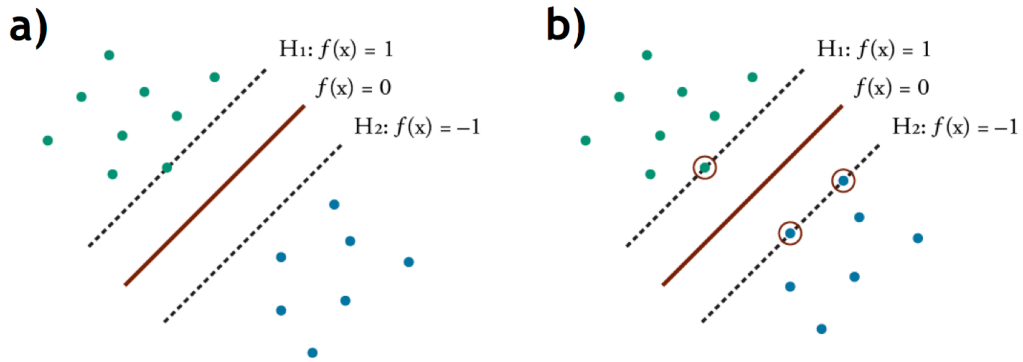

Figura 3.5: Ilustração do hiperplano separador e do estabelecimento das margens (H_1 e H_2) no espaço em que os dados de treinamentos estão mapeados.

Realizando um escalonamento, a partir de uma mesma constante em termos de $\boldsymbol{w}$ e em b não altera-se a distância dos pontos x_i em relação ao hiperplano separador, a expressão $y_i(\boldsymbol{w} \cdot x_i + b)$ pode ser considerada unitária. Desta forma, a distância mínima entre o hiperplano e os dados de treinamento é de $1/||w||$. Busca-se, então, maximizar essa distância com a minimização de $||w||$, valendo-se do problema de otimização na expressão seguinte

$$\min\{\frac{1}{2}||\boldsymbol{w}||^2\}, \tag{3.7}$$

onde são consideradas as seguintes restrições

$$\text{Restrições: } y_i(\boldsymbol{w} \cdot x_i + b) \geq 1, i \in (1, n) \tag{3.8}$$

de forma a garantir que não haverão dados de treinamento entre as margens de separação de classes, daí então a nomenclatura "margens rígidas"[4].

Esta é uma importante propriedade da SVM, tornando a determinação dos parâmetros do modelo uma otimização convexa, na qual a solução mínima é obrigatoriamente a solução global [30].

Para solucionar o problema de otimização da equação 3.7, pode ser utilizada uma função Lagrangiana, associando parâmetros a_i (multiplicadores de Lagrange) e tornando suas derivadas parciais nulas. A resolução completa deste problema pode ser encontrada nas referências [30] e [4]. Valendo-se do resultado dessa operação, tem-se a definição da função utilizada para a classificação dos dados que serão fornecidos ao sistema, dada pela seguinte equação

$$g(x) = sgn(f(x)) = sgn\left(\sum_{i=1}^{n} a_i y_i k(\boldsymbol{x}, \boldsymbol{x_i}) + b\right) \tag{3.9}$$

sendo a função sinal do resultado da função de Lagrange aplicada nesse caso.

Pode-se perceber que na Equação 3.9 empregou-se o conceito de *kernel* na resolução final do problema de otimização. Define-se a função *kernel* como $k(x, x') = \phi(\boldsymbol{x})^T\phi(\boldsymbol{x'})$, na qual $\phi(\boldsymbol{x})$ indica um mapeamento, ou transformação, para um determinado espaço de características (*feature space*), $\phi : X \mapsto \eth$. Apesar de, com o uso da função *kernel* na formulação, o esforço computacional tornar-se mais elevado, dá-se flexibilidade ao modelo que pode ser reformulado usando diferentes tipos de *kernel* e, então, pode ser aplicado em outros casos nos quais a dimensionalidade excede o número de amostras de treinamento [30].

3.3.1.2 SVM com Margens Suaves

A SVM com Margens Suaves é uma adaptação da versão do algoritmo com Margens Rígidas para que o aprendizado seja mais eficiente em casos de dados reais, os quais, comumente, não se apresentam conjuntos de dados estritamente bem comportados, ou seja, linearmente separáveis. Com essa versão da SVM, propõe-se lidar com conjuntos de treinamento mais gerais. Para isso, exceções às restrições apresentadas na expressão 3.8. Sendo assim, são estabelecidas variáveis de folga (ξ_i, $i \in (1, n)$) e são adicionadas ao problema e a equação 3.8 é modificada para:

$$y_i(\boldsymbol{w} \cdot x_i + b) \geq 1 - \xi_i,\ \xi_i > 0,\ i \in (1, n). \tag{3.10}$$

O efeito dessa modificação é que, devido a essa folga adicional, alguns pontos dos dados de treinamento podem permancer entre as margens H_1 e H_2. Por outro lado, isto pode gerar erros de classificação [4].

A forma de desenvolvimento do modelo é similar à SVM com Margens Rígidas, ou seja, também emprega-se a função de Lagrange, anulando suas derivadas parciais. No entanto, os parâmetros multiplicadores de Lagrange a_i sofrem modificações. A equação 3.7 é, nesse caso, alterada:

$$\min\{\frac{1}{2}||\boldsymbol{w}||^2 + K\sum_{i=1}^{n}\xi_i\}. \tag{3.11}$$

Sabe-se que um erro no conjunto de treinamento é indicado por $\xi_i > 1$, portanto, para que o erro gerado por ξ_i seja minimizado, no qual a constante *K* é um termo de regularização à minimização dos erros no conjunto de treinamento.

Como no caso anterior, os pontos denominados Vetores de Suporte são aqueles em que a condição $a_i > 0$ é satisfeita. Porém, neste caso, há outros tipos de Vetores de Suporte, denominados livres e limitados, que atendem às equações a seguir

$$a_i(y_i(\boldsymbol{w} \cdot \boldsymbol{x_i} + b) - 1 + \xi_i) = 0, \tag{3.12}$$

$$(K - a_i)\xi_i = 0. \tag{3.13}$$

Os vetores de suporte chamados livres são aqueles que se encontram exatamente em cima da margem, obedecendo $a_i \leq K$ e $\xi_i = 0$. Já os vetores de suporte limitados possuem $a_i = K$ e podem ser erros ($\xi_i > 1$) ou pontos classificados corretamente ($0 < \xi_i \leq 1$), porém, se localizando entre as margens. Esses vetores são ilustrados na Figura 3.6.

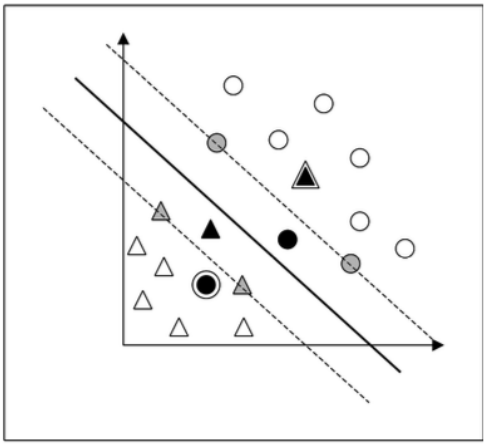

Figura 3.6: Ilustração dos tipos de vetores de suporte: livres (cinza) e limitados (preto) [4].

Em conclusão, obtém-se a mesma equação de classificação do modelo anterior (equação 3.9), porém os parâmetros a_i são calculados de forma distinta, com restrições que dependem da constante *K*.

3.3.1.3 SVM não-linear

Apesar do modelo SVM com Margens Suaves ser mais abrangente do que aquele com Margens Rígidas, este modelo corresponde a um modelo de separação de categorias no qual um hiperplano é obtido, ou seja, é necessária uma característica linear na distribuição dos dados. No entanto, em aplicações reais, em alguns casos pode não ser possível seprar o subconjunto de dados por um hiperplano, como exemplificado na Figura 3.7.

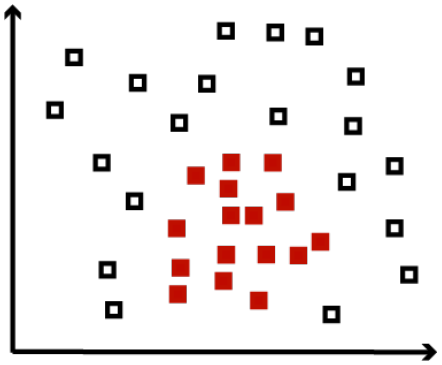

Figura 3.7: Ilustração de conjunto de dados de treinamento que não é linearmente separável

Desta forma, atentando-se para a utilização da função *kernel* (ver equação 3.9), percebe-se que é possível empregar a SVM em uma classificação não linear, empregando um "truque de *kernel*" (*kernel trick*) que se resume em mapear os dados de entrada em um espaço com dimensão superior, denominado espaço de atributos (*feature space*), como apresentado na Figura 3.8 [64] [65]. Neste caso, observa-se $\phi : X \mapsto \eth$, no qual $\eth$ é o novo espaço de características. O bom desempenho do classificador depende de uma escolha apropriada de ϕ, para que uma divisão linear nos dados de treinamento possa ser realizada.

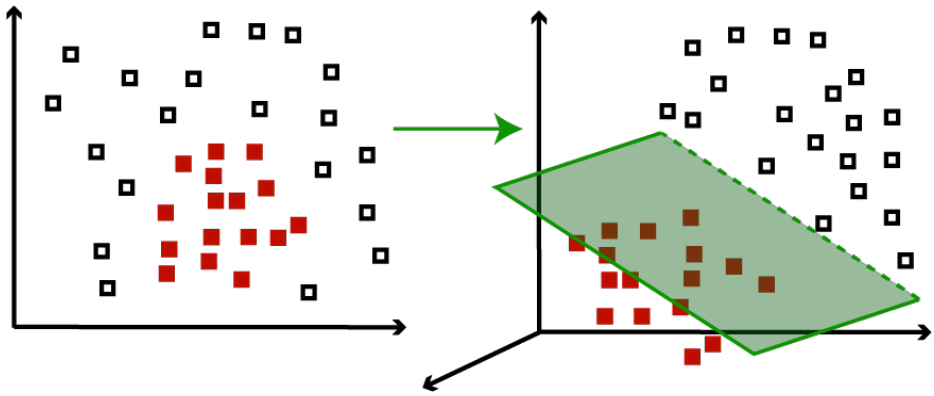

Figura 3.8: Ilustração de mapeamento de conjunto de dados de treinamento em novo espaço de características a partir da utilização de função *kernel*, possibilitando a divisão linear entre as categorias.

Segundo o teorema de Cover, para que os dados mapeados para o novo espaço de características tenham alta probabilidade de serem linearmente separáveis, duas condições devem ser satisfeitas [66]. Primeiramente, a transformação (ϕ) deve ser não-linear e, em segundo lugar, a dimensão do espaço de características deve ser suficientemente alta.

No processo de treinamento, ocorre o mapeamento dos dados de treinamento para o espaço de caracteristicias aplicando-se ϕ no conjunto de dados. Com o objetivo de obter um nível relativamente baixo de ruídos, emprega-se a SVM com Margens Suaves. Após isso, as equações para otimização são calculadas, com a aplicação do operador ϕ. Sendo assim, o classificador é o descrito na equação 3.9.

