UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO

Lisandra Sousa da Cruz

Comparação de algoritmos de reconhecimento de gestos aplicados à sinais estáticos de Libras

Recife

2019

Lisandra Sousa da Cruz

Comparação de algoritmos de reconhecimento de gestos aplicados à sinais estáticos de Libras

Monografia apresentada ao Curso de Bacharelado em Sistemas de Informação da Universidade Federal Rural de Pernambuco, como requisito parcial para obtenção do título de Bacharel em Sistemas de Informação.

Universidade Federal Rural de Pernambuco – UFRPE

Departamento de Estatística e Informática

Curso de Bacharelado em Sistemas de Informação

Orientador: Filipe Cordeiro

Coorientador: Valmir Macario Filho

Recife

2019

À todos que me apoiaram e contribuíram para eu chegar até aqui.

Agradecimentos

Aos meus pais, Silvia e Josué, por serem esses pais sensacionais que desde sempre priorizaram minha educação e lutaram por isso diante de todas as dificuldades, parafraseando mainha: ”é pra estudar? a gente vai dar um jeito”, dedico todo o meu sucesso a vocês.

A minha ruralinda (UFRPE) e a BSI por serem minha segunda casa durante anos, onde aprendi muito e conquistei maravilhosos amigos, em especial os pruprus e ruralindxs, vocês são demais. A secretária mais desenrolada da UFRPE, Carol sem você BSI não seria o mesmo. A todos os professores que passaram pela minha trajetória, em especial Roberta Macêdo, Ceça Moraes e Gabriel Alves, vocês me inspiraram a ser quem sou hoje, uma educadora que acredita na equidade, conhecimento e empatia associados a educação com forma de mudar o meio. Filipe Cordeiro e Walmir Macário, vocês foram uma dupla sensacional, obrigada por me orientarem nessa nesse trabalho.

Dedico também a todas as mulheres que no passado lutaram para que hoje esse direito me fosse garantido e as que hoje continuam a lutar por isso, computação é pra mulher sim, porque lugar de mulher é onde ela quiser. Um agradecimento especial a Rai e Gabs, pois tenho certeza que sermos alicerces uma das outras foi parte fundamental para que concluíssemos.

Ao meu parceiro de bugs cerebrais Demis, por ter se tornado meu melhor amigo, namorado e companheiro de aventuras (seja para fazer rapel ou inovar na reta final do curso). Mais uma vez obrigada por ser esse cara sensacional, que a todos os dias me lembrou onde eu já tinha chegado e não me deixou desistir, sem os seus post-its motivacionais talvez isso não fosse possível.

Por fim, dedico esse passo importante da minha vida àqueles que mesmo mais distantes acreditaram no meu potencial e estiveram sempre torcendo por mim, me apoiando e incentivando para superar todos percalços e a toda a comunidade surda, em especial a meu irmão Lucas, que trava batalhas diárias frente a falsa inclusão existente hoje.

“Eu quero ser tudo que sou capaz de me tornar.”
(Katherine Mansfield)

Resumo

A Língua Brasileira de Sinais (Libras) foi criada a fim de suprir uma necessidade de comunicação não-verbal para os surdos, que durante muito tempo foram doutrinados à ter o português como sua primeira língua. Atualmente, a Libras é a segunda língua oficial do Brasil e primeira língua dos surdos, assim como o português é para o ouvinte. Entretanto, mesmo com tamanho reconhecimento, a segunda língua oficial do Brasil não é conhecida pela maior parte da população brasileira. O processo de inclusão visa proporcionar igualdade aos deficientes, de forma que a deficiência não seja um fator impeditivo à convivência em sociedade. Com o advento da tecnologia e avanços da Inteligência Artificial (IA), foram criados artifícios tecnológicos visando propiciar inclusão. Na IA, o reconhecimento de padrões é um dos subtemas mais abordados na atualidade, sendo bastante aplicada para a classificação de gestos de diversas línguas de sinais na literatura. Essa pesquisa tem como principal tarefa identificar as mãos que formam um determinado sinal de Libras e em seguida reconhecer a que classe pertence, classificando-o. Baseado na classificação da Língua de Sinais Americana, a *Feature Fusion-based Convolutional Neural Network* (FFCNN), uma rede estendida da *Convolutional Neural Network* (CNN), obteve a melhor acurácia em comparação a outras redes, dentre elas a *Visual Geometry Group* (VGG). Diante desse cenário, esse trabalho aplica a FFCNN à gestos estáticos de Libras a fim de verificar se a FFCNN obtém a melhor acurácia assim como obteve na Língua de Sinais Americana. Para alcançar esse objetivo são comparados três classificadores: VGG com uma variação da CNN com 13 e 16 camadas; FFCNN e uma rede *Multi Layer Perceptron* (MLP) usada no reconhecimento de gestos estáticos de Libras na literatura. Os algoritmos foram aplicados em um *dataset* de Libras que contém 9.600 imagens de 40 sinais. Os resultados demonstram que a rede VGG com 16 camadas obteve a maior acurácia dentre modelos descritos neste trabalho, com valor de 99,45%.

Palavras-chave: Libras, Inteligencia Artificial, Visão Computacional, CNN, VGG, MLP.

Abstract

Brazilian Sign Language (BSL) has been created in order to cope with a necessity of a non-verbal communication for the deafs, which during a long time were indoctrinated to learn the Brazilian Portuguese as their first language. Nowadays, the BSL is the Brazil's second official language and first deaf's language, as well as the Portuguese for the listener. Nevertheless, even with large recognition, the Brazil's second official language is not known by the majority of the Brazilian population. The inclusion process aims to allow equality for the impaired, such that the deficiency does not become an impediment factor for living together in society. With the technology arrival and the Artificial Inteligence (AI) advances, it was created technologic artifices to allow inclusion. In the AI, the pattern recognition is one of more approached subthemes in the present, and it is widely applied for the gesture classification of many sign languages in literature. This research has, as key task, the identification of the hands that form a certain BSL gesture and, thus, the recognition of the class it belongs to. Based on American Sign Language (ASL) classification, the Feature Fusion-based Convolutional Neural Network (FFCNN), an extended network from Convolutional Neural Network (CNN), obtained the best accuracy in comparison to other networks, such as Visual Geometry Group (VGG). Therefore, based on this scenario, this work applies the FFCNN to BSL static gestures to verify whether the FFCNN obtain the best accuracy as well as obtained in ASL or not. In order to achieve the goal, this work compares three classifiers: the Visual Geometry Group (VGG), a CNN with variation of 13 and 16 layers, the FFCNN, and a Multi Layer Perceptron network used in recognition of BSL static gestures in literature. The algorithms were applied in a BSL dataset with 9,600 images of 40 signals. The results demonstrate that VGG with 16 layers obtained the best accuracy regarding the described models in this work, corresponding to 99,45%.

Keywords: Brazilian Sign Language, Artificial Intelligence, Computer Vision, CNN, VGG, MLP.

Lista de ilustrações

Lista de tabelas

Lista de Siglas

ASID Ação Social para Igualdade das Diferenças

ASL *American Sign Language*

ASL *American Sign Language*

CNN *Convolutional Neural Network*

FFCNN *Feature Fusion-based Convolutional Neural Network*

FN Falso Negativo

FP Falso Positivo

HOG Histograma de Gradientes Orientados

IA Inteligência Artificial

IBGE Instituto de Geografia e Estatística

ILSVRC *ImageNet Large Scale Visual Recognition Challenge*

Libras Língua Brasileira de Sinais

LS Línguas de Sinais

LSA Língua de Sinais Argentina

MIZ Momentos Invariantes de Zernike

MLP *Multi-Layer Perceptron*

OMS Organização Mundial de Saúde

PcD Pessoa com Deficiência

ReLU *Rectifier Linear Unit*

SMS *Short Message Service*

TDD *Telecommunications Device for the Deaf*

VGG *Visual Geometry Group*

VN Verdadeiro Negativo

VP Verdadeiro Positivo

Sumário

1 Introdução

A Organização Mundial de Saúde (OMS) visa dentro do sistema das Nações Unidas coordenar e dirigir a saúde internacional. Segundo a OMS, cerca de 1 bilhão de pessoas vivem com pelo menos um tipo de deficiência: visual, auditiva, motora ou mental (OMS, 2019). No último censo realizado pelo Instituto de Geografia e Estatística (IBGE), foram contabilizadas cerca de 45,6 milhões de brasileiros que declararam algum tipo de deficiência (LOSCHI, 2019), dentre esses 5,1% alegaram ter alguma perda auditiva (IBGE, 2012).

Assim como os ouvintes que nascem no Brasil e tem sua língua materna o português falado, os surdos brasileiros têm como sua primeira língua a Libras (Língua Brasileira de Sinais). Em décadas passadas, a maioria dos surdos eram escondidos por seus familiares por serem consideradas pessoas fora do padrão da sociedade e sua forma de comunicação através de sinais era malvista por ela. Consequentemente, os surdos sentiam-se isolados e sozinhos, acarretando em problemas psicocomportamentais e dificuldade no desenvolvimento cognitivo (MONTEIRO, 2006). Com o passar do tempo, esses gestos tornaram-se uma linguagem e em 24 de abril de 2002, através da lei nº 10.436, a Libras tornou-se a segunda língua oficial do Brasil [1].