Comumente, aplica-se o *kernel* sem necessarimente a determinação prévia do operador ϕ, deixando-o como uma operação implícita durante o treinamento. Esse é mais um motivo para usar a função *kernel*, uma vez que se trata de um operador com cálculo simples (produto escalar) que representa bem os espaços. Para garantir que as condições expostas nesta seção sejam cumpridas, utilzam-se alguns tipos de *kernel* que satisfazem as condições de Mercer [4]. As condições de Mercer são satisfeitas em um *kernel* caso ele gere matrizes $\boldsymbol{K}$, em que cada elemento é definido por $K_{i,j} = K(\boldsymbol{x_i}, \boldsymbol{y_j})$, para todo $i, j = 1, ..., n$. Todos os *kernels* que foram utilizados neste projeto são apresentados na Tabela 3.1.

Tipo	Função $k(x, x')$	Parâmetros
Linear	$(\delta(x \cdot x') + \kappa)$	δ, κ
Polinomial	$(\delta(x \cdot x') + \kappa)^d$	δ, κ, d
RBF	$\exp(\frac{-\|\|x-x'\|\|^2}{2\sigma^2})$	σ
Sigmoidal	$\tanh(\delta(x \cdot x') + \kappa)$	δ, κ

Tabela 3.1: Funções *kernel* e seus parâmetros, utilizadas neste projeto [4] e [7]

3.3.1.4 SVM Multiclasse

Apesar de se tratar de um classficador que é intrinsicamente binário, a SVM pode também ser aplicada em problemas que apresentem várias categorias. Para tal, foram desenvolvidas ao longos dos anos novas soluções para a multiclassificação a partir de combinações múltiplas do classificador binário explicitado previamente nesta seção. Alguns modelos de adaptação do SVM para variadas classes são Um-contra-um (OvO, do inglês *one vs. one*), Gráfico Acíclico Dirigido (DAG, do inglês *Directed Acyclic Graph*), Código de Correção de Erro de Saída (ECOC, do inglês *Error Corrected Output Coding*) e Um-contra-todos (OvA, do inglês *One vs. All*), sendo este último utilizado neste projeto.

Supondo que um conjunto de dados possui m classes, empregando a abordagem Um-contra-todos, um total de m classificadores binários SVM devem ser criados para efetuar a classificação de uma determinada categoria em relação às $m - 1$ demais classes. Então, no momento de testes ou de classificação de novos dados, esses dados são classificados criando-se uma margem (limiar de decisão) a partir do hiperplano separador e a classe desse dado será aquela que possuir uma função de decisão com maior valor, que corresponde à SVM que possui a maior margem dentre os demais [67] [30].

Sendo assim, para um problema deste tipo, devem ser determinados m hiperplanos. Consequentemente, é necessária a solução de m problemas de otimização, sendo que, em cada um deles, uma classe é separada das demais. Desta forma, esta é uma desvantagem dessa abordagem, já que, durante a fase de treinamento, este processo é computacionalmente oneroso. Ademais, supondo que todas as classes possuam uma quantidade igual de amostras para treinamento, já que a SVM se comporta como um classificador binário, uma classe terá bem menos amostras para treino do que a outra classe, apresentando uma razão de $1 : (m - 1)$ [67].

3.3.2 Floresta Aleatória

O método Floresta Aleatória foi inicialmente proposto por Breiman [68] e pode ser aplicado tanto em problemas de classificação, quanto em problemas de regressão. O modelo classificador consiste em um conjunto de árvores aleatórias de decisão que são criadas durante o processo de treinamento do modelo, sendo escolhidos aleatoriamente os atributos dentro do vetor de entrada fornecido ao classificador [69].

Passa-se, então, para o cálculo individual da entropia pertencente aos atributos. O atributo que

apresenta a maior entropia é escolhido como sendo o nó que será utilizado na separação de classes nesta determinada posição na árvore. No caso deste modelo, o resultado final de classificação é obtido por meio da soma ponderada de predições sugeridas por diferentes conjuntos de parâmetros pertencentes às variadas árvores que compõem a floresta [57].

O método Floresta Aleatória possui várias vantagens, dentre elas [68]:

1. É um modelo relativamente robusto a ruídos;
2. É simples e é facilmente implementada de forma paralela;
3. Nas diversas aplicações na área, este método tem apresentado um melhor desempenho que outros em relação ao índice de acerto de classificação, métodos tais como a SVM e as Redes Neurais Artificias [70].

3.3.2.1 Árvores de Decisão

As árvores de decisão são uma técnica utilizada para a construção de modelos de classificação, que distribuiem os exemplos em um número finito de categorias. A técnica realiza a análise de um atributo, realizando a subdivisão dos dados do conjunto.

Sua estrutura se assemelha à uma árvore invertida, começando por um nó raiz e se subdividindo em ramos e em outros nós, até chegar em suas folhas, que são os pontos finais do sistema. Os nós internos são os nós de decisão e realizam o teste de atributos. Cada ramo está relacionado ao valor do atributo e, por último, cada folha representa uma classe [71].

Em geral, constroi-se uma árvore de decisão com um conjunto de dados de entrada, a partir do qual o nó raiz é criado e, após a realização de um teste, divide-se a árvore em ramos. Esse processo de avaliação do nó e da divisão em ramos é realizado nos nós subsequentes também, continuando a ramificação da árvore. Esse processo é realizado até que se atinja uma folha, não havendo mais ramificação. Todo esse procedimento é realizado recursivamente para conjunto de dados [72].

3.3.2.2 Procedimento da Floresta Aleatória

A técnica de Floresta Aleatória é composta por um conjunto de árvores de decisão, no processo de crescimento de cada árvore é usado algum tipo de técnica aleatória. O conjunto de dados completo é dividido em subconjuntos, que serão os dados iniciais fornecidos à cada árvore. A aleatoriedade pode ser empregada durante o processo de divisão do conjunto de dados e na seleção do teste de cada nó.

No algoritmo de Floresta Aleatória, as árvores são binárias e os testes para decisão em cada nó podem ser escolhidos de forma aleatória, independentemente dos dados, ou por meio de um algoritmo que escolhe o teste que separa de melhor forma as amostras de treinamento. Neste caso, faz-se uma avaliação de acordo com a seguinte equação

$$\Delta E = -\sum_{i} \frac{|Q_i|}{|Q|} E(Q_i) \tag{3.14}$$

na qual ΔE é o ganho de informação, que é gerado quando um conjunto de dados Q é dividido em subconjuntos Q_i.

$E(q)$ é a entropia apresentada na seguinte expressão

$$E(q) = -\sum_{j=1}^{N} p_j log_2(p_j) \tag{3.15}$$

na qual p_j é a proporção dos exemplos em q que pertencem à classe j. Todo esse processo é recursivo e seu critério de parada pode ser pré-determinado escolhendo-se a quantidade de estimadores que serão utilizados na floresta, como no caso deste projeto. Outro critério de parada que pode ser utilizado é avaliar quando um nó não recebe exemplos suficientes.

Neste classificador, utiliza-se o conceito de probabilidade a posteriori, que se trata de probabilidade condicional de determinado evento aleatório. Durante o processo de treinamento do modelo, a probabilidade a posteriori é calculada para cada classe c em cada folha l, que pertence à determinada árvore a. Essa probabilidade pode ser calculada como mostra a seguinte equação

$$P_{a,l}(Y(I) = c) = \frac{N_{c,l}}{N_l} \tag{3.16}$$

sendo que $Y(I)$ é a classe c de determinanda imagem I, $N_{c,l}$ é o número de imagens da classe c que chegaram à folha l e, por último, N_l é o número do total de imagens que alcançaram a folha l [72].

O processo de treinamento, avaliando as imagens do conjunto de treinamento, é inicializado no nó raiz e, então, passa pelas árvores de decisão até atingir alguma folha. Sendo assim, a classe daquela imagem é definida tomando-se o argumento máximo da média artimética de todas as probabilidades a posteriori.

No capítulo seguinte, retrata-se a metodologia utilizada no desenvolvimento deste projeto, relatando os processos de composição do banco de imagens, bem como os estágios de pré-processamento, extração de atributos e de treinamento utilizando-se os classificadores descritos no presente capítulo.

4 METODOLOGIA

Neste capítulo são descritas as etapas de desenvolvimento do projeto que já foram implementadas, tendo em vista a proposta apresentada para construção de um produto de tecnologia assistiva para ensino de tipografia a deficientes visuais, mais especificamente, do sistema de classificação de tipografias.

O processo de desenvolvimento do sistema se deu em três etapas principais, que são listadas a seguir:

1. Estruturação do projeto;
2. Composição do banco de dados (imagens das diversas tipografias);
3. Implementação do algoritmo para classificação das imagens.

A primeira etapa deste trabalho foi caracterizada por debates para a idealização do produto final de tecnologia assistiva oferecido ao deficiente visual. A segunda e a terceira etapas, por sua vez, foram desenvolvidas e aprimoradas simultaneamente. O processo de desenvolvimento adotado nessas etapas é explicitado abaixo.

Todos os algoritmos implementados para este projeto, bem como o banco de imagens, podem ser encontrados no *link*: *github.com/fegvilela/TCC2*.

4.1 ESTRUTURAÇÃO DO PROJETO

Como primeiro estágio de desenvolvimento do projeto, foram discutidos modelos para a estruturação do produto e realizadas pequenas entrevistas não-estruturadas com deficientes visuais, sendo eles possíveis usuários do produto ou não.

Em oportunidades de teste, descritos e implementados no trabalho de Cruz [2], para aperfeiçoamento das peças táteis no desenvolvimento do produto, também foram realizadas as entrevistas com os deficientes visuais. O foco dessas entrevistas foi o maior entendimento sobre o contato real dos deficientes com o computador e com a informática. Nestas ocasiões, foram questionados aspectos sobre os leitores de tela disponíveis no contexto brasileiro, além de a popularização do sistema operacional *DOSVOX*. O objetivo dessas questões foi direcionar o desenvolvimento futuro do sistema, para que ele seja adaptado às condições reais de tecnologia dos deficientes visuais brasileiros. Além disso, houve a apresentação da proposta do sistema, a qual foi avaliada positivamente.

Vale ressaltar que o contato realizado com estes deficientes visuais foi importante para a idealização e desenvolvimento do projeto até o momento, sendo um vínculo importante para o segui-

mento deste posteriormente.

4.2 COMPOSIÇÃO DO BANCO DE IMAGENS

Passando-se então para as etapas de implementação do sistema, como já descrito, foram escolhidas nove tipografias para o projeto e, para cada uma delas, construído um conjunto de imagens. Para melhor organização, o banco foi montado tipografia a tipografia. As imagens que compõem o banco de dados em sua versão final são caracteres separados de cada tipografia, amostras podem ser vistas na Figura 4.1.

A principal fonte de obtenção das imagens originais foi o *Fonts in Use*, um website que disponibiliza fotografias de projetos envolvendo design gráfico que já estão classificadas por tipografia [73]. Além deste, foram utilizados sites de busca convencionais, utilizando a opção de imagens nas buscas. Ainda, para os casos em que poucas imagens foram encontradas, a saber as tipografias *Franklin Gothic* e *Adobe Jenson*, imagens complementares foram cedidas como cortesia pela estudante de design parceira neste projeto.

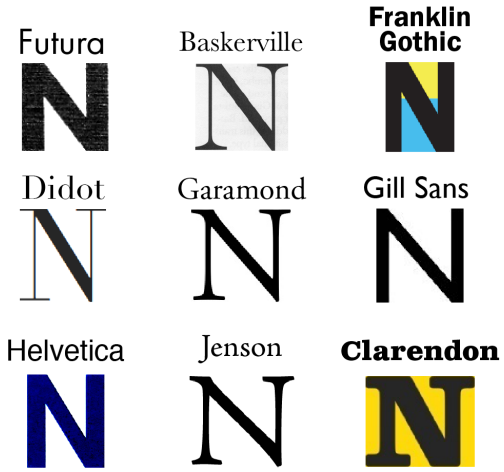

Figura 4.1: Amostras do banco de imagens, especificando-se a tipografia.