O panorama da inclusão de deficientes no Brasil, segundo a Ação Social para Igualdade das Diferenças (ASID) , pode ser lembrado por duas vertentes: escolar e profissional [2]. No âmbito escolar, a inclusão de Pessoas com Deficiência (PcD) vêm sendo assegurada pela Lei Brasileira de Inclusão da Pessoa com Deficiência - n°13.146/2015 (CIVIL, 2015), que visa promover condições de igualdade com o intuito de garantir a inclusão social, objetivando também atender ao desafio da inclusão trazido pela Meta 4 do Plano Nacional de Educação (MEC, 2019). No contexto profissional, há a Lei de Cotas - n° 8.213/1991 (CI2017VIL, 1991), a qual estabelece cotas compulsórias de vagas de empregos - entre 2% e 5%, que devem ser respeitadas pelas empresas privadas com mais de cem empregados.

Com todo esse amparo legal à PcD, é esperado que o deficiente seja capaz de conviver melhor em sociedade e de forma mais inclusiva, onde a deficiência não seja um agente causador de exclusão, sendo assim uma conquista para a população portadora. No entanto, na prática não há garantia de um real atendimento especializado às crianças que precisam, visto que, os profissionais não tem formação especializada como afirma a pesquisa realizada por (OLIVEIRA et al., 2012) em que 70% dos professores entrevistados afirmam não terem recebido preparo para alunos especiais durante

[1] https://inclusaoja.com.br/legislacao/
[2] https://asidbrasil.org.br/panorama-da-inclusao-no-brasil

sua formação.

A fim de propiciar inclusão aos surdos, alternativas inclusivas foram criadas e dentre elas temos o *TeleTypewriter* também conhecido por *Telecommunications Device for the Deaf* (TDD) . O TDD surgiu por volta de 1960 objetivando que surdos pudessem se comunicar através da telefonia fixa existente na época. A comunicação é feita através de mensagens de texto e faz-se necessário quatro agentes para que a chamada possa ser realizada: o surdo/pessoa que deseja ligar, um aparelho TDD, uma linha telefônica e um interlocutor. Para dar prosseguimento à chamada é preciso acoplar o telefone fixo no TDD, como representado na Figura 1, e discar para uma central, onde o interlocutor atende e conversa com o surdo através de texto. Simultaneamente o interlocutor liga para o número solicitado e age como um intermediador na conversa ouvindo e transpondo às mensagens.

Figura 1 – *Telecommunications Device for the Deaf* (BERKE, 2018)

Alguns anos depois surgiu os celulares e com ele o *Short Message Service* (SMS), que conquistou os surdos devido à mobilidade, praticidade e independência para comunicação à distância. Com isso, o TDD passou a ser visto como um aparelho grande, fixo e dependente de uma central. Segundo (LIEBENBERG; LOTRIET, 2010), em um estudo realizado na África do Sul, o TDD era o aparelho menos usado pelos surdos em comparação ao email, SMS e fax. Os autores destacaram que a dependência de um outro ouvinte, a falta de conhecimento do uso do aparelho e a pouca quantidade destes foram os fatores que mais contribuíram para que o TDD não fosse adotado em larga escala pela comunidade surda.

Os celulares continuaram a evoluir até que se tornassem smartphones, novos meios de acessibilidade também surgiram juntamente com a tecnologia e com o reconhecimento da Libras como segunda língua oficial do Brasil. Em decorrência desses fatos, surgiram alguns aplicativos de dicionário e tradutores de Português - Libras, tais como: Hand Talk, VLibras, Rybená e Uni Libras. A Hand Talk também oferece totens es-

palhados em alguns shoppings do país, como o RioMar de Recife[3] e Fortaleza. Deste modo, um surdo pode tirar dúvidas e se localizar melhor em um ambiente movimentado.

Em paralelo a essa evolução, uma nova gama de possibilidades surgiu devido a uma maior aplicação de conceitos da Inteligência Artificial (IA). A IA pode ser definida como "a capacidade de uma máquina realizar funções que, se realizadas pelo ser humano, seriam consideradas inteligentes" (MCCARTHY; MINSKY; ROCHESTER, 1959). A IA pode ser aplicada em diversos cenários como recomendação para usuários, *chatbots*, processamento de linguagem natural e reconhecimento de imagens e vídeos. Este último cenário poderia ser deveras útil para a inclusão de surdos, dado que reconhecendo os sinais de uma certa pessoa, seria possível identificar o que ela quis dizer.

O Alexa é um projeto *Open Source* que utiliza o *TensorFlow.js* e a *Web Cam* para fazer reconhecimento de gestos através de vídeo, desenvolvido por Abhishek Singh em 2018. Através da câmera do notebook e de uma assistente pessoal da Amazon, o Alexa é capaz de identificar os sinais de *American Sign Language* (ASL) e reproduzir de forma falada o que foi dito para que o assistente pessoal ouça e responda a pergunta feita. Ao responder, o Alexa ouve e transcreve na tela para que a resposta seja lida [4]. Tendo o Alexa como inspiração para o desenvolvimento de uma tecnologia assistiva voltada à comunidade surda que tem a Libras como sua primeira língua, o presente trabalho também foca no reconhecimento e classificação de gestos.

Devido à complexidade de identificar gestos dinâmicos, o foco foi dado aos gestos estáticos e que utilizam apenas as mãos. Em um trabalho anterior (BASTOS, 2016), foi aplicada a rede Perceptron Multi Camadas - em inglês, *Multi-Layer Perceptron* (MLP) - para classificar uma base de dados de gestos estáticos de Libras. Em (CHEVTCHENKO et al., 2018) é aplicada a rede *Convolutional Neural Network* (CNN): *Visual Geometry Group* (VGG) e *Feature Fusion-based Convolutional Neural Network* (FFCNN) em tarefas de reconhecimento de gestos da ASL, onde a FFCNN obteve a melhor acurácia, correspondente a 92,53%. Diante disso, supõe-se que a rede FFCNN também pode obter a melhor acurácia na classificação dos gestos de Libras em comparação com a MLP e VGG.

[3] https://vivariomarrecife.com.br/dia-a-dia-sustentavel/conheca-hugo-totem-digital-em-libras-no-riomar/

[4] https://medium.com/tensorflow/getting-alexa-to-respond-to-sign-language-using-your-webcam-and-tensorflow-js-735ccc1e6d3f

1.1 Objetivos

O principal objetivo deste trabalho consiste em identificar dentre algoritmos implementados com as redes MLP, VGG e FFCNN o que demonstra melhor acurácia na classificação de gestos estáticos de Libras. A hipótese levantada no início deste estudo reside em, assim como no reconhecimento de gestos de ASL, a FFCNN obtém a melhor acurácia. De modo a alcançar o objetivo, pretende-se:

- Realizar uma analise de classificadores para gestos de Libras não explorados na literatura;
- Estender a análise realizada no trabalho de (CHEVTCHENKO et al., 2018);
- Utilizar testes estatísticos para embasar a comparação realizada entre os algoritmos.

Um segundo objetivo desse trabalho, mas não menos importante, é poder dispor à literatura mais um trabalho que envolva a Libras e IA. A segunda língua oficial do Brasil não é conhecida pela maioria dos brasileiros e não faz parte da grade curricular escolar, sendo usada quase que exclusivamente pelos surdos. Devido ao baixo índice de ouvintes fluentes em Libras, os surdos tornam-se bastante dependentes de alguém que os entenda para realizar atividades básicas como ir ao médico ou em um posto policial.Portanto, além da contribuição computacional, esse trabalho também busca contribuir e incentivar para que novas pesquisas possam surgir e dessa forma, trazer mais visibilidade a Libras, tornando-a mais conhecida e por conseguinte uma sociedade mais inclusiva.

1.2 Escopo

Este trabalho contempla os seguintes itens:

- A contextualização da Libras na sociedade brasileira e sua importância para a cultura surda;
- Uma introdução ao Aprendizado de Máquina como um meio para auxiliar a inclusão de surdos;
- A apresentação de redes perceptron e convolucionais que propõem a classificação de imagens de gestos estáticos de Libras;
- Uma comparação baseada em testes estatísticos que identifica o algoritmo que melhor classifica os gestos estáticos de Libras.

Embora diretamente relacionados à este trabalho, não serão contemplados:

- A classificação de gestos dinâmicos de Libras;
- A classificação de gestos estáticos mas que envolvam partes do corpo e/ou rosto;
- A aplicação dos algoritmos em imagens não inseridas na base de dados estudada.

1.3 Organização da Monografia

Este trabalho está dividido em seis capítulos. Após este capítulo introdutório, o capítulo 2 fala sobre os conceitos que serão aplicados. O capítulo 3 traz a revisão da literatura, enquanto que o capítulo 4 discorre sobre a metodologia utilizada neste trabalho. O capítulo 5 descreve os resultados obtidos neste estudo, enquanto que o capítulo 6 conclui o estudo apresentando os trabalhos futuros.

2 Fundamentos

Este capítulo aborda conceitos necessários para o entendimento deste trabalho. A seção 2.1 discorre sobre a Libras. A seção 2.2 fala sobre o aprendizado de máquina, uma fundamental área da IA. Por último, a seção 2.3 descreve os algoritmos utilizados neste trabalho baseados em redes neurais.

2.1 Língua Brasileira de Sinais (Libras)

A história da Libras está intimamente ligada com a história dos surdos brasileiros, visto que a sua criação visava suprir uma necessidade de comunicação existente na época. Até o século XV os surdos eram considerados ineducáveis, até que no século seguinte, na Europa, ocorreram mudanças quanto a esse ponto de vista. A convite de D. Pedro II em 1857, o surdo europeu Eduard Huet veio ao Brasil para fundar o Imperial Instituto de Surdos Mudos (atualmente renomeado Instituto Nacional de Educação dos Surdos – INES), sendo esta a primeira escola para surdos do país. Durante muitos anos foi a unica instituição especializada na educação de surdos no Brasil e América Latina, e apenas pessoas do sexo masculino podiam frequentar [1].

O termo ”surdo”é vastamente utilizado para se referir a pessoas portadoras da deficiência auditiva, em primeira instancia pode causar desconforto a quem fala, com receio de ofender o surdo. Porém, o termo ”deficiente auditivo”não é bem aceito pela comunidade surda, e isso se deve ao fato de que, segundo (ESPOTE; SERRALHA; SCORSOLINI-COMIN, 2013) a palavra ”deficiente”tornou-se um termo pejorativo. A expressão ”surdo-mudo”, para os surdos é motivo de desagrado e desconforto, levando em consideração que os surdos tem a capacidade biológica de fala em perfeito estado (GESSER, 2008) e a maioria não fala pelo simples fato de não ouvir. A mudez é uma outra deficiência e os casos em que uma pessoa nasce surda-muda são considerados raros [2].

Juntamente com o advento do INES, através da junção da Língua Francesa de Sinais e de gestos já utilizados por surdos brasileiros, foi dada origem à Libras por volta de 1951. No entanto, apenas em 2002 a Lei que Libras foi sancionada, reconhecendo a Libras como meio legal de comunicação e expressão.

Durante o processo de elaboração dos sinais de Libras, cinco parâmetros são levados em consideração, são eles: a configuração de mão, ponto/local de articulação,

1 http://helb.org.br
2 http://blog.handtalk.me/historia-lingua-de-sinais/

orientação/direcionamento das mãos, movimento e expressão facial e/ou corporal.

O presente trabalho utiliza um *dataset* composto por sinais estáticos e que não dependem do rosto e/ou corpo para serem executados. Visando seguir um padrão para o processo de reconhecimento e classificação do sinal executado, apenas a mão é levada em consideração, visto que um dos algoritmos utilizados para a comparação nessa pesquisa segmenta apenas a mão durante o processo.

2.2 Aprendizagem de máquina

Segundo (MITCHELL et al., 1990), a aprendizagem de máquina pode ser definida como um "*programa de computador aprende a partir da experiência E com relação a tarefas T e medida de desempenho P, se o seu desempenho em tarefas em T, medida pelo P, melhora com a experiência E*". Por exemplo, dado um programa que executa a classificação de imagens de animais, a partir da tarefa T de classificar um animal e a experiência E de previamente conhecer a imagem de vários animais previamente, ele pode melhorar sua acurácia P à medida que sua experiência aumenta, ou seja, quando ele classifica diversas imagens de animais.

De modo geral, um algoritmo de aprendizagem de máquina necessita de uma base de dados com muitos valores. Deste modo, é possível que a máquina consiga inferir um modelo capaz de captar as nuances do cenário específico. A base de dados geralmente é dividida entre dois conjuntos: treino e teste. O conjunto de treino é maior, englobando entre 60% e 80% dos dados, sendo usado para criar um modelo capaz de gerar boas estimativas. Por outro lado, o conjunto de teste é utilizado para verificar se o modelo classificou corretamente os dados.

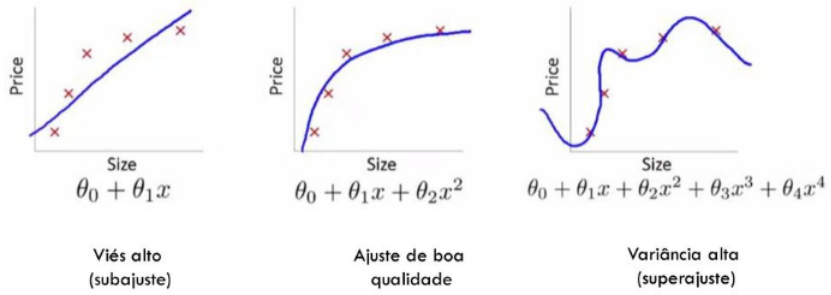

Figura 2 – Possibilidades que podem ocorrer no conjunto de treino: subajuste, desejado, e sobreajuste. Fonte: (BEZERRA, 2018)

No modelo criado no conjunto de treino, o ideal é que as estimativas sejam generalizadas ao ponto que, mesmo com um leve desvio, o modelo possa identificar o resultado de maneira próxima aos dados reais, como mostrado no centro na Figura 2. Quando o modelo não consegue ter boas estimativas, é dito que houve um subajuste

(em inglês, *underfitting*), e o modelo possui um alto erro dado que ele foi estimado sem generalizar as nuances dos dados. Por outro lado, quando o modelo gerado a partir do conjunto de treino é extremamente próximo aos dados deste conjunto, é dito que houve um superajuste (em inglês, *overfitting*). Este caso ocorre quando o modelo decora os dados do conjunto de treino e, quando é testado com novos dados no conjunto de teste, acaba tendo um baixo desempenho.

O foco deste trabalho consiste em empregar técnicas de aprendizagem supervisionada, ou seja, classificar um conjunto de dados que já estão rotuladas ou anotadas (DOUGHERTY; KOHAVI; SAHAMI, 1995). Por exemplo, dado um grupo de imagens com gestos de Libras, elas serão a entrada do modelo junto com a resposta de cada uma (se é o sinal da letra a, b, entre outros), e então o modelo irá inferir a partir das cores, posição dos dedos, ângulos entre dedos e outras características, qual o padrão para que uma imagem seja classificada como a letra a, b ou outra do alfabeto. Após o treino, o modelo é aplicado no conjunto de teste sem as respostas, ou seja, ele tentará inferir sobre os dados não treinados qual é a resposta de cada entrada. No fim, as respostas inferidas pelo modelo são comparadas com as respostas reais do conjunto de teste e são calculadas métricas como a acurácia. A Figura 3 mostra o desempenho esperado comparando a acurácia do conjunto de treino e o conjunto de teste (na Figura, generalização), ou seja, eles devem convergir. Caso haja uma diferença grande entre eles, ou há um sobreajuste (quando a acurácia do treino é muito maior do que o teste) ou um subajuste (quando ambas as acurácias são baixas).

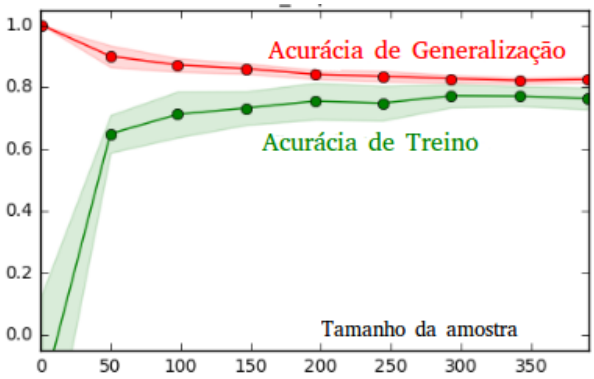

Figura 3 – Gráfico comparativo da acurácia dos conjuntos de generalização (teste) e treino. Fonte: (FACURE, 2018)

A aprendizagem não-supervisionada ocorre quando precisamos dividir os dados em grupos específicos, sendo bastante usada em sistemas de recomendação de séries, filmes e músicas, uma vez que grupos de pessoas tendem a escutar ou assistir as mesmas coisas. Por outro lado, a abordagem semi-supervisionada busca penalizar o modelo quando ele age de modo errado, como por exemplo em carros autônomos, onde um movimento errado pode ser extremamente perigoso. Assim, a partir da penalidade, os erros são diminuídos.

2.3 Redes Neurais

Assim como na biologia, as redes neurais são estruturas capazes de transmitir informação a partir de estímulos. Tais estímulos são passados entre os neurônios, a unidade fundamental de uma rede neural. Os neurônios são ativados por estes estímulos e repassam o que souberam para o próximo neurônio através das sinapses.

No aspecto computacional, as redes neurais são ligações entre neurônios que dão peso à informações recebidas pelos neurônios anteriores, de modo a convergir para os resultados previamente conhecidos (HANSEN; SALAMON, 1990) e, portanto, é uma técnica de aprendizagem supervisionado. A Figura 4 mostra um exemplo de uma rede neural. Os neurônios são os componentes circulares, e as ligações entre eles são as retas. Os dois neurônios iniciais, em verde, são as entradas do modelo, como por exemplo o tamanho e a quantidade de quartos de um apartamento. A partir destes valores, os neurônios passam as informações para os neurônios em sequência, que calculam um valor baseado nas entradas através de uma função de ativação e passam para o neurônio de saída (em amarelo). Este neurônio calcula também uma função e produz a saída do modelo, como por exemplo, o preço do apartamento.

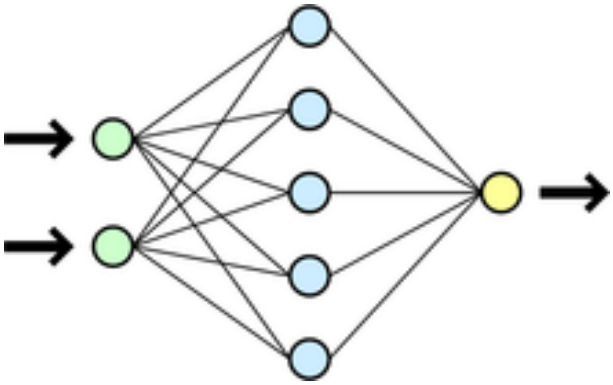

Figura 4 – Exemplo de uma rede neural. Fonte: (DAKE, 2005)

Para alcançar o resultado, os conexões entre neurônios possuem um peso, que é calculado a partir das entradas e do resultado associado. Quanto mais distante do valor desejado, maior deve ser a mudança dos pesos das ligações. Por exemplo, se o preço do apartamento tem maior relação com o tamanho do que com o número de quartos, as arestas que saem do do neurônio que representa o tamanho tendem a ter maior valar do que as que saem dos neurônios que representam o número de quartos.