É comum que tipografias digitais apresentem várias versões distintas entre si. Uma vez que as tipografias digitais são representações da versão física original dos tipos, diversas fundidoras de tipos propõem seu próprio modelo representativo para aquela tipografia, ocasionando em variações nas versões digitais. Um exemplo utilizando a tipografia *Garamond* é apresentado na Figura 4.2. Escolheu-se, para ilustração, a letra "a", sendo possível verificar as diferenças no formato de três partes específicas da letra: bojo, remate e terminal em gota. A versão adotada no projeto é a *Adobe Garamond Pro*. Logo, antes de adquirirmos imaens deste tipo em websites, foi necessária uma análise manual para verificar se a imagem se tratava da tipografia correta, em sua versão empregada no projeto.

Sendo assim, a adição de uma imagem de uma outra versão da tipografia poderia gerar um

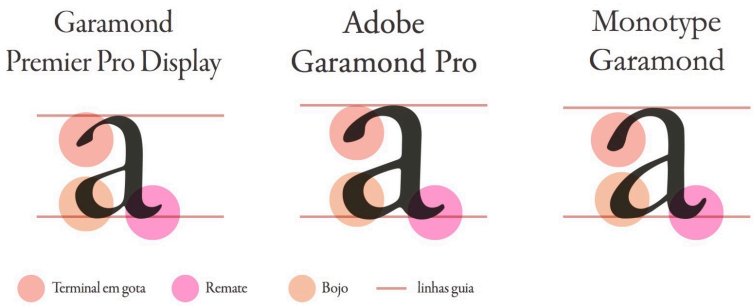

Figura 4.2: Comparação entre três versões da fonte Garamond, evidenciando as partes anatômicas distintas.

erro de classificação, já que algumas tipografias empregadas no projeto são similares entre si, fato ilustrado na Figura 4.3 que apresenta a similaridade entre duas tipografias presentes no projeto. Portanto, a análise foi realizada de forma minuciosa, utilizando conceitos da anatomia das letras, como o formato de suas serifas e o contraste entre as partes [74]. Ademais, as imagens originais obtidas muitas vezes continham variações que deveriam ser excluídas manualmente, como mais de uma tipografia em sua composição, ou então uma mistura entre os variados estilos dentro de uma família tipográfica: itálico, algumas variações de negrito e versalete.

Figura 4.3: Comparação entre as fontes *Adobe Garamond Pro* e *Adobe Jenson Pro*, atestando o alto índice de similaridade entre elas.

Para o processo de separação das imagens em caracteres únicos, implementou-se um algoritmo em Python para automatização do processo de reconhecimento da localização dos caracteres na imagem e de formação de novas imagens individuais, algo similar a "recortes" da imagem original. A principal biblioteca utilizada nesse algoritmo foi a *OpenCV* (*Open Source Computer Vision Library*), uma biblioteca de visão computacional e aprendizado de máquina disponível em código livre (*open source*). Ela foi escolhida por ser amplamente utilizada por especialistas da área e em empresas renomadas como Google, Microsoft, Intel e IBM [75].

O algoritmo desenvolvido é composto de uma seção principal denominada `mainBancoDados`, e de módulos importados, que incluem os métodos (estruturas em Python análogas à funções) implementados para desempenhar operações específicas nas imagens. O fluxograma do código é ilustrado na Figura 4.4. Ao executar o algoritmo, deve-se escolher em qual parte do banco de dados, ou seja, qual tipografia, deseja-se analisar. A seção principal dá acesso à execução de três operações (métodos implementados nesse projeto): recorte das imagens originais, exclusão das imagens de dimensão pequena e renomeação das imagens.

Inicia-se com a aplicação da operação de recorte de imagens (o método `imCrop`) caso as imagens não sejam letras individuais. Então, caso haja erro no reconhecimento das letras, sendo formada alguma imagem que não apresenta um único caractere por completo, passa-se para o método de exclusão de imagens (`imApaga`). Em seguida, utilizando o método `changeName`, realiza-se a renomeação das imagens, caso elas não possuam o nome adequado.

Vale ressaltar que a formatação padrão para o nome das imagens foi escolhida como: *numero_tipografia*. Por exemplo, o nome da sétima imagem presente no diretório da tipografia *Helvetica* será "7_helvetica". O caractere "_" foi escolhido nesse caso para que a classe da imagem, a tipografia, seja de fácil identificação pelo algoritmo de treinamento do modelo classificador.

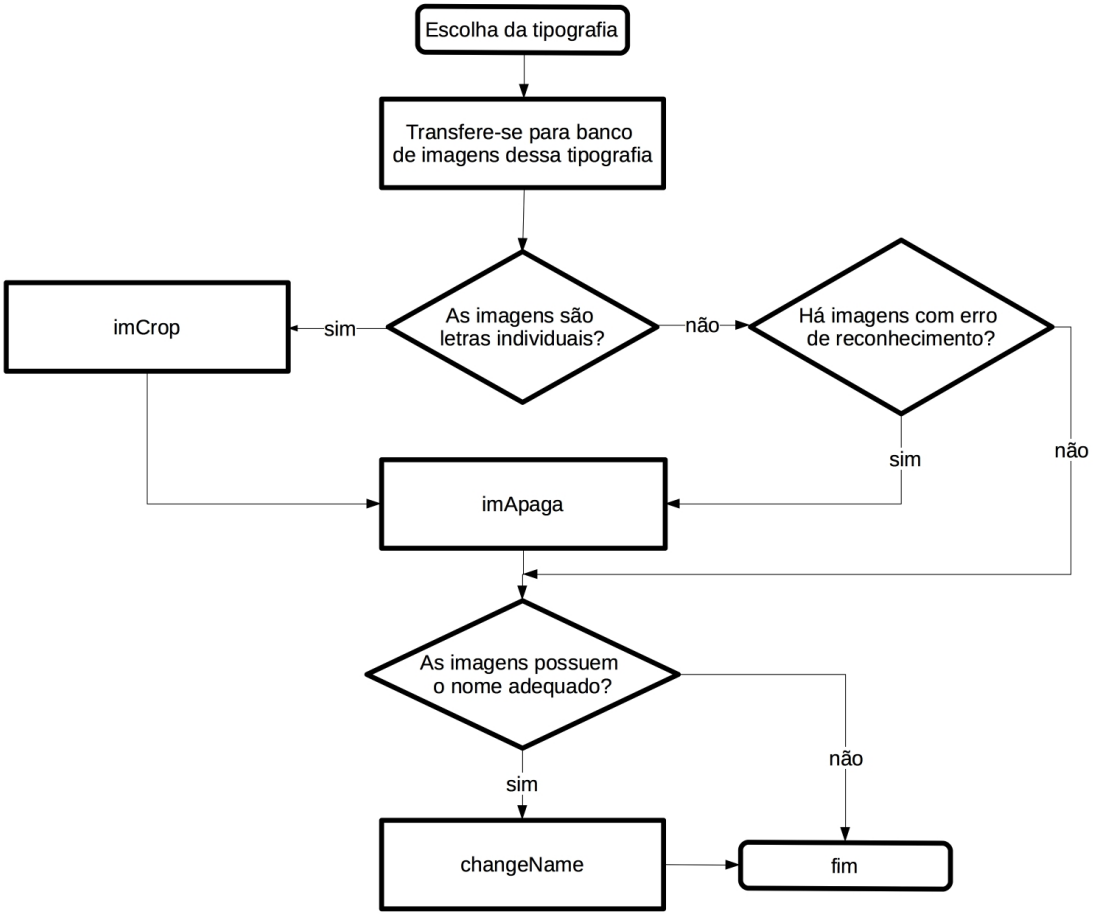

Figura 4.4: Fluxograma da seção principal do algoritmo para composição do banco de dados

Passa-se, então, para a descrição de cada um desses métodos separadamente. Em primeiro lugar, o método `imCrop` realiza o recorte das imagens originais, criando imagens individuais de caracteres. O fluxograma está apresentado na Figura 4.5. Quando chamado na seção principal, o método `imCrop` recebe como parâmetro de entrada o nome da fonte em que se deseja operar, já que o banco de imagens é dividido por tipografias. Então, o diretório de operação passa a ser o banco de imagens da fonte escolhida. Todas as imagens contidas naquele diretório são convertidas para o formato *PNG*, que é um dos formatos aceitos pela biblioteca *OpenCV*.

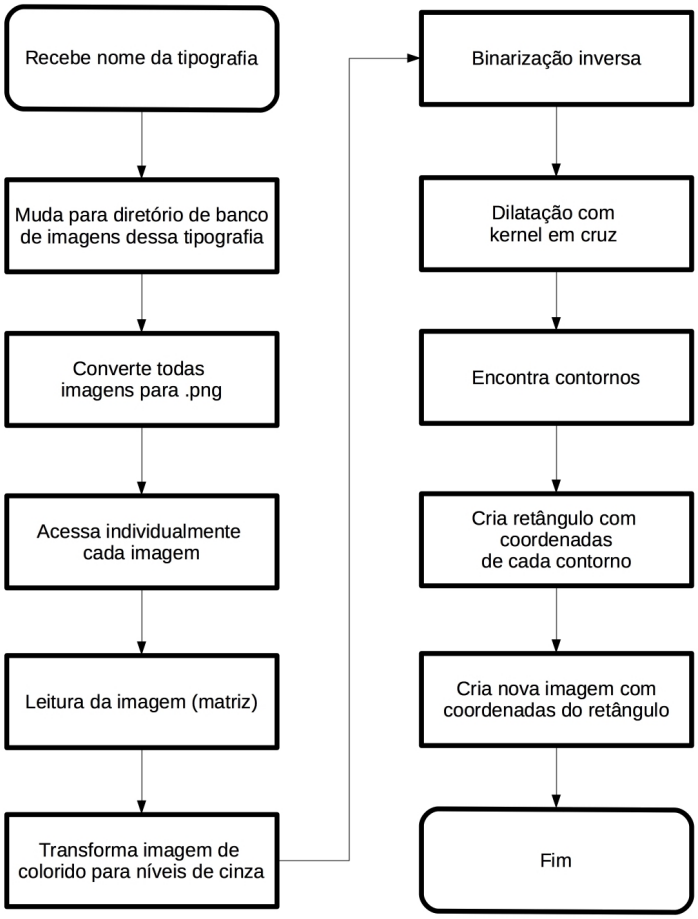

Figura 4.5: Fluxograma do método `imCrop` para recortar as imagens originais, criando imagens separadas de cada caractere.

As imagens são acessadas individualmente e lidas uma por vez. Os valores dos pixels de uma imagem são armazenados em uma matriz. A imagem colorida é convertida para níveis de cinza e, em seguida, é realizada a binarização inversa da imagem em tons de cinza, utilizando um limiar de nível 88, ou seja, valores acima de 88 são atribuídos como valor zero (preto) e valores abaixo, como valor unitário (branco). O próximo passo é a dilatação morfológica usando um *kernel* em cruz, com dimensão 3x3. Após isso, encontram-se os contornos de cada parte branca da imagem. São criados retângulos limitadores ao redor dessas partes da imagem, utilizando as coordenadas dos contornos e novas imagens são criadas a partir desses retângulos. Todo o processo é exemplificado na Figura 4.6 para uma imagem da fonte *Futura*.

Vale ressaltar que, por envolver um processo de binarização inversa no algoritmo, não foi possível efetuar a detecção de borda nas letras das imagens que, em escala de cinza, possuíam cor de fundo com tom mais escuro que aquele da letra. Portanto, se fez necessária uma correção na coloração de algumas imagens para melhor desempenho do algoritmo, processo realizado manualmente, no software *Pré-Visualização* do sistema operacional *Mac OS X* e também no software *Adobe Lightroom*. Além disso, em algumas imagens, as letras em texto encontravam-se muito próximas uma à outra, o que gerou a demanda de uma limpeza de partes residuais de outras

letras, após a formação das imagens de caracteres individuais.