Os pesos são calculados baseados no número de iterações, também chamadas épocas. As entradas são inseridas na rede várias vezes em lotes (em inglês, *batches*), de modo a reduzir o erro do modelo em comparação ao valor real. O erro é calculado a partir da comparação do valor estimado e do valor real, como mostrado na Eq. 2.1. Os valores de θ_0 e θ_1 correspondem aos pesos das conexões que saem dos neurônios de

entrada (em nosso exemplo, o tamanho e o número de quartos de um apartamento), calculados pela fórmula do erro médio quadrático.

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2 \tag{2.1}$$

O erro é diminuído a partir do gradiente descendente. Esta técnica muda os pesos das conexões baseado nas iterações e no valor de α: a taxa de aprendizagem. Quanto maior o valor de alpha, maior a mudança de parâmetros, enquanto que uma taxa de aprendizagem baixa pode levar o modelo a convergir de modo extremamente lento. A equação 2.2 mostra o cálculo do gradiente descendente baseada na função de custo.

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} f(\theta_0, \theta_1) \text{ for j=0,1} \tag{2.2}$$

Dado o número de iterações definidas, a rede tende a convergir e o erro diminui. Quando houver a convergência, o valor dos pesos estará definido e dará ao modelo a melhor estimativa possível quando uma nova entrada no modelo for calculada. Neste trabalho, alguns tipos de redes são aplicadas, como descrito nas próximas subseções.

2.3.1 Perceptron Multi camada (MLP)

A MLP consiste em uma das abordagens mais genéricas de uma rede neural, porém bastante poderosa. A Figura 5 mostra a estrutura básica de uma MLP, o perceptron. Note que o perceptron é parte de uma rede neural, no qual um neurônio recebe a entrada de outros neurônios e calcula uma função baseada nos pesos destas entradas. Neste exemplo, x_1, x_2 e x_3 são as entradas, enquanto w_1, w_2 e w_3 são os pesos. A função $g(.)$ é chamada de função de ativação, na qual o valor de y dá o resultado da função. A função de ativação para tarefas de classificação funciona como um limiar: caso o valor gerado por essa função dê acima do limiar, é classificada como 1, se não, classifica como zero.

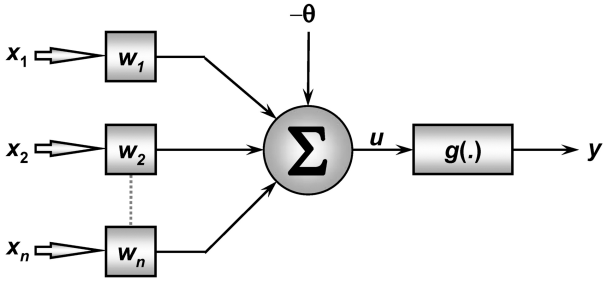

Figura 5 – Exemplo de um perceptron. Fonte: (LAMA, 2016)

Uma das funções de ativação mais conhecidas é a sigmoide, como definida na Eq. 2.3. Tal função, $h_\theta(x)$, é calculada a partir da soma dos pesos e das entradas, definida por z, e retorna um valor entre 0 e 1. A função sigmoide tem um limiar de 0.5, ou seja, se o valor calculado está acima de 0.5, o perceptron classifica como 1. Por outro lado, se o valor estiver abaixo de 0,5, o perceptron classifica como zero. A *Rectifier Linear Unit* (ReLU), a tangente hiperbólica (tanh), entre outras funções de ativação, também podem ser usadas.

$$h_\theta(x) = \frac{1}{1 + e^{-z}} \tag{2.3}$$

A MLP, como diz o nome, consiste em perceptrons com mais de duas camadas. Por exemplo, a rede neural apresentada na Figura 4 pode ser classificada como uma MLP, uma vez que é uma rede de três camadas. A primeira camada é a de entrada, possuindo o número de neurônios iguais à quantidade de entradas. A segunda camada é chamada escondida (em inglês, *hidden*), e busca incrementar as estimativas uma vez que ela recebe todas as combinações de entrada e infere características que não seriam abordadas se a próxima camada fosse a de saída. Por último, a camada de saída (em inglês, *output*) dá o resultado da estimativa. Caso hajam mais classes para estimar (por exemplo, definir se é gato, cachorro ou panda), é preciso inserir o número de neurônios igual ao número de classes e retorna-se o valor de 0 ou 1 para cada uma.

A rede neural pode melhorar o aprendizado com o emprego do *dropout*, que consiste em retirar aleatoriamente alguns elementos da camada escondida e treinar a rede com os neurônios restantes. Deste modo, para cada iteração, há o efeito de que uma rede diferente está sendo treinada[3], o que pode criar uma melhor combinação de pesos para aplicar no problema em questão.

2.3.2 Redes Neurais Convolucionais (CNN)

Dentre as redes neurais existentes, a CNN (também chamada de ConvNet) compreende a classe de redes usadas para detecção de imagens e vídeos (ZHANG et al., 1988). A CNN se baseia na similaridade de características entre pixels próximos, ou seja, em uma determinada região da imagem um determinado conjunto de pixels pode representar uma boca, um olho ou um nariz. A CNN recebe como entrada uma matriz - geralmente do tamanho do número de *pixels* da altura e largura da imagem - e retorna a classificação da imagem com base nas respostas previamente conhecidas.

A CNN se difere das demais redes neurais pois, antes da imagem ser classificada por uma rede densamente conectada (similar à MLP), ela passa por várias camadas de filtros e ajusta-os durante o treinamento de modo a garantir uma melhor

[3] http://deeplearningbook.com.br/capitulo-23-como-funciona-o-dropout/

acurácia. Uma destas camadas de filtros é a convolução que busca extrair características importantes da imagem como as bordas, sombras, tons de preto, entre outros. A Figura 6 mostra um exemplo desta abordagem. A imagem original recebe os chamados filtros, que buscam realçar características importantes da imagem e criam o mapa de características (em inglês, *feature maps*). Tal mapa representa uma visão diferente da imagem acerca das diferentes características e, por conseguinte, pode-se repetir este procedimento para os demais filtros e criar mais mapas. Desse modo, cada mapa de características detecta alguma parte importante da imagem que a diferencia de outra classe, como por exemplo os olhos do gato em comparação a um cachorro ou a silhueta da moto em relação ao carro.

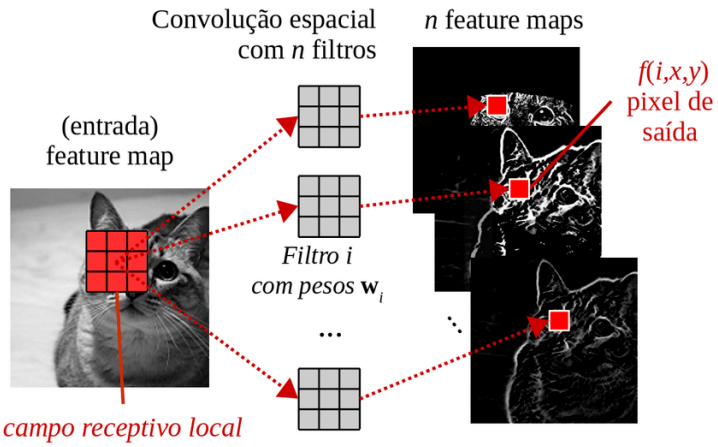

Figura 6 – Exemplo de uma convolução aplicando filtros à uma imagem. Fonte: (PONTI; COSTA, 2018)

A CNN aplica os filtros por toda a matriz de pixels da imagem. Em imagens coloridas, é comum a aplicação dos filtros em cada um dos canais RGB (*Red,Green, and Blue*). A Figura 7 mostra um exemplo de convolução no canal vermelho. O filtro é de tamanho 2x2 e passa por toda a imagem de dimensão 3x3, iniciando da extremidade superior esquerda até a inferior direita. Neste exemplo, o filtro multiplica os valores correspondentes na imagem 3x3 e os soma, ou seja, no primeiro exemplo retorna $1 \times 1 + 1 \times 3 + 6 \times 0 + 18 \times 0 = 4$. De modo geral, após a convolução, o valor de cada pixel segue uma função de ativação, como se cada um dos pixels fosse um neurônio numa rede. A mais utilizada pela CNN é a ReLU, cuja dá o valor zero para resultados negativos e o valor resultante para positivos. Esta abordagem é extremamente útil uma vez que valores iguais a zero representam a ausência de cores (neste caso, o preto).

A convolução pode ser configurada por dois parâmetros principais: o tamanho de passo (em inglês, *stride*) e a profundidade (em inglês, *depth*). O primeiro parâmetro configura de quantos em quantos pixels o filtro vai percorrer na imagem, ou seja, neste exemplo o tamanho do passo é 1. Por outro lado, a profundidade especifica o número de filtros de uma convolução. A imagem resultante após a convolução é menor do que

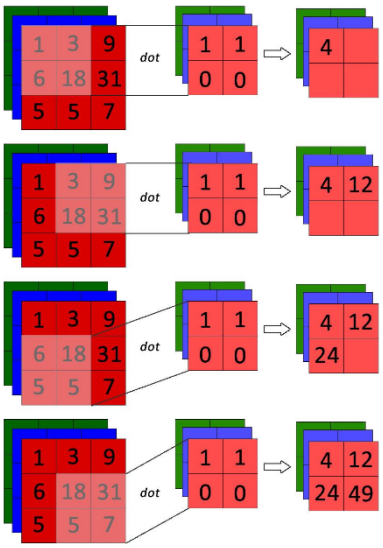

Figura 7 – Exemplo de uma convolução com passo um em um canal vermelho. Fonte: (AZEVêDO, 2016)

a original, mantendo a estratégia de que pontos próximos dão características muito semelhantes.