Figura 4.6: Resultado por etapa do processo de reconhecimento de caracteres em imagens do banco de imagens. i)imagem original, ii)escala de cinza, iii)binarização inversa, iv)dilatação com *kernel* em cruz e v)contornos e retângulos limitadores

O próximo método utilizado na seção principal do algoritmo é `imApaga`, responsável por excluir as imagens que possuem ambas dimensões inferiores a 45 pixels. O fluxograma descritivo pode ser visto na Figura 4.7.

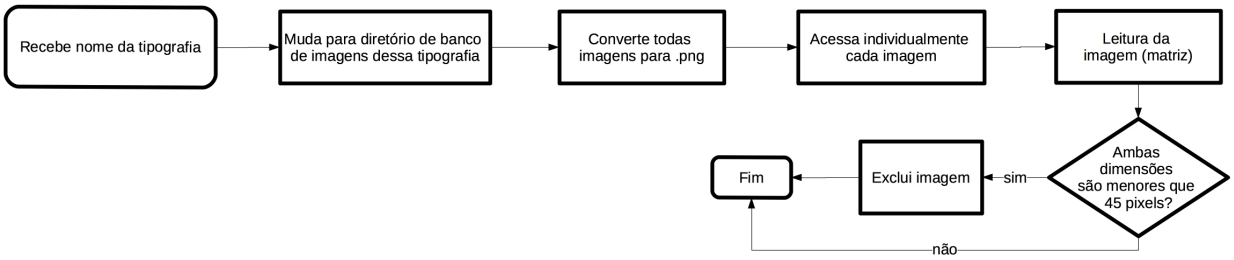

Figura 4.7: Fluxograma do método `imApaga` para excluir imagens com dimensões pequenas

Assim como no método `imCrop`, recebe-se como parâmetro de entrada o nome da tipografia em que se deseja operar e muda-se para o diretório do banco de imagens da tipografia. Faz-se também a conversão de todas imagens para *PNG*, que são depois acessadas individualmente e lidas, armazenando as intensidades dos pixels em uma matriz. Após isso, as dimensões da imagem são reconhecidas e avaliadas. Caso as duas dimensões sejam menores que 45 pixels, a imagem é excluída. Caso contrário, o programa passa para a próxima imagem, até finalizar a análise de todas as imagens presentes naquele diretório.

Por sua vez, o método `changeName` é utilizado para renomear as imagens presentes no diretório de acordo com a tipografia a que pertencem, seguindo o padrão estabelecido para o projeto (*numero_tipografia*). O fluxograma desse método é apresentado na Figura 4.8. Assim como os demais métodos dessa seção, começa-se recebendo o nome da tipografia em qual se deseja operar e transfere-se para o diretório do banco de imagens dessa tipografia.

A biblioteca (*OS*) em *Python* utilizada nesse método para a renomeação de arquivos pode gerar um problema durante o processo. Caso o novo nome escolhido para renomear um arquivo seja pertencente a um outro arquivo já existente no diretório, este arquivo prévio é excluído.

Portanto, para evitar que arquivos sejam excluídos erroneamente apenas por receberem nomes repetidos, primeiramente verifica-se se há o caractere "_" no nome dos arquivos. Caso negativo, utiliza-se a nomenclatura padrão supracitada, pois, dessa forma, não existirá arquivos prévios com nomenclatura semelhante a essa (*numero_tipografia*). Porém, caso a verificação apresente resultado positivo, nomeiam-se todos os arquivos com nomenclatura sem o caractere "_". Esse processo é realizado para garantir que, naquele diretório, nenhuma imagem possui a nomenclatura padrão, prevenindo-a de ser excluída. Então, posteriormente, utiliza-se a nomenclatura padrão em segundo processo automático de renomeação.

O banco de imagens, em versão final, constitui-se de 6750 imagens no total, sendo 750 imagens por tipografia, com dimensão mínima de 45 pixels em altura ou largura. No entanto, inicialmente, para as primeiras quatro tipografias, o banco contava com 1895 imagens para cada uma delas. Porém, muitas imagens possuiam dimensão pequena, apresentando baixa resolução e ruídos, fato que comprometia o treinamento e, consequentemente, a acurácia do classificador. Sendo assim, decidiu-se reduzir o tamanho do banco de imagens, exluindo automaticamente aquelas que fossem menor do que a dimensão desejada. Além disso, em alguns casos, aprouve-se operar uma exclusão manual de imagens provenientes de erros de reconhecimento realizado pelo algoritmo ou que apresentavam baixa resolução, apesar de dimensão maior do que 45 pixels.

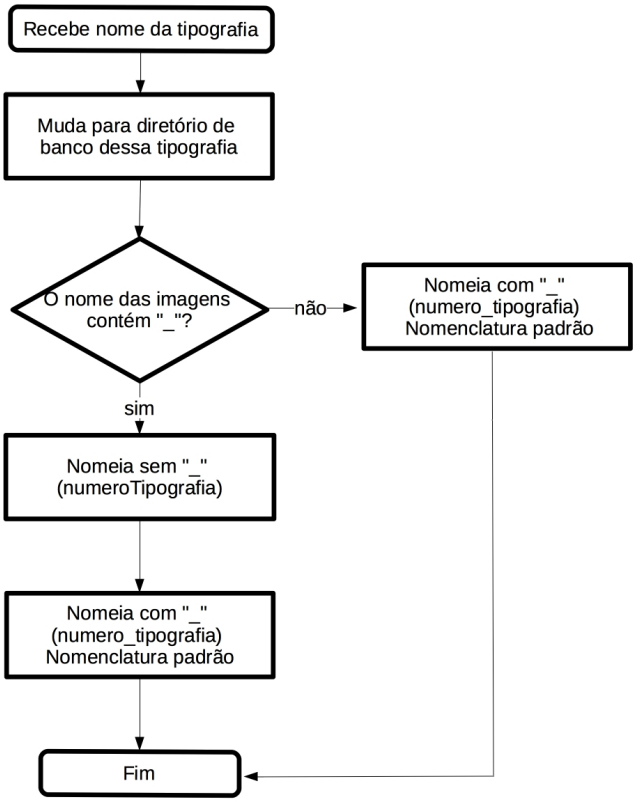

Figura 4.8: Fluxograma do método `changeName` para renomear imagens de determinado diretório, preparando-as para o treinamento do modelo.

4.3 ALGORITMO PARA TREINAMENTO DO MODELO

Essa seção destina-se à descrição do algoritmo implementado para treinamento do modelo da máquina, bem como das ferramentas utilizadas para seu desenvolvimento. O algoritmo é composto de três estágios listados a seguir, sendo que o resultado de um é o dado de entrada para o estágio subsequente:

1. Pré-processamento;
2. Extração de atributos;
3. Treinamento do modelo classificador e testes de predição.

Vale ressaltar que as etapas de pré-processamento e de extração de atributos são consecutivas e feitas imagem a imagem, até que todas as imagens do banco sejam avaliadas. Sendo assim, uma imagem passa pelo estágio de pré-processamento, em seguida, pelo estágio de extração de atributos, para que, então, siga-se para a próxima imagem, realizando o mesmo processo.

O algoritmo foi também implementado em Python utilizando como bibliotecas principais o *SciKit-learn*, o *SciKit-image*, o *NumPy* e o *OpenCV*. Todas as bibliotecas citadas, exceto a *OpenCV*, são derivadas de uma só coleção de bibliotecas, denominada *SciPy*, que apresenta softwares de código livre para computação científica em Python. No entanto, as bibliotecas *SciKit*, abreviação de *SciPy Toolkits*, são pacotes complementares ao *SciPy*, sendo desenvolvidas independentemente. O motivo de escolha dessas bibliotecas foi que eles são amplamente utilizados na área de processamento de imagens e aprendizado de máquina [76] [77] [78] [79].

4.3.1 Estágio de pré-processamento

Para essa etapa, a principal biblioteca utilizada foi o *OpenCV*, com a qual foi implementado um processo similar ao método `imCrop` descrito na seção anterior. Todos os passos do pré-processamento encontram-se ilustrados na Figura 4.9.

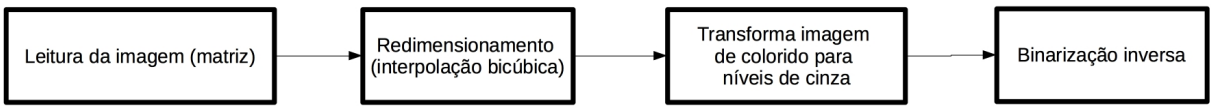

Figura 4.9: Etapas do estágio de pré-processamento do algoritmo de treinamento do modelo.

Inicia-se pela leitura da imagem, transformando-a em matriz, seguido de seu redimensionamento por interpolação bicúbica para altura fixa de 126 pixels, garantindo, assim, uma melhor uniformidade das imagens para o processo de extração de atributos. O valor ótimo da nova dimensão foi encontrado por meio de iterações e testes de acurácia. Para esse teste, criou-se um *loop* para alterar o valor da nova dimensão das imagens e, para cada valor de dimensão, o modelo era treinado e avaliada a acurácia. Desta forma, escolheu-se a dimensão que proporcionou maior

acurácia. Posteriormente, a imagem é convertida para escalas de cinza e, por fim, é realizada uma binarização inversa da imagem.

4.3.2 Estágio de extração de atributos

No estágio de extração de atributos, as bibliotecas usadas foram *NumPy* e *SciKit-image*. A biblioteca *NumPy* é diretamente associada ao *SciPy* e uma de suas vantagens principais é oferecer grande praticidade ao empregar matrizes, principalmente com uma vasta quantidade de elementos, por isso é utilizada em aplicações com imagens, como no caso deste projeto.

Dessa mesma forma, a biblioteca *SciKit-image* foi utilizada no algoritmo por ser uma biblioteca desenvolvida para processamento de imagem. Apenas um módulo da biblioteca, a saber `feature`, foi utilizado para implementar a extração de atributos por meio do *Local Binary Pattern* (LBP).

A seção de extração de atributos das imagens foi implementada como uma classe nomeada `LocalBinaryPattern`, em um módulo separado, e o modelo utilizado foi o LBP. Os parâmetros necessários para a sua aplicação na imagem, como explicado no capítulo 3, são: o raio (R) e a quantidade de pontos avaliados (P).

Sendo assim, esses valores devem ser fornecidos ao se utilizar a classe na seção principal do algoritmo. Os valores adotados nesse caso foram um raio de 9 unidades e 21 pontos avaliados. Estes valores foram encontrados por meio de um processo iterativo para determinar a combinação ótima. Foram criados dois *loops* em cascata para realizar a variação do valor de R e o do valor de P. Para cada combinação destes dois parâmetros, o modelo foi treinado e a acurácia avaliada. Assim, os valores de R e P que proporcionaram melhor acurácia foram escolhidos.

Além disso, implementou-se um método, `describe`, baseado no tutorial encontrado em [80], para que o LBP de determinada imagem seja computado. Todo o processo é ilustrado no fluxograma na Figura 4.10. Logo, para haver a extração de atributos de uma imagem, esse método é chamado na seção principal do algoritmo, fornecido a ele o raio e o número de pontos desejados para o modelo e a imagem que será avaliada. Em seguida, o modelo em sua versão uniforme é aplicado na imagem.