Além da convolução e da ativação, uma outra técnica utilizada é a de *pooling*. O *pooling* consiste em reduzir ainda mais a imagem, o que diminui o número de entradas da próxima camada e, com isso, mitiga a possibilidade de sobreajuste. A ideia do *pooling* é reduzir a convolução e extrair características. A imagem é percorrida por uma unidade de área e, dentro dela, o valor máximo ou o médio dentre eles é escolhido e passado para a imagem resultante. A Figura 8 exemplifica a operação de *max pooling*, ou seja, resultando o valor máximo, em uma área 2x2 com passo dois. Note que a imagem era 4x4 e passou a ser 2x2 uma vez que o passo foi de tamanho 2, o que permitiu um salto de dois pixels.

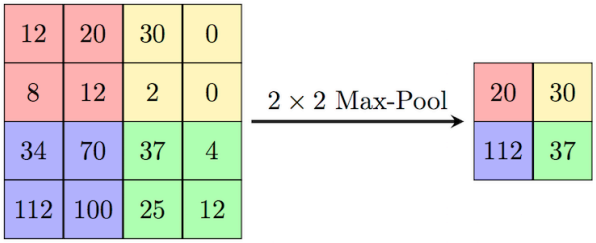

Figura 8 – Exemplo de uma uma operação de *max pooling* com passo dois e tamanho do *pool* 2x2. Fonte: (SCIENCE, 2018)

As camadas de convolução, ativação e *pooling* são aplicadas o número de vezes definido na arquitetura. Após a CNN resultar o número de mapas de características requerido pela arquitetura, estes mapas são aplainados (em inglês, *flattened*, o que mantém os dados em uma única dimensão) e são inseridos como entradas em

uma rede neural tradicional totalmente conectada (geralmente uma MLP). A Figura 9 mostra um exemplo de uma CNN que classifica uma célula como normal ou anormal. É válido notar que as camadas de convolução, ativação (ReLU) e *pooling* foram aplicadas duas vezes cada, sendo o resultado de uma servindo como entrada na seguinte. No final, todos os pixels correspondentes aos mapas de características resultantes serviram como entrada em uma rede totalmente conectada com neurônios de saída que classificavam como normal ou anormal.

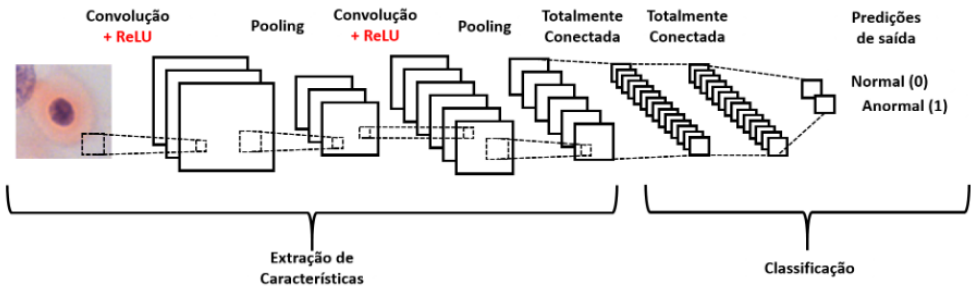

Figura 9 – Exemplo de uma CNN para classificação de uma célula normal ou anormal. Fonte: (ARAÚJO et al., 2017).

Em resumo, a CNN busca extrair da imagem um conjunto de características que as identifiquem de maneira quase única e classificam este conjunto de modo tradicional usando uma rede totalmente conectada como a MLP. Existem tipos de CNN que diferenciam-se pelo modo em que a convolução ou o *pooling* é realizado ou pelo número de camadas. A seguir vamos diferenciar as duas formas de CNN usadas neste trabalho: A FFCNN e a VGG.

2.3.2.1 FFCNN

Proposta por (CHEVTCHENKO et al., 2018), a FFCNN busca juntar o resultado de diferentes aplicações de filtros em uma imagem em uma única CNN. Desse modo, ao invés de criar uma rede com múltiplas camadas, é possível criar duas redes com menos camadas e juntá-las posteriormente, o que diminui a quantidade de parâmetros a serem calculados. A FFCNN baseia-se em três pilares:

- A rede CNN tradicional que recebe uma imagem 32x32 com duas camadas de convolução + ativação e duas de *max pooling*;
- A mesma CNN, porém, que recebe uma imagem após ser aplicada por um filtro Gabor;
- A filtragem do contorno do gesto com os momentos de Zernike.

O filtro Gabor busca eliminar os ruídos de uma imagem através de equações senoidais. Ele é usado em tarefas de segmentação (JAIN; FARROKHNIA, 1991) e

pode ser aplicado, por exemplo, em impressões digitais como mostrado na Figura 10. Deste modo, a aplicação da imagem com e sem ruídos em duas redes idênticas pode aumentar significativamente a acurácia, uma vez que mais características podem ser extraídas. Para mais detalhes sobre o formalismo matemático do filtro, veja (JONES; PALMER, 1987).

Figura 10 – Aplicação de um filtro Gabor em uma impressão digital. Note a redução significativa dos ruídos. Fonte: (NAYAN, 2016)

A outra característica importante é o contorno da mão, filtrada pelo momento de Zernike (ZERNIKE, 1934). Deste modo, como ilustrado na Figura 11, a mão tem seus contornos inseridos também como entrada da rede densamente conectada que executa a classificação da imagem em M atributos de características, assim como os mapas de características obtidos a partir da imagem filtrada por Gabor (mais acima) e a original (mais abaixo).

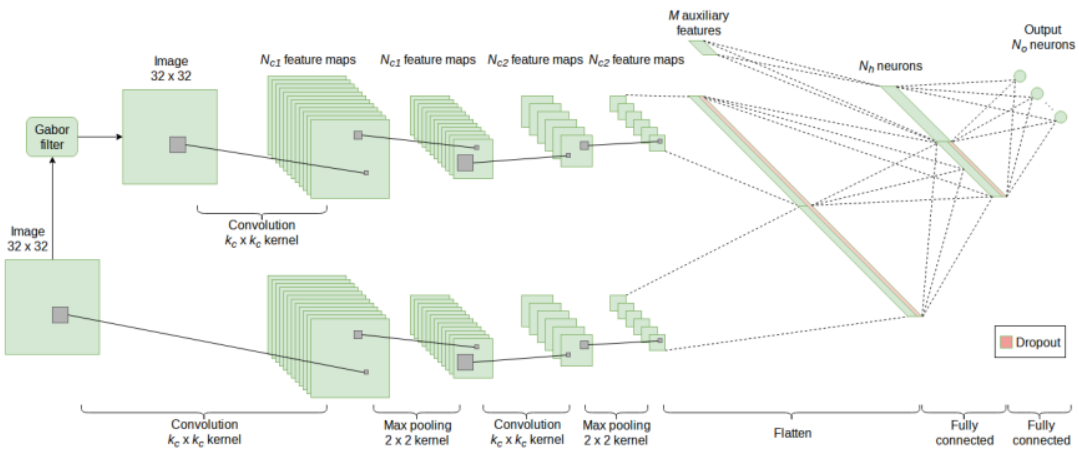

Figura 11 – Exemplo de uma FFCNN. Note que são criadas duas redes idênticas, mas a mais acima é aplicada após o filtro Gabor. Os M atributos de características correspondem aos momentos de Zernike. Fonte: (CHEVTCHENKO et al., 2018).

2.3.2.2 VGG

A arquitetura VGG surgiu em 2014 (SIMONYAN; ZISSERMAN, 2014) como proposta na competição da *ImageNet Large Scale Visual Recognition Challenge* (ILSVRC) em 2014, sendo a vice campeã em relação à taxa de erros, mas com uma maior taxa de acurácia em tarefas de localização, ou seja, em reconhecer que uma determinada parte da imagem é uma face, por exemplo. A VGG pode possuir entre 11 e 19 camadas, o que dá o nome às redes (VGG11, VGG13, VGG16, VGG19). A VGG pode ser esquematizada seguindo algumas características:

- Os filtros de convolução são sempre 3x3, reduzindo o número de parâmetros em comparação a um filtro 7x7;
- O tamanho do passo é um para extrair o máximo da imagem;
- Usa o ReLU para ativar os pixels após a convolução;
- Usa a estratégia *max pooling*;
- Possuem entre 11 e 19 camadas, sendo ao menos três camadas densamente conectadas e cinco de *max pooling*, sendo estas últimas sempre após uma camada de convolução.

3 Revisão Bibliográfica

Este capítulo visa fazer um levantamento do estado da arte de pesquisas e estudos que abordam o reconhecimento de gestos através de conceitos bem difundidos no meio computacional.

O ramo da IA responsável por identificar conteúdos de imagens é a visão computacional, que é discutida em diversos outros trabalhos presentes na literatura. Conforme (MARENGONI; STRINGHINI, 2009) a visão computacional normalmente tem seu início com o processamento de imagem. E, segundo (RAFAEL et al., 2010), a lacuna entre processamento de imagem e visão computacional se dá em três níveis: baixo-nível, nível-médio e alto-nível. Em seu trabalho (MARENGONI; STRINGHINI, 2009) define o baixo-nível como responsável por melhorias consideradas primitivas na imagem, tais como alteração de níveis de brilho, cor e contraste; o nível-médio como operações um pouco mais complexas tratando de segmentação (separar o que se deseja do resto da imagem); e por fim, o alto-nível é capaz de executar tarefas cognitivas análogas à visão humana.