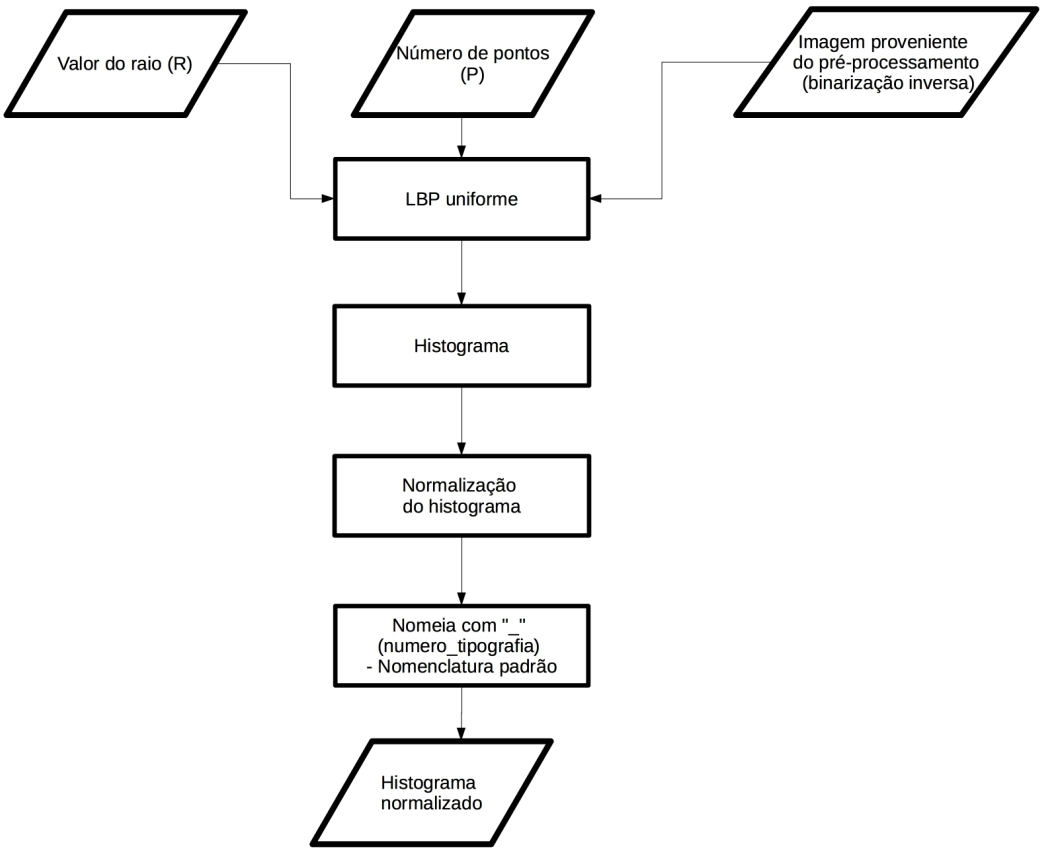

Figura 4.10: Fluxograma do método `describe` para computar o LBP de uma imagem, retornando seu histograma normalizado.

Utilizando-se a biblioteca *NumPy* para a estruturação e a manipulação dos vetores de atributos, o histograma da representação da imagem em LBP é então computado e normalizado. O resultado de todo esse processo aplicado às amostras de imagens do banco de imagens é apresentado na Figura 4.12. Na Figura 4.11, são apresentadas as imagens com binarização inversa, que são as imagens utilizadas como dados de entrada neste estágio (extração de atributos).

A imagem que está sendo analisada segue, então, para a determinação do rótulo referente à sua classe, processo realizado analisando o nome do arquivo. Após isso, o histograma normalizado do LBP, proveniente da extração de atributos, é armazenado em uma lista, denominada Lista de Dados. Em seguida, o rótulo é também armazenado em uma lista, denominada Lista de Rótulos, sendo relacionado ao histograma (representativo dos atributos) por partilharem de mesma posição nas listas de armazanamento. Esse processo é repetido a cada imagem. Toda essa etapa é descrita no fluxograma da Figura 4.13

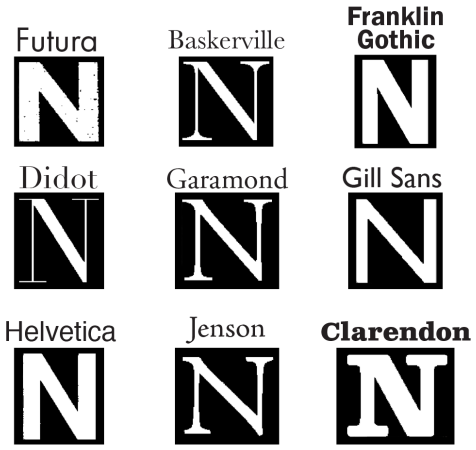

Figura 4.11: Amostras de imagens do banco de imagens após processo de binarização inversa. Imagens de entrada no estágio de extração de atributos.

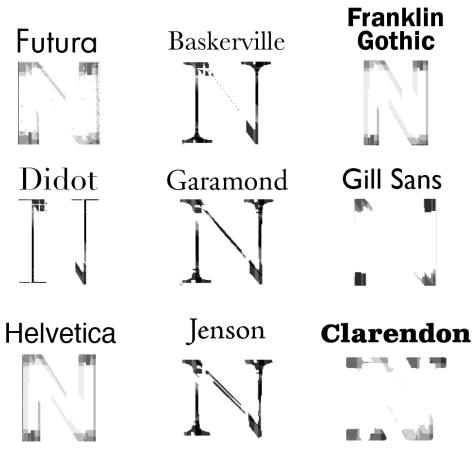

Figura 4.12: Amostras de imagens do banco de imagens após processo de extração de atributos, aplicação do LBP.

Para tornar o processamento mais eficiente, a Lista de Rótulos obtida a partir das imagens é convertida para uma lista formada por inteiros, na qual cada número representa uma classe.

Sendo assim, os elementos de saída deste estágio de extração de atributos são duas listas de dados: uma referente aos rótulos de cada imagem (Lista de Rótulos) e outra, aos histogramas provenientes do LBP (Lista de Dados). São essas duas listas que irão servir de dados de entrada para o estágio de treinamento do modelo.

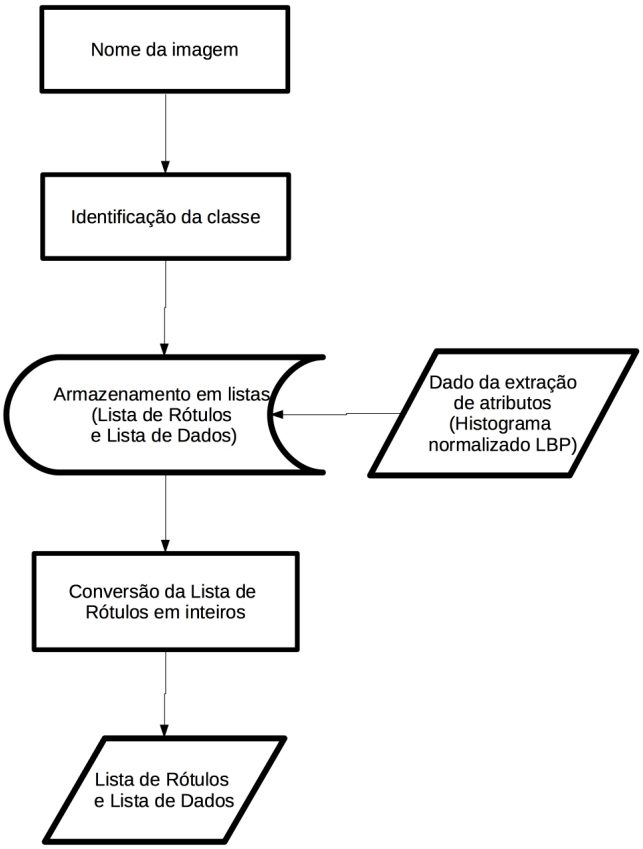

Figura 4.13: Fluxograma da formação da Lista de Dados e da Lista de Rótulos, ambas dados de entrada para o treinamento do modelo.

4.3.3 Estágio de treinamento do modelo classificador e testes de predição

Para o algoritmo desta etapa, a principal biblioteca utilizada foi a *SciKit-learn*, que disponibiliza uma série de ferramentas para várias etapas pertinentes ao aprendizado de máquina. Os módulos utilizados, sendo eles `svm`, `ensemble`, `model_selection`, `preprocessing`, oferecem os dois classificadores empregados, métricas para a escolha do modelo e outras ferramentas auxiliares, como a validação cruzada (*Cross Validation*). Todo o processo do estágio de treinamento do classificador e testes de predição é descrito no fluxograma da Figura 4.14.

Nesta etapa, dois modelos foram usados para a classificação das imagens. Primeiramente, utilizou-se o classificador SVM, variando a função *kernel*, a saber, em suas versões Linear, RBF, Sigmoidal e Polinomial. Porém, por apresentar, em todas as alternativas, um índice de acerto de classificação considerado baixo para a aplicação, fato explicitado no próximo capítulo, um novo modelo classificador foi adotado, a Floresta Aleatória.

Para o caso da SVM, foram fornecidos ao modelo alguns parâmetros para sua estruturação, nesse caso, o parâmetro de penalidade C do termo erro na classificação, adotado como 100 (o inverso do multiplicador de Lagrange), e o estado randômico (*random state*) para o embaralhamento das imagens, adotado o valor unitário. Para alguns *kernels*, foram utilizados parâmetros

específicos, como no caso do RBF, utilizou-se o parâmetro γ, com valor de 10950, que é a regulação da curvatura do hiperplano. O parâmetro γ também foi utilizado com o *kernel* Polinomial, com valor 19, e com o *kernel* Sigmoidal foi utilizado como valor automático, que é implementado como o inverso da quantidade de atributos.

Além disso, aplicando-se a SVM em uma multiclassificação, utilizou-se a abordagem um-contra-todos, ou seja, são treinados nove classificadores binários, em que uma tipografia é treinada como uma classe e as demais oito tipografias, como a outra classe. Para a classificação de uma nova amostra, são aplicados os nove classificadores binários, aquele que apresenta maior valor, é escolhido como a classe desta imagem.

Para o caso da Floresta Aleatória, os parâmetros fornecidos são o número de árvores de decisão (utilizou-se 82) e o número de tarefas que irão ser realizadas em paralelo (ajustou-se para a execução de número máximo de tarefas em paralelo).

Após isso, passa-se para o ajuste dos parâmetros da validação cruzada a ser utilizada, no caso *Stratified K Fold*. Utilizando essa implementação de validação cruzada, uma variação do modelo *K Fold*, no processo de divisão em subconjuntos (pastas), a porcentagem de amostras de cada classe é mantida de acordo com o conjunto original das imagens. Os parâmetros referentes à validação cruzada utilizada são o número de subconjuntos em que as imagens serão divididas, adotado como sete neste projeto, a opção de embaralhamento e, como no classificador, o estado randômico para o embaralhamento, no qual foi usado valor unitário.

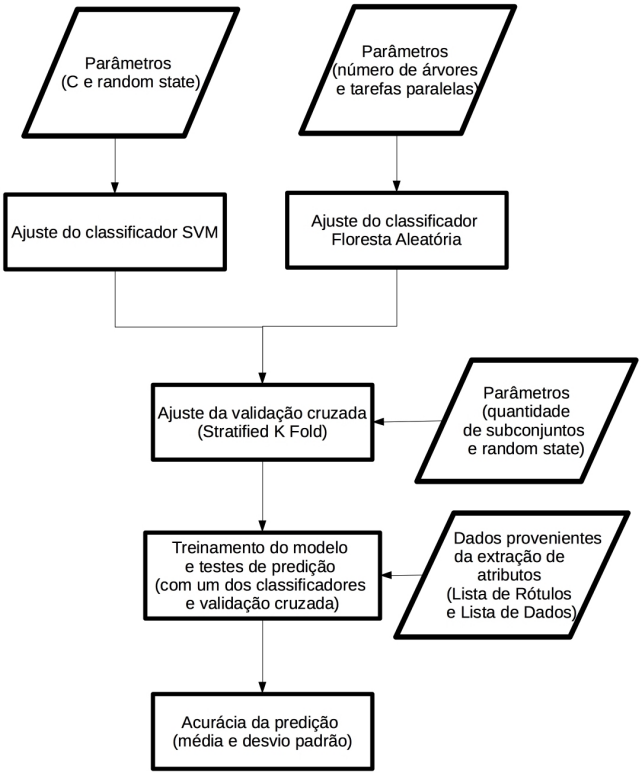

Figura 4.14: Fluxograma da segunda parte do treinamento do modelo classificador, do ajuste dos modelos classificadores ao resultado de acurácia das predições.

Em sequência, o classificador é aplicado no conjunto de imagens, o processo se dá com a utilização da validação cruzada previamente ajustada, nesse caso, o treinamento é realizado e, em seguida, o teste da predição. Baseada em todas as iterações da validação cruzada, a acurácia da classificação é calculada, apresentada como a média e o desvio padrão.

Todo o processo descrito nessa seção foi realizado em várias iterações, com a variação de muitos parâmetros que compõem os modelos para que os valores ótimos fossem encontrados. Todos seus resultados serão apresentados no capítulo seguinte.

5 RESULTADOS E DISCUSSÃO

Este capítulo é destinado à exposição dos resultados do sistema de classificação de tipografias, contendo a análise e discussão destes no caso da utilização dos dois diferentes modelos classificadores, SVM e Floresta Aleatória.

5.1 CLASSIFICAÇÃO DAS TIPOGRAFIAS

O sistema desenvolvido deve receber uma imagem e indicar a qual tipografia o caractere presente na imagem pertence, dentre aquelas escolhidas para compor o projeto. Sendo assim, as possíveis classes para as imagens são:

1. Adobe Jenson
2. Adobe Garamond
3. Baskerville
4. Didot
5. Clarendon Bold
6. Franklin Gothic URW Demi
7. Helvetica
8. Futura Book
9. Gill Sans

Como descrito no capítulo anterior, o resultado da classificação das tipografias foi mensurado a partir de testes de predição utilizando validação cruzada e computando o índice de acerto variadas vezes. Esse índice é calculado como a porcentagem de imagens que foram corretamente classificadas em relação ao total de imagens de entrada no estágio de teste. Além disso, o sistema foi sendo desenvolvido com o banco de imagens sendo incrementado de tipografia a tipografia e, em cada passo, realizada a validação cruzada e os testes de predição, bem como calculado o índice de acerto de classificação desse estágio. Todos esses dados são explicitados nas seções a seguir.

5.1.1 Utilizando Support Vector Machine

Primeiramente, utilizou-se para extração de atributos o modelo LBP e, para o classificador, o modelo linear SVM. Na tabela 5.1 pode-se verificar o resultado de índice de acerto de classificação obtido mediante a utilização de banco de imagens de diferentes versões. Além disso, o leitor pode acompanhar na Figura 5.1 o histórico do índice de acerto do sistema classificador utilizando os modelos citados. Vale ressaltar que, em algoritmo implementado anteriormente, as tipografias presentes em cada etapa eram adicionadas arbitrariamente no banco de imagens, o que poderia gerar alterações nos resultados finais. Sendo assim, implementou-se uma adição aleatória de cada tipografia no banco de imagens, repetindo o processo dez vezes e calculando sua média e desvio padrão.

Quantidade de tipografias no banco de imagens	2	3	4	5	6	7	8	9
Índice de acerto [%]	82,54	71,05	67,77	57,74	56,33	50,84	48,59	45,17
Desvio padrão [%]	23	23	14	9	9	4	5	0

Tabela 5.1: Resultados de índice de acerto da classificação em testes de validação cruzada do sistema classificador utilizando o modelo SVM linear, avaliado com banco de imagens em processo de incrementação tipografia à tipografia.

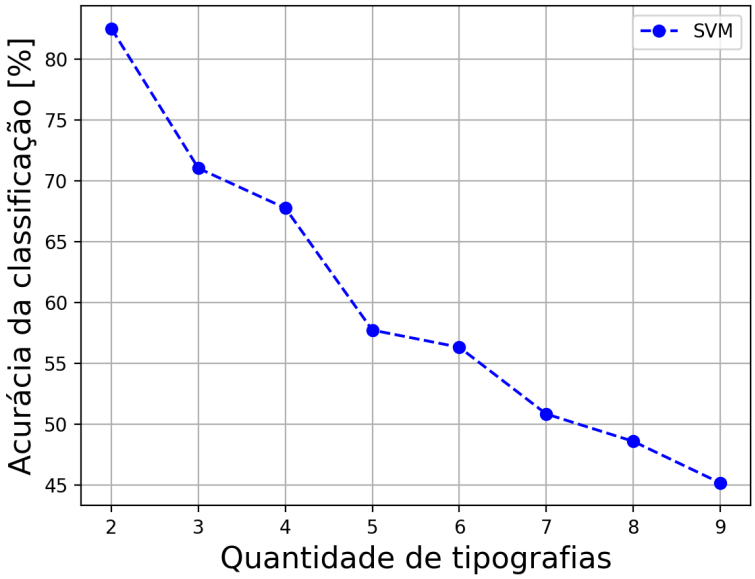

Figura 5.1: Gráfico do desempenho do sistema classificador com o modelo SVM em relação à quantidade de tipografias presentes no banco de imagens.

Posteriormente, foi utilizada a SVM com *kernel* RBF em lugar da versão linear. Também sendo aplicada em uma banco de imagens em processo de incrementação tipografia à tipografia. Os resultados foram superiores à versão linear e podem ser vistos na figura 5.2 e também na tabela 5.2.

Quantidade de tipografias no banco de imagens	2	3	4	5	6	7	8	9
Índice de acerto [%]	90,11	85,85	83,21	80,90	78,40	77,17	76,08	74,70
Desvio padrão [%]	8	4	4	2	4	4	2	0

Tabela 5.2: Resultados de índice de acerto da classificação em testes de validação cruzada do sistema classificador utilizando o modelo SVM com *kernel* RBF, avaliado com banco de imagens em processo de incrementação tipografia à tipografia.

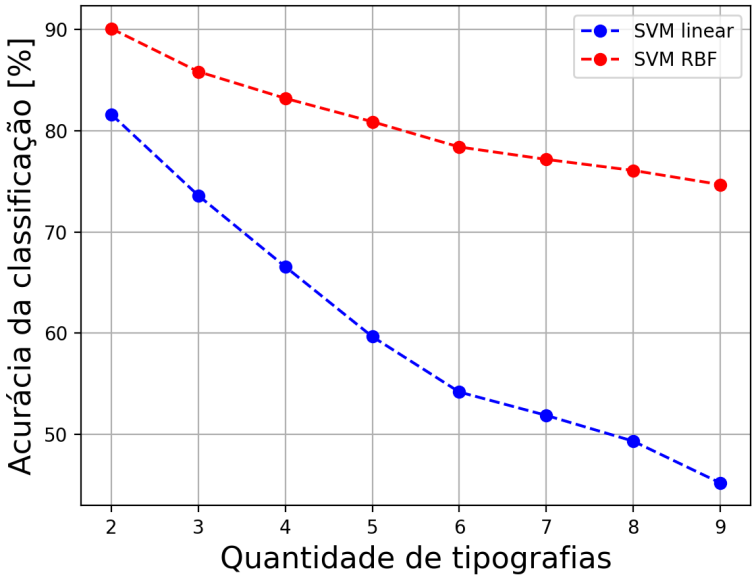

Figura 5.2: Gráfico do desempenho do sistema classificador com o modelo SVM linear e SVM com *kernel* RBF em relação à quantidade de tipografias presentes no banco de imagens.

Percebe-se que, a cada incremento no banco de imagens, o nível de acerto da classificação é comprometido, chegando a 74,70% com banco de imagens com as nove tipografias. Diante do índice de acerto de classificação baixo para esta aplicação e, além disso, em comparação com trabalhos similares com índices de 85% e 96,91%, decidiu-se aplicar outro modelo classificador [81] [82]. O resultado abaixo dos índices encontrados na literatura pode ser fruto de uma sensibilidade do modelo SVM quanto aos atributos (dados) que lhe são fornecidos durante a etapa de treinamento [4]. Além disso, como já comentado anteriormente, a classificação é dificultada diante do alto grau de similaridade entre algumas tipografias presentes no projeto.

A SVM com *kernel* RBF foi aquela que obteve melhor resultado apesar de vários treinamentos da máquina terem sido efetuados com outras opções de *kernel*, que podem ser vistos na Tabela 5.3. Implementações com os modelos Polinomial e Sigmoidal foram realizados somente no estágio final do banco de imagens, ou seja, com conjunto de nove tipografias.

Vale notar que a SVM linear obtém bom desempenho como classificador em conjuntos de dados com distribuição linear, ou que se aproximem desse padrão. Já no caso da utilização de *kernel* não-linear, o conjunto de treinamento é mapeado para um espaço com dimensão superior

e que seja mais suscetível à uma separação linear das classes [4].

Kernel	Índice de Acerto [%]	Desvio Padrão [%]
Linear	45,17	0
RBF	74,70	0
Polinomial	54,19	4
Sigmoidal	35,61	5

Tabela 5.3: Índices de acerto da classificação em testes de validação cruzada do sistema com o modelo SVM avaliado com diferentes *kernels* em banco de imagens com nove tipografias.

No entanto, os dois casos possuem um grau de dependência da distribuição do conjunto original de dados para que apresentem bom desempenho na classificação. Além disso, como pode-se perceber por sua característica linear intrínseca, o modelo SVM foi concebido como um classificador binário e, posteriormente, adaptado para conjuntos não-lineares em casos de multiclassificação [65]. Dessa forma, a utilização da SVM nesse caso depende de uma série de parâmetros que devem ser bem ajustados para que ocorra a separação linear de forma ótima, o que dificulta sua utilização, fator que pode ter sido decisivo em seu desempenho neste projeto.

Portanto, para continuar empregando o classificador SVM, um modelo mais robusto de extração de atributos deveria ser utilizado, o que seria também uma opção para implementação do sistema. Porém, optou-se por manter o LBP para extração de atributos devido ao seu baixo custo computacional e rapidez de execução, critério importante no cenário de grande volume de dados e também por ser uma aplicação interativa, em se tratando de um produto final.

É ainda importante enfatizar a influência do conjunto de dados composto para o treinamento e testes no caso de aprendizado de máquina. Como descrito no capítulo anterior, o primeiro banco de imagens criado para treinamento e testes do sistema classificador comprometeu severamente o índice de acerto da classificação realizada pelo sistema. A Tabela 5.4 apresenta uma comparação entre os resultados obtidos com o banco de imagens precedente e o banco de imagens após maior refinamento. Vale ressaltar que o banco de imagens anterior foi desenvolvido apenas até conter quatro tipografias. Além disso, a SVM Linear foi o único modelo utilizado nessa avaliação.

Quantidade de tipografias no banco de imagens	2	3	4
Helvetica	X	X	X
Garamond	X	X	X
Clarendon		X	X
Futura			X
Índice de Acerto no Banco Antigo[%]	91,95	76,83	59,38
Desvio padrão [%]	2	2	2
Índice de Acerto no Banco Melhorado[%]	94,2	83,16	64,47
Desvio padrão [%]	2	5	4

Tabela 5.4: Resultados comparativos de índice de acerto da classificação em testes de validação cruzada do sistema classificador utilizando o modelo SVM linear, avaliado com banco de imagens antes e depois do refinamento, em processo de incrementação tipografia à tipografia, até quatro tipografias.