O processo de visão pelos seres-humanos se dá primeiramente pelo reflexo da luz nos objetos que segue em linha reta para o olho, que processa essa imagem e manda através de impulsos elétricos pelo nervo óptico para o cérebro. De modo quase instantâneo, o cérebro é capaz de processar e armazenar a informação que a ele chega. Logo, há um processo longo e complexo e de forma análoga, a visão computacional utiliza hardwares e softwares avançados em busca de modelar, de forma artificial, a visão humana e suas funções.

Em seu trabalho, (KOROISHI; SILVA, 2015) busca através de sensores 3D do Kinect, classificar 12 sinais de Libras diferentes utilizando uma abordagem probabilística. O Kinect captura a imagem, segmenta a mão e calcula a probabilidade do significado daquele sinal. O desafio principal foi reconhecer as configurações de mão e para isso foram utilizados modelos estruturados em nuvens de pontos e o algoritmo *Iterative Closest Point*. O sistema foi testado 48 vezes tendo o tempo médio para reconhecimento de 127s por teste e obteve uma taxa de acurácia de aproximadamente 65%.

O reconhecimento de gestos tem sido amplamente estudado nos últimos anos (KHAN; IBRAHEEM, 2012), assim como é o trabalho de (BASTOS, 2016), que visa reconhecer determinados gestos de Libras através de descritores e classificadores que extraem características das imagens relevantes para o seu reconhecimento. Para a extração dessas informações optou-se pelos descritores Histograma de Gradientes Orientados (HOG) e Momentos Invariantes de Zernike (MIZ) . O classificador Percep-

tron Multicamada foi utilizado para o reconhecimento. O autor enfatiza a inexistência de *datasets* públicos de Libras, dessa forma justificando o esforço de criar um *dataset* de 9600 imagens subdivididas em 40 sinais da Libras. Após a realização de testes com um conjunto de imagens diferentes do qual o classificador foi treinado, foi obtido uma taxa de acurácia de 96,77%, evidenciando dessa forma, um alto índice de acertos.

Ainda tratando do reconhecimento de Libras, (TEODORO, 2015) propõe o reconhecimento automático de sinais de Libras em vídeos, simulando ao máximo situações reais, não utilizando ambientes controlados de som e imagem ou luvas especiais para identificação dos movimentos. Primeiramente, os gestos em Libras são identificados nos vídeos e segmentados em sequencias de imagens, para que em seguida seja feita a classificação do sinal segmentado e sua tradução para o português. Durante a segmentação são aplicados alguns algoritmos, tais como: *Kovac*, *Osmann*, *Swift* e para a classificação o *Random Forest* é utilizado. O autor criou um banco de imagens com 30 palavras básicas escolhidas por um especialista em Libras. No preparo para a segmentação foi verificada a capacidade do algoritmo de diferenciar o que era pele do que não era, obtendo assim uma acurácia de 90% para posterior segmentação. Dentre os sinais segmentados, foi obtida uma assertividade de 70%, o que corresponde a 422 amostras de palavras do banco de imagens construíıdo, sendo todas essas classificadas corretamente.

Além da Libras, outras Línguas de Sinais (LS) também são utilizadas em trabalhos similares. É o caso de (NETO; MENEZES et al., 2018) que utiliza a Língua de Sinais Argentina (LSA) . O uso da LSA ao invés da Libras é justificado pela falta de *datasets* públicos e a mesma afirmação também é feita por (BASTOS, 2016) e (TEODORO, 2015) em seus respectivos trabalhos. A dissertação de (NETO; MENEZES et al., 2018) utiliza técnicas de visão computacional e aprendizado de máquina através de uma arquitetura 3D *Convolutional Neural Network* (CNN) visando desenvolver uma arquitetura genérica para o reconhecimento automático de qualquer língua de sinais. A LSA foi aplicada na arquitetura desenvolvida e uma acurácia de 94,22% foi obtida, mostrando-se uma metodologia promissora se comparada aos trabalhos relacionados no reconhecimento automático de línguas de sinais.

A *American Sign Language* (ASL) é usada por (CHEVTCHENKO et al., 2018) visando reconhecer gestos estáticos em tempo real através do desenvolvimento de uma arquitetura baseada em CNN capaz de suportar variações na imagem como de escala, iluminação e/ou rotação. O FFCNN, arquitetura proposta, combina redes convolutivas com descritores de características tradicionais. São extraídas características tradicionais das imagens através de MIZ, momentos de Hu, filtros de Gabor e propriedades de contorno das mãos. A base de dados foi subdividida em representações: profundidade, escala de cinza e binária, com o intuito de mensurar como a representação da

imagem pode influenciar nos classificadores. Como forma de avaliar cada arquitetura foram levados em consideração os seguintes pontos: velocidade de classificação, acurácia e métodos de validação. O resultado foi de 92,53% de acurácia, 5,93% melhor em comparação ao modelo mais acurado existente até então. Dessa forma, o modelo gerado supera o estado da arte quando se trata de reconhecer gestos em tempo real.

A Tabela 1 mostra um resumo da comparação dos trabalhos apresentados neste capítulo. Neste estudo, a LS utilizada é a Libras assim como (KOROISHI; SILVA, 2015), (BASTOS, 2016) e (TEODORO, 2015). Neste trabalho pretende-se comparar o FFCNN, VGG e MLP em uma base de dados criada por (BASTOS, 2016). Dentre os trabalhos correlatos descritos nessa seção, apenas (NETO; MENEZES et al., 2018) e (CHEVTCHENKO et al., 2018) utilizam a CNN e apenas (BASTOS, 2016) utiliza a MLP. Levando em consideração o tamanho do *dataset*, o que contém maior variedade de sinais distintos é o utilizado por (NETO; MENEZES et al., 2018) composto por 64 gestos. Logo em seguida tem-se o *dataset* criado por (BASTOS, 2016) composto por 40 gestos que também é utilizado nesse trabalho. Em todos os trabalhos, exceto em (BASTOS, 2016) e nesse proposto, a captura dos sinais é feita através de vídeos em tempo real. Por último, um diferencial do presente trabalho reside no uso de testes estatísticos para validar se os valores obtidos são estatisticamente diferentes. Tais testes foram aplicados apenas no de (CHEVTCHENKO et al., 2018) como uma comparação de melhoria da rede após aplicar um procedimento para esticar as imagens

Tabela 1 – Tabela comparativa dos trabalhos apresentados neste capítulo.

Trabalho	Usa Libras	Algoritmos de Reconhecimento	Número de gestos	Reconhecimento por vídeos	Testes estatísticos
(KOROISHI; SILVA, 2015)	Sim	Nuvem de pontos e ICP	12 (Libras)	Sim	Não
(BASTOS, 2016)	Sim	MLP	40 (Libras)	Não	Não
(TEODORO, 2015)	Sim	Random Forest	30 (Libras)	Sim	Não
(NETO; MENEZES et al., 2018)	Não	CNN	64 (LSA)	Sim	Não
(CHEVTCHENKO et al., 2018)	Sim	FFCNN, CNN e VGG	36 (ASL)	Sim	Não
Proposta	Sim	FFCNN, VGG e MLP	40 (Libras)	Não	Sim

4 Metodologia

Este capítulo descreve como o trabalho foi avaliado e oferece detalhes referentes à arquitetura, dataset, métricas e infraestrutura necessárias para este estudo. A seção 4.1 mostra os algoritmos comparados neste trabalho. A seção 4.2 descreve o dataset de imagens, enquanto que a secão 4.3 define as métricas usadas neste trabalho. Finalmente, a seção 4.4 descreve as tecnologias utilizadas.

4.1 Algoritmos de reconhecimento

Nesta seção, são definidos como os algoritmos foram implementados para realizar a comparação. Este trabalho compara duas arquiteturas baseadas em CNNs: FFCNN e VGG. A escolha destas arquiteturas baseou-se no trabalho de (CHEVTCHENKO et al., 2018), no qual comparou as duas arquiteturas para reconhecer gestos de ASL. Neste trabalho, estas arquiteturas buscam reconhecer gestos de Libras de modo a verificar se a FFCNN obtém também o melhor resultado como obtido em (CHEVTCHENKO et al., 2018). Estas CNNs foram implementadas no trabalho e tiveram seus resultados coletados. Os resultados da MLP proposta por (BASTOS, 2016) são, então, comparados com os resultados obtidos, ou seja, a MLP não é implementada neste trabalho.

4.1.1 FFCNN

A rede FFCNN aplicada nas tarefas de classificação é a mesma implementada no trabalho de (CHEVTCHENKO et al., 2018). A rede possui uma classificação via CNN com a imagem em 32x32 original e outra com a mesma CNN na mesma imagem após a aplicação do filtro de Gabor. As CNNs idênticas que são aplicadas em cada imagem são compostas, sequencialmente, por: uma camada de convolução com 32 filtros 5x5, tamanho de passo 1 e ativação via ReLU; uma camada de *max pooling* 2x2 com tamanho de passo 2; uma segunda de convolução idêntica à primeira com exceção do número de filtros (cai de 32 para 16) e, em sequência, uma outra camada de *max pooling* também idêntica à primeira desta.

A camada de MLP (densamente conectada) possui o número de neurônios de entrada iguais à soma da quantidade de *pixels* dos mapas de características das saídas das CNNs anteriores (veja a Figura 11 para mais detalhes). Estes neurônios têm uma taxa de *dropout* de 61% e servem de entrada para a camada escondida seguinte, que possui 200 neurônios com uma taxa de *dropout* de 76%. Esta camada escondida

recebe também os valores dos momentos de Zernike e Hu com mais 200 neurônios e sem dropout. No fim, a camada escondida gera valores para a camada de saída e classifica o sinal. A função de ativação em todas as camadas da MLP é a softmax.

Para mais detalhes sobre os parâmetros usados nos filtros de Gabor e nos momentos de Zernike e Hu, veja (CHEVTCHENKO et al., 2018).

4.1.2 Descritores e MLP

A rede MLP utilizada neste trabalho foi desenvolvida por (BASTOS, 2016) em sua dissertação. O primeiro passo, consiste em aplicar dois descritores em cada uma das imagens: o primeiro é o MIZ, enquanto que o segundo é o Histograma de Gradientes Orientados (em inglês, *Histogram Oriented Gradient* - HOG). O HOG é utilizado em tarefas de detecção de objetos em imagens, reduzindo efeitos de luz e sombra (DALAL; TRIGGS, 2005). Para mais detalhes sobre os parâmetros usados no HOG e no Zernike, veja o trabalho de (BASTOS, 2016).

Após a aplicação do HOG e do momento de Zernike, o resultado consiste em um vetor de características. Tal vetor é usado como entrada na MLP. Esta MLP usa uma abordagem de dois estágios: o primeiro classificava a imagem em diferentes grupos baseado na similaridade dos gestos (ex: T e F); enquanto que o segundo classificador recebia a classificação do grupo anterior e classificava qual dos sinais existentes no grupo era referente àquela imagem. A arquitetura classificava inicialmente a imagem em um dos doze grupos definidos compostos de 2 à 6 sinais, como mostrado na Figura 12. Esta etapa de classificação possui 44 neurônios na camada escondida e doze neurônios referentes à cada grupo. Então, dentro de cada grupo, uma nova camada escondida é aplicada, sendo 20 neurônios para o grupo 1; 23 para os grupo 2 e 8; 16 neurônios para os grupos 3, 5, 6, 9 e 11; 15 neurônios para o grupo 4; 23 neurônios para os grupos 7 e 10; e 13 neurônios para o grupo 12. O número de neurônios de saída de cada grupo corresponde ao número de imagens presentes nele.

A classificação utilizou a função de ativação sigmoide. dada na Eq. 2.3. Os valores classificados como maiores que 0,7 são considerados saídas positivas, enquanto que 0,3 são saídas negativas (BASTOS, 2016). Valores entre 0,3 e 0,7 são consideradas como incorretas (por exemplo, sinais que não estão no grupo). Considerando o grupo 6 da Figura 12, uma imagem da letra F pode ser corretamente avaliada se a ativação do neurônio F for 0,9 e a do neurônio T for 0,2. Em caso de uma imagem que não é F nem T, cada um dos neurônios do grupo 6 pode até classificar a imagem entre 0,3 e 0,7, mas será dado como incorreto.

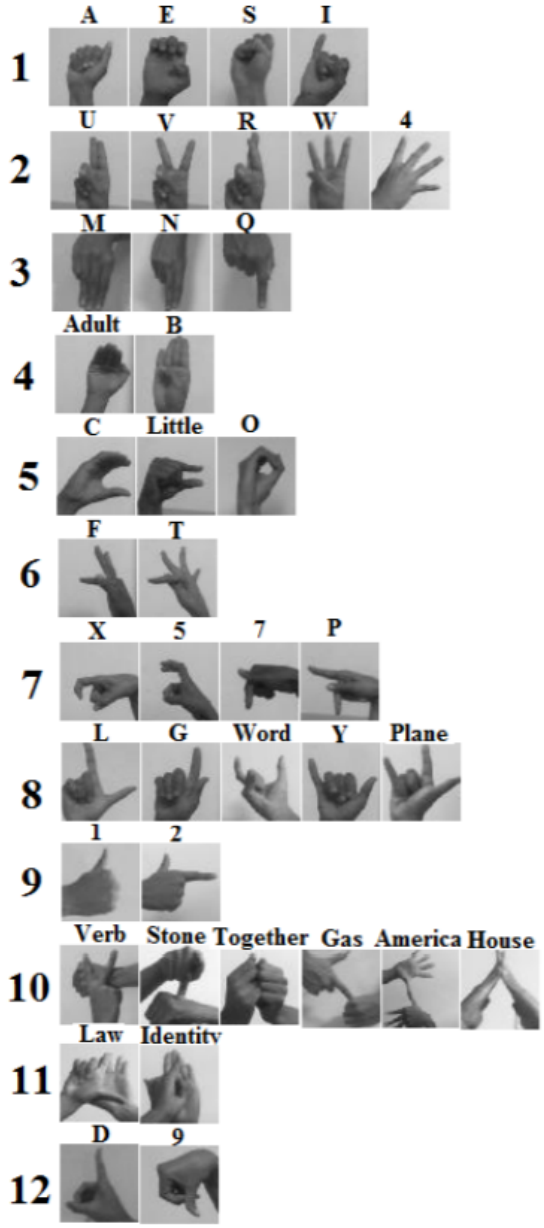

Figura 12 – Agrupamento de sinais usados em (BASTOS, 2016)

4.1.3 VGG

A rede VGG aplicada nas tarefas de classificação é a mesma utilizada no trabalho de (SIMONYAN; ZISSERMAN, 2014) e dentre as VGGs descritas em seu trabalho, as VGG16 e VGG13 foram implementadas e serão utilizadas nessa pesquisa.

A VGG16 é uma arquitetura CNN composta por 16 camadas com pesos associados à suas operações, como em camadas convolutivas e densas (ou totalmente conectadas). A primeira e segunda camada são convolutivas composta por 64 filtros 3x3; a terceira e a quarta contém 128 filtros, também 3x3; a quinta, sexta e sétima camada são compostas por 256 filtros 3x3; da oitava a décima são compostas por 512 filtros 3x3; da décima primeira à décima terceira temos a mesma configuração da ante-

rior; o último passo da VGG16 é composto por uma MLP cuja entrada é a quantidade de *pixels* de modo unidimensional resultantes da camada anterior (dada por mapas de características), seguida por duas camadas ocultas compostas por 4096 neurônios e ativação relu e por fim a última camada da CNN é a camada de saída, onde o número de neurônios é estritamente igual a o número de classes existentes no modelo e a função de ativação é a softmax. Dentre cada uma camadas agrupadamente descritas há uma camada de *max pooling* de tamanho 2. Todas as camadas de convolução têm passo de tamanho 1, a função de ativação relu e logo em seguida o *batch normalization*. *O dropout* entre os grupos de camadas convolutivas e de *max pooling* de 0,25, enquanto que entre a última camada oculta e a de saída é de 0,5.

De modo similar, a VGG13 possui 13 camadas com pesos e se difere da VGG16 nas camadas que possuem 256 e 512 filtros, reduzindo duas camadas de 256 e uma camada de 512. As demais configurações são idênticas à VGG16. O uso da VGG13 busca verificar se o seu resultado é similar ou até mesmo melhor do que o da VGG16, uma vez que é possível aplicar a VGG13 no mesmo cenário com um número de camadas menor que a VGG16, reduzindo o custo de treinamento.

4.2 Dataset de imagens

Encontrar uma base de dados para a Libras não é uma tarefa fácil devido à pouca quantidade de imagens de gestos agrupadas na internet. Deste modo, o trabalho de (BASTOS, 2016) buscou criar um *dataset* de Libras para aplicar a MLP e deixá-lo disponível para a comunidade.

A criação do *dataset* utilizou gestos que não têm nenhum tipo de movimento, ou seja, as letras H, K, J e Z não foram inseridas. Além disso, sinais praticamente idênticos como 3 (em comparação ao W) e 8 (em comparação ao S) também foram excluídos. Os sinais dos números 6 e 9 também foram evitados uma vez que mudava apenas a rotação. Sinais que envolvem outras partes do corpo como por exemplo “conhecer” (envolve a cabeça), “amigo” (envolve o peito) e “aluno” (envolve o braço), mesmo que estáticos, também foram excluídos pois o foco era em captar imagens de uso exclusivo das mãos.

As imagens foram adquiridas com auxílio de 3 especialistas e 2 alunos surdos fluentes em Libras. No processo, 40 imagens foram adquiridas: os numerais de um dígito (exceto 0, 3, 6 e 9), o alfabeto (sem as letras K, H, J e Z) e as palavras “adulto”, “américa”, “avião”, “casa”, “gasolina”, “identidade”, “juntos”, “lei”, “palavra”, “pedra”,“pequeno” e “verbo”. Três adultos do sexo feminino, um adulto do sexo masculino e um adolescente do sexo masculino foram os indivíduos escolhidos de modo que a base possua diferentes tamanhos de mão, cor e variações de posição do sinal. Para

cada sinal, 240 imagens foram criadas, sendo a metade de máscaras binárias de modo a identificar a região da pele dos sinais. A Figura 13 mostra alguns sinais do dataset com suas respectivas máscaras binárias.

Figura 13 – Alguns sinais e suas respectivas máscaras binárias. Fonte: (BASTOS, 2016)

4.3 Métricas de avaliação

As métricas de avaliação são utilizadas para medir a eficiência de determinado modelo de classificação. A obtenção de um resultado em que a classe positiva é classificada corretamente é nomeado por Verdadeiro Positivo (VP). Ainda falando sobre uma classificação definida como correta, quando uma classe negativa é classificada como tal, esta é nomeada Verdadeiro Negativo (VN). Quando uma classificação é feita de maneira errônea dois casos podem acontecer: os Falsos Positivos (FP) são aqueles em que uma classe negativa é classificada como uma classe positiva; e, de forma contrária aos FP, um modelo pode classificar uma classe positiva como negativa, sendo esses resultados conhecidos por Falsos Negativos (FN). A seguir, são descritas as métricas usadas neste trabalho que relacionam as os valores de VP, VN, FV e FN.