5.1.2 Utilizando Floresta Aleatória

O segundo classificador utilizado foi o Floresta Aleatória, ainda com o modelo LBP para extração de atributos das imagens. Assim como na primeira versão do sistema, o modelo classificador foi treinado com oito versões do banco de imagens. A primeira versão é composta de duas tipografias que são suficientemente distintas entre si e então, a cada versão, o banco de imagens é incrementado com um conjunto de imagens de uma das tipografia presentes no projeto. Os resultados de índice de acerto da classificação performada pelo sistema utilizando Floresta Aleatória como classificador são expostos na Tabela 5.5. Na Figura 5.3, pode-se ver a comparação dos resultados do classificador SVM, com *kernel* linear e RBF, e do classificador Floresta Aleatória a medida em que o banco de imagens foi sendo incrementado.

Quantidade de tipografias no banco de imagens	2	3	4	5	6	7	8	9
Índice de acerto [%]	95,78	93,40	92,74	88,86	88,79	86,47	86,24	84,91
Desvio padrão [%]	9	7	3	3	5	2	2	0

Tabela 5.5: Resultados de índice de acerto da classificação em testes de validação cruzada do sistema com o modelo Floresta Aleatória, avaliado com banco de imagens em processo de incrementação tipografia à tipografia.

Sendo assim, o resultado final obtido para a classificação das nove tipografias apresentou um índice de acerto de 84,91%. O desempenho encontrado foi superior ao caso anterior, elevando o índice de acerto do sistema classificador. Apesar de ainda ser passível de melhoria, o desempenho foi similar ao encontrado em trabalho semelhante, que apresentou índice de acerto de 85%, o que pode ser considerado adequado para um primeiro estágio no momento [81].

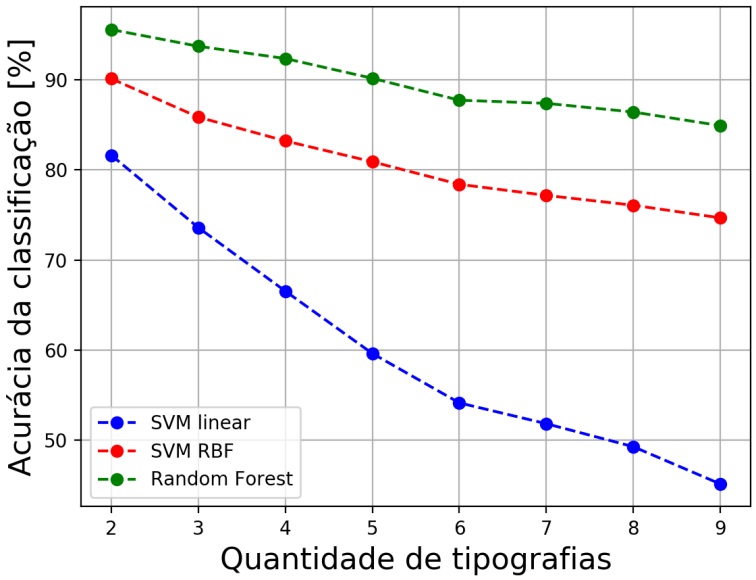

Figura 5.3: Gráfico do desempenho do sistema classificador com os dois modelos utilizados, SVM e Floresta Aleatória, em relação à quantidade de tipografias presentes no banco de imagens.

Nota-se a grande discrepância entre o desempenho do sistema empregando o classificador Floresta Aleatória e a SVM. Esse resultado pode ser derivado de vários aspectos, entre eles, o perfil do conjunto de dados necessário para garantir um bom funcionamento da SVM como classificador.

Em relação ao modelo classificador Floresta Aleatória, os motivos pelos quais sua aplicação resultou em um melhor desempenho podem estar relacionados à robustez do modelo em relação a ruídos, bem como à distribuição e características gerais do conjunto de dados de entrada na máquina [68]. Além disso, por se tratar de um modelo construído por árvores de decisão (*decision trees*), possui uma capacidade ampliada para lidar com espaços de variadas dimensões e também com um número alto de amostras de treinamento. Sendo assim, o modelo Floresta Aleatória pode ser considerado como um modelo que é intrinsicamente ajustado para multi-classificação, fato que facilita a sua utilização no caso aqui apresentado.

6 CONCLUSÃO

Dado que a proposta deste trabalho é desenvolver um sistema de reconhecimento de padrões em tipos para aplicação no projeto de Tipografia Tátil, que tem como objetivo auxiliar o ensino de tipografia a deficientes visuais, nesta monografia foi implementado um sistema capaz de reconhecer a qual tipografia pertence o caractere apresentado em imagem, classificando-o de acordo com as tipografias escolhidas no projeto.

A criação do banco de imagens é uma contribuição para outros projetos de mesma área, já que é uma etapa que, no geral, demanda uma grande quantidade de tempo no processo de desenvolvimento de um projeto de Aprendizado de Máquina. Para o acesso ao banco de imagens completo, tem-se um *link* apresentado no capítulo anterior, bem como o de todos algoritmos desenvolvidos.

Em relação ao produto de tecnologia assistiva como um todo, testes foram feitos com deficientes visuais para ajustes no desenvolvimento das peças táteis. No entanto, em relação ao sistema computacional interativo, foram feitas apenas entrevistas com alguns deficientes visuais. Após o desenvolvimento de mais partes do sistema computacional, é necessário que sejam feitos mais testes para que ajustes possam ser realizados no sistema, de forma a torná-lo de fácil utilização para o usuário.

Como trabalhos futuros, deve-se buscar melhorar o índice de acerto do sistema classificador, de forma a atingir um nível próximo ou superior ao obtido por outros autores [40]. Para tal, pretende-se tentar uma abordagem parecida com a usada no trabalho citado, criando, para cada letra, um sistema classificador distinto para realizar a OFR.

O descritor de imagem utilizado, LBP, foi suficiente para a descrição das imagens para esta aplicação. Porém, outra possibilidade é utilizar outros descritores de imagem para a extração de atributos, ou uma combinação, como, por exemplo, um combinação do LBP e HOG (*Histogram of Oriented Gradient*), que apresenta melhor desempenho em variados casos [83] [84]. Ainda, o modelo classificador Floresta Aleatória obteve um índice de acerto de classificação suficiente para primeira versão do sistema, porém pode-se escolher um novo modelo classificador para a implementação para a fase final do trabalho. Por último, pretende-se alimentar o conjunto de dados de treinamento com mais exemplos, o que pode resultar em um melhor desempenho do sistema [52].

Finalmente, para que o sistema esteja completo é necessário implementar o OCR para que o caractere da peça seja reconhecido. É necessário também que todas as funcionalidades do sistema sejam agrupadas. Sendo assim, uma fase de integração dos algoritmos com uma interface para o usuário é necessária. Este passo vai permitir integrar o sistema classificador com a síntese de voz, que vai fornecer os comandos para guiar o usuário e informá-lo segundo material didático já desenvolvido para o projeto [2].

REFERÊNCIAS BIBLIOGRÁFICAS

1 GRAHAM, B. *Classic Posters - Bill Graham Series*. 2017. Disponível em: <https://www.classicposters.com/Bill_Graham>.

2 CRUZ, L. *Tipografia Tátil para Deficientes Visuais*. Tese (Bacharelado em Design) — Universidade de Brasília, 2017.

3 MIRANDA, R. A. R.; SILVA, F. A. da; ARTERO, A. O.; PITERI, M. A. Handwritten character recognition based on frequency, character-edge distances and densities. In: *Proceedings of the IX Workshop de Viso Computacional*. [S.l.: s.n.], 2013.

4 LORENA, A. C.; CARVALHO, A. C. de. Uma introdução às support vector machines. *Revista de Informática Teórica e Aplicada*, v. 14, n. 2, p. 43–67, 2007.

5 AMARAL, V. do; THOMAZ, C. E. Extraçao e comparaçao de caracterısticas locais e globais para o reconhecimento automático de imagens de faces. In: *VIII Workshop de Visao Computacional*. [S.l.: s.n.], 2012.

6 ZHAO, X.; ZHANG, S. Facial expression recognition based on local binary patterns and kernel discriminant isomap. *Sensors*, Molecular Diversity Preservation International, v. 11, n. 10, p. 9573–9588, 2011.

7 MARTIN, C. H. *Kernels Part 1: What is an RBF Kernel? Really?* 2012. Disponível em: <https://calculatedcontent.com/2012/02/06/kernels_part_1/>.

8 BRASIL, S. F. *Estatuto da Pessoa com Deficiência*. 2015. Disponível em: <http://www.planalto.gov.br/ccivil{_}03/{_}Ato2015-2018/2015/Lei/L131>.

9 BERSCH, R. Tecnologia Assistiva. 2013.

10 RADABAUGH, M. Nidrr's long range plan-technology for access and function research section two: Niddr research agenda chapter 5: Technology for access and function. *United States*, 1993.

11 LÉVY, P.; COSTA, C. I. da. *tecnologias da inteligência, As*. [S.l.]: Editora 34, 1993.

12 NETO, A. Do braille às tecnologias digitais de informação e comunicação: leituras e vivências de cidadãos-cegos, suas relações com a informação e com a construção de conhecimento. 2006.

13 BECK-WINCHATZ, B.; RICCOBONO, M. A. Advancing participation of blind students in Science, Technology, Engineering, and Math. *Advances in Space Research*, v. 42, n. 11, p. 1855–1858, 2008. ISSN 02731177.

14 PLIMMER, B.; CROSSAN, A.; BREWSTER, S. a.; BLAGOJEVIC, R. Multimodal collaborative handwriting training for visually-impaired people. *Proceeding of the twenty-sixth annual CHI conference on Human factors in computing systems - CHI '08*, p. 393, 2008. Disponível em: <http://portal.acm.org/citation.cfm?doid=1357054.1357119>.

15 BALLESTERO-ALVAREZ, J. A. Multissensorialidade no ensino de desenho a cegos. p. 1–121, 2003. Disponível em: <http://www.diaadiaeducacao.pr.gov.br/diaadia/diadia/arquivos/File/conteudo/artigos{_}teses/2010/Arte/dissertacao/ens{_}desen>.

16 SOLER, M.-A. *Didáctica multisensorial de las ciencias: Un nuevo método para alumnos ciegos, deficientes visuales, y también sin problemas de visión*. [S.l.]: Grupo Planeta (GBS), 1999. v. 40.

17 FERNANDA, D.; MORAIS, P. D. Acessibilidade da arte ao público deficiente visual. p. 201–212, 2007.

18 BORGES, J. A. Dosvox: uma nova realidade educacional para deficientes visuais. *Revista Benjamim Constant*, 1998.

19 AMORIM, E. S. M. d. S.; CARVALHO, d. J. L.; MENEZES, L. K. B. Educação de cegos mediada pela tecnologia. *Secretaria de Educação de Salvador. Salvador*, p. 1–12, 2009. Disponível em: <http://www.educacao.salvador.ba.gov.br/Site/documentos/espaco-virtual/espaco-autorias/artigos/educacaodecegosmediadapelastecnologias.pdf>.

20 FREEDOMSCIENTIFIC. *Blindness Solutions: JAWS*. 2016. Disponível em: <http://www.freedomscientific.com/Products/Blindness/JAWS>.

21 MICROPOWER. *Virtual Vision: Acessibilidade para pessoas com deficiência visual*. 2016. Disponível em: <http://www.virtualvision.com.br/>.

22 FREEDOMSCIENTIFIC. *Low Vision Solutions: OpenBook and PEARL*. 2016. Disponível em: <http://www.freedomscientific.com/Products/LowVision/OpenBook>.

23 SANT'ANNA, L. *O que é um Display Braille?* 2008. Disponível em: <http://www.acessibilidadelegal.com/33-display-braille.php>.

24 INDEXBRAILLE. *TactileView - Tactile Graphics*. 2016. Disponível em: <http://www.indexbraille.com/en-us/support/braille-editors/tactileview-tactile-graphics>.

25 ARDITI, A. Adjustable typography: an approach to enhancing low vision text accessibility. *Ergonomics*, v. 47, n. 5, p. 469–482, 2004. ISSN 0014-0139.

26 VELASCO, C.; WOODS, A. T.; HYNDMAN, S.; SPENCE, C. The taste of typeface. *i-Perception*, v. 6, n. 4, p. 1–10, 2015. ISSN 20416695.

27 COSTA, H. M. P. d. O protestantismo e a palavra impressa: ensaios introdutórios. *Ciências da Religião - História e Sociedade*, v. 6, n. 2, p. 124 – 135, 2008.

28 CRUZ, L. E.; MAYNARDES, A. C. Tipografia tátil Tactile typography. *XX Congreso de la Sociedad Iberoamericana de Gráfica Digital*, v. 3, n. 1, p. 360–365, 2016.

29 WORLDSTANDARDS. *The World's Scripts and alphabets*. 2015. Disponível em: <http://www.worldstandards.eu/other/alphabets/>.

30 BISHOP, C. M. Pattern recognition and machine learning. v. 128, p. 1–3, 2006.

31 SAMUEL, A. L. Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, v. 3, n. 3, p. 210–229, 1959.

32 NG, A. *Machine Learning: Introduction*. 2016.

33 LIBRALÃO, G. L.; OSHIRO, R. M.; NETTO, A. V.; CARVALHO, A.; OLIVEIRA, M. Técnicas de aprendizado de máquina para análise de imagens oftalmológicas. *São Paulo. Universidade de São Paulo*, 2003.

34 MIRANDA, R. A. R.; SILVA, F. A. da; PAZOTI, M. A.; ARTERO, A. O.; PITERI, M. A. Algoritmo Para O Reconhecimento De Caracteres Manuscritos. *Colloquium Exactarum*, v. 5, n. 2, p. 109–127, 2013. ISSN 21788332. Disponível em: <http://revistas.unoeste.br/revistas/ojs/index.php/ce/article/view/943/998>.

35 MAINDONALD, J. Parametric models for discrimination & classification. 2007.

36 FERIS, R. S. Representation learning beyond human labels: Practical Applications in Visual Analysis of People. 2016.

37 GRIMMER, J. We are all social scientists now: how big data, machine learning, and causal inference work together. *PS: Political Science & Politics*, Cambridge Univ Press, v. 48, n. 01, p. 80–83, 2015.

38 LI, A.; LIU, L.; WANG, K.; LIU, S.; YAN, S. Clothing attributes assisted person reidentification. *IEEE Transactions on Circuits and Systems for Video Technology*, IEEE, v. 25, n. 5, p. 869–878, 2015.

39 SZELISKI, R. *Computer vision: algorithms and applications*. [S.l.]: Springer Science & Business Media, 2010. 575–621 p.

40 ZRAMDINI, A. *Study of optical font recognition based on global typographical features*. Tese (Doutorado) — University of Fribourg (Switzerland, 1995.

41 SHI, H.; BROOK, S.; BROOK, S. Font Recognition and Contextual Processing for More Accurate Text Recognition. p. 39–44, 1997.

42 MANNA, S. L.; COLIA, A.; SPERDUTI, A. Optical font recognition for multi-font OCR and document processing. *Proceedings of the Tenth International Workshop on Database and Expert Systems Applications*, p. 549 – 553, 1999.

43 YANG, Z.; YANG, L.; QI, D.; SUEN, C. Y. An EMD-based recognition method for Chinese fonts and styles. *Pattern Recognition Letters*, v. 27, n. 14, p. 1692–1701, 2006. ISSN 01678655.

44 SLIMANE, F.; KANOUN, S.; HENNEBERT, J.; ALIMI, A. M.; INGOLD, R. A study on font-family and font-size recognition applied to Arabic word images at ultra-low resolution. *Pattern Recognition Letters*, Elsevier B.V., v. 34, n. 2, p. 209–218, 2013. ISSN 01678655. Disponível em: <http://dx.doi.org/10.1016/j.patrec.2012.09.012>.

45 ZAHEDI, M.; ESLAMI, S. Farsi/Arabic optical font recognition using SIFT features. *Procedia Computer Science*, Elsevier, v. 3, p. 1055–1059, 2011. ISSN 18770509. Disponível em: <http://dx.doi.org/10.1016/j.procs.2010.12.173>.

46 OJALA, T.; PIETIKAINEN, M.; HARWOOD, D. Performance evaluation of texture measures with classification based on kullback discrimination of distributions. In: IEEE. *Pattern Recognition, 1994. Vol. 1-Conference A: Computer Vision & Image Processing., Proceedings of the 12th IAPR International Conference on*. [S.l.], 1994. v. 1, p. 582–585.

47 BIERUT, M.; HELFAND, J.; POYNOR, R. Textos clássicos do design gráfico. *São Paulo: WMF Martins*, 2010.

48 BRINGHURST, R. *Elementos do estilo tipográfico*. [S.l.]: Editora Cosac Naify, 2005.

49 LUPTON, E. *Thinking with type: A critical guide for designers, writers, editors, & students*. [S.l.]: Chronicle Books, 2014.

50 KANE, J. Manual dos tipos. *Barcelona: G. Gili*, 2012.

51 ROCHA, C. Novo projeto tipográfico. *São Paulo: Rosari*, 2012.

52 PEROTTO, F.; ÁLVARES, L. *Aprendizagem de Máquina*. 2005.

53 KOHAVI, R. et al. A study of cross-validation and bootstrap for accuracy estimation and model selection. In: STANFORD, CA. *Ijcai*. [S.l.], 1995. v. 14, n. 2, p. 1137–1145.

54 GONZALEZ, J. A. *Sistemas Inteligentes: Classificação e Regras de Classificação*. 2013.

55 SCHNEIDER, J. *Cross Validation*. 1997. Disponível em: <https://www.cs.cmu.edu/~schneide/tut5/node42.html>.

56 MUSCI, M. et al. *Padrões Binários Locais na Classificação de Imagens de Sensoriamento Remoto In: XV Simpósio Brasileiro de Sensoriamento Remoto, 2011, Curitiba*. [S.l.]: Anais.

57 PRINCE, S. J. *Computer vision*. 1. ed. [S.l.]: Cambridge UNi. Press, 2012.

58 GONZALEZ, R. C.; WOODS, R. E. *Digital image processing*. 3. ed. [S.l.]: Dorling Kindersley, 2014.

59 MITCHELL, T. M. *Machine learning*. [S.l.]: MacGraw-Hill, 1997.

60 NOBLE, W. S. et al. Support vector machine applications in computational biology. *Kernel methods in computational biology*, Cambridge, p. 71–92, 2004.

61 KIM, K. I.; JUNG, K.; PARK, S. H.; KIM, H. J. Support vector machines for texture classification. *IEEE transactions on pattern analysis and machine intelligence*, IEEE, v. 24, n. 11, p. 1542–1550, 2002.

62 BEN-HUR, A.; HORN, D.; SIEGELMANN, H. T.; VAPNIK, V. Support vector clustering. *Journal of machine learning research*, v. 2, n. Dec, p. 125–137, 2001.

63 ROSEBROCK, A. *An intro to linear classification with Python - PyImageSearch*. 2017. Disponível em: <http://www.pyimagesearch.com/2016/08/22/an-intro-to-linear-classification-with-python/>.

64 SHAWE-TAYLOR, J.; CRISTIANINI, N. *Kernel methods for pattern analysis*. [S.l.]: Cambridge university press, 2004.

65 BOSER, B. E.; GUYON, I. M.; VAPNIK, V. N. A training algorithm for optimal margin classifiers. In: ACM. *Proceedings of the fifth annual workshop on Computational learning theory*. [S.l.], 1992. p. 144–152.

66 COVER, T. M. Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. *IEEE transactions on electronic computers*, IEEE, n. 3, p. 326–334, 1965.

67 PAL, M. Multiclass approaches for support vector machine based land cover classification. *arXiv preprint arXiv:0802.2411*, 2008.

68 BREIMAN, L. Random forests. *Machine learning*, Springer, v. 45, n. 1, p. 5–32, 2001.

69 GUEDES, A. R. M.; GUIMARÃES, V. L. Sistema de identificação de íris utilizando local binary pattern e random forest.

70 CARUANA, R.; KARAMPATZIAKIS, N.; YESSENALINA, A. An empirical evaluation of supervised learning in high dimensions. In: ACM. *Proceedings of the 25th international conference on Machine learning*. [S.l.], 2008. p. 96–103.

71 NETO, C. *Potencial de técnicas de mineração de dados para o mapeamento de áreas cafeeiras*. [s.n.], 2014. 14-16 p. Disponível em: <http://wiki.dpi.inpe.br/lib/exe/fetch.php?media=ser300:alunos2014:cesare_monografia.pdf>.

72 BOSCH, A.; ZISSERMAN, A.; MUNOZ, X. Image classification using random forests and ferns. In: IEEE. *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*. [S.l.], 2007. p. 1–8.

73 USE, F. in. *Fonts in Use – type at work in the real world*. 2017. Disponível em: <http://fontsinuse.com>.

74 ROCHA, C. *Tipografia comparada*. 1. ed. [S.l.]: Rosari, 2004.

75 TEAM, O. *OpenCV library - About*. 2017. Disponível em: <http://opencv.org/about.html>.

76 WALT, S. van der; SCHÖNBERGER, J. L.; Nunez-Iglesias, J.; BOULOGNE, F.; WARNER, J. D.; YAGER, N.; GOUILLART, E.; YU, T.; CONTRIBUTORS the scikit-image. scikit-image: image processing in Python. *PeerJ*, v. 2, p. e453, 6 2014. ISSN 2167-8359. Disponível em: <http://dx.doi.org/10.7717/peerj.453>.

77 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.

78 DEVELOPERS, S. *Scientific Computing Tools for Python — SciPy.org*. 2017. Disponível em: <https://www.scipy.org/about.html>.

79 NUMPY, D. *NumPy — NumPy*. 2017. Disponível em: <http://www.numpy.org/>.

80 ROSEBROCK, A. *Local Binary Patterns with Python and OpenCV*. 2015. Disponível em: <http://www.pyimagesearch.com/2015/12/07/local-binary-patterns-with-python-opencv/>.

81 MANNA, S. L.; COLIA, A.; SPERDUTI, A. Optical font recognition for multi-font ocr and document processing. In: IEEE. *Database and Expert Systems Applications, 1999. Proceedings. Tenth International Workshop on*. [S.l.], 1999. p. 549–553.

82 ZRAMDINI, A.; INGOLD, R. Optical font recognition using typographical features. *IEEE Transactions on pattern analysis and machine intelligence*, IEEE, v. 20, n. 8, p. 877–882, 1998.

83 ZHANG, J.; HUANG, K.; YU, Y.; TAN, T. Boosted local structured hog-lbp for object localization. In: IEEE. *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. [S.l.], 2011. p. 1393–1400.

84 WANG, X.; HAN, T. X.; YAN, S. An hog-lbp human detector with partial occlusion handling. In: IEEE. *Computer Vision, 2009 IEEE 12th International Conference on*. [S.l.], 2009. p. 32–39.