4.3.1 Acurácia (*Accuracy*)

A acurácia retrata a fração de previsões que o modelo acertou, levando em consideração o número de predições feitas corretamente pelo número de predições totais. Porém, esta medida não se comporta bem quando tem-se um número de VN maior que o de VP, pois seu percentual tenderá a ser alto mesmo que o modelo não traga resultados condizentes. Por exemplo, suponha que existam 10 sinais da letra F e 90 sinais da letra T e um certo algoritmo busca identificar se a letra é T ou F. Se o algoritmo tem a tarefa de classificar a letra F, classificando corretamente 2 letras como F (VP) e 88 como não F (VN), a acurácia será de 90%, uma vez que 90 sinais de 100 foram identificados com sucesso. Porém, em relação aos sinais da letra F, ele classificou apenas 2 de 10 com sucesso, gerando 8 FP e 2 VP.

A acurácia pode ser descrita através da resposta obtida pela pergunta: De forma geral, com que frequência classificador está correto? A resposta para essa pergunta, pode ser obtida pela fórmula presente na Eq. 4.1.

$$Acuracia = \frac{VP + VN}{VP + VN + FP + FN} \tag{4.1}$$

4.3.2 Precisão (*Precision*)

A precisão leva em consideração a classe positiva, identificando a frequência com que a classe foi predita corretamente. Utilizando a mesma problemática descrita na seção 4.3.1, a precisão da classificação da letra F é 50%, uma vez que ele classificou corretamente 2 sinais como F e outros dois sinais erroneamente como F (visto que ele entendeu que 88 dos 90 sinais de T eram de fato T), ou seja, 2 dos 4 sinais entendidos como F eram realmente F. Esta métrica pode ser descrita através da resposta obtida pela pergunta: Dentre os que foram classificados como corretos, quantos realmente eram? A resposta para essa pergunta, pode ser obtida pela fórmula descrita na Eq. 4.2.

$$Precisao = \frac{VP}{VP + FP} \tag{4.2}$$

4.3.3 Revocação (*Recall*)

Essa métrica tende a indicar o quão preciso foi o modelo, levando em consideração todos os valores que deveriam ter sidos classificados como positivos e os que foram. Dada a problemática descrita na seção 4.3.1, a revocação da classificação da letra F é 20% dado que 2 de 10 sinais de F foram corretamente classificados. A Revocação pode ser descrita através da resposta obtida pela pergunta: De todos os possíveis rótulos positivos, quantos o modelo identificou corretamente? A resposta para essa pergunta, pode ser obtida pela fórmula descrita na Eq. 4.3.

$$Revocacao = \frac{VP}{VP + FN} \tag{4.3}$$

4.3.4 F1-score

Como descrito na seção 4.3.1 usar apenas a acurácia pode não ser tão confiável para atestar a qualidade de um modelo. Diante disso, o F1-score pode ser considerado uma medida de confiabilidade da acurácia, quanto mais alto, menor as possíveis dis-

torções. Esta medida é composta por outras duas medidas estudadas nesse capítulo: revocação e precisão, a partir de uma média harmônica demonstrada na Eq. 4.4

$$F1 = \frac{2 * Precisao * Revocacao}{Precisao + Revocacao} \tag{4.4}$$

Utilizando o mesmo exemplo da seção 4.3.1, a F1-score da letra F é 28% dado que a precisão corresponde a 0,5 e a revocação 0,2. Desta forma, é verificado que a acurácia correspondente a 90% não é tão confiável se analisada juntamente com o F1-score.

4.4 Tecnologias Utilizadas

A linguagem de programação utilizada durante todo o trabalho foi o Python 3.6.8. Para definir a arquitetura do projeto, foi necessário uso do OpenCV 2 como dependência do Keras que usou como backend o TensorFlow 1.13. Para rodar os experimentos foi utilizada uma máquina com as seguintes configurações:

- Processador: Intel Core i7 (Sétima Geração) - 2 núcleos,
- Memória RAM: 8GB,
- SSD: 256GB,
- GPU: GeForce 940MX.

Para acelerar o tempo gasto com testes, alguns experimentos foram rodados de forma paralela no Google Colab, com as seguintes configurações:

- Processador: Intel(R) Xeon(R) - 2 núcleos,
- Memória RAM: 13GB,
- GPU: Tesla T4.

4.5 Parâmetros utilizados na comparação

Neste trabalho, as redes VGG13, VGG16 e a FFCNN são executadas na base de dados disponibilizadas por (BASTOS, 2016). A MLP em conjunto com os descritores proposta por (BASTOS, 2016) não foi executada, ou seja, os resultados desta rede serão comparados a partir dos valores obtidos em sua dissertação. A Tabela 2 lista os valores utilizados na comparação. O estudo varia a taxa de aprendizagem entre três

valores para entender o impacto desta taxa no conjunto de treino e, em estudos futuros, buscará variar os demais hiperparâmetros. Este trabalho não visa encontrar os melhores hiperparâmetros do modelo e, portanto, aplica o modelo gerado no conjunto de treino diretamente no conjunto de teste, obtendo a acurácia estudada. Os valores de tamanho de lote são os recomendados para cada rede. Cada experimento é realizado dez vezes, ou seja, em cada repetição, de modo aleatório, 75% das imagens são escolhidas para treino e 25% delas para teste. Ao final deste processo, a média e o desvio padrão das acurácias são obtidos.

Tabela 2 – Parâmetros usados nesta comparação

Parâmetros	**Valores**
% de treino	75%
% de teste	25%
Repetições	10
Otimizador	SGD (VGG), Adam (FFCNN)
Função de erro	*Categorical Cross Entropy*
Tamanho do lote	32 (VGG), 100 (FFCNN)
Número de épocas	100
Taxa de aprendizagem	0,01 - 0,001 - 0,0001

5 Resultados e Discussões

Este capítulo descreve os resultados obtidos neste estudo. A seção 5.1 mostra a análise das acurácias obtidas em cada algoritmo. A seção 5.2 descreve os valores de precisão, revocação e F1 das redes obtidas. A seção 5.3 indica se as acurácias encontradas por cada algoritmo pode ser considerada estatisticamente diferente em relação às demais. Por último, a seção 5.4 apresenta a discussão.

5.1 Acurácia dos algoritmos

A Figura 14 mostra uma comparação entre as redes estudadas neste trabalho. As redes VGG obtém melhores acurácias quando a taxa de aprendizagem é mais alta (0,01), ou seja, a rede consegue aprender rapidamente. Os valores obtidos foram de 99,1% para a VGG13 e 99,4% para a VGG16. Por outro lado, a FFCNN obtém o seu pior resultado com apenas 3,2%. Nesse mesmo cenário, os desvios padrão também obtiveram os melhores resultados, a FFCNN com 0,45%, seguida pela VGG13 e VGG16 correspondentes à 0,54% e 0,85% respectivamente.

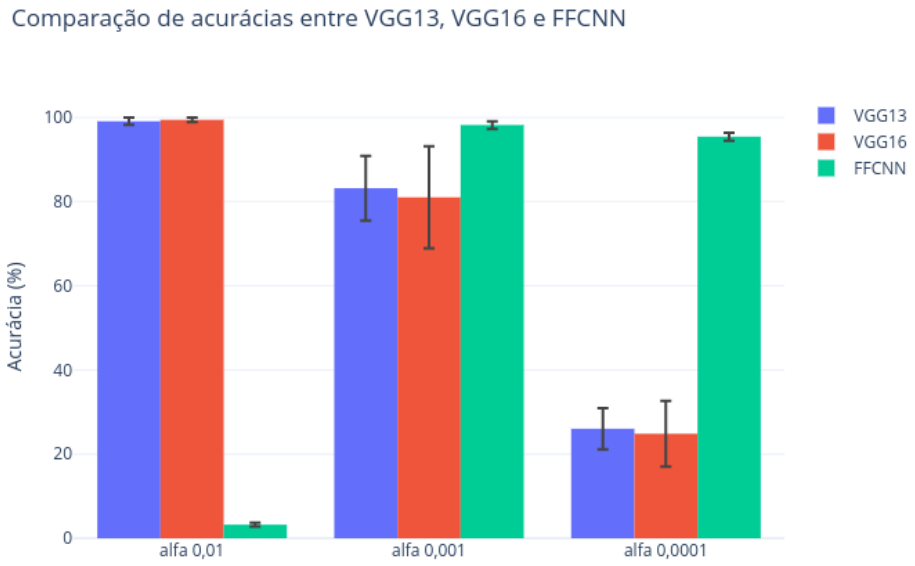

Figura 14 – Comparação das acurácias obtidas nos algoritmos estudados. A barra de erros mostra o desvio padrão das amostras.

Quando a taxa de aprendizagem é menor, a FFCNN obtém resultados melhores e equiparáveis à VGG. Com a taxa em 0,001, a FFCNN obtém uma acurácia de

98,2% e um desvio padrão de 0,91% contra 81% de acurácia da VGG16 e 12,1% de desvio padrão, por fim a VGG13 detém 83,2% de acurácia e 7,68% de desvio padrão. Quando a taxa é ainda menor, as redes alcançam resultados piores em comparação à taxa anterior. A FFCNN obtém cerca de 95% de acurácia e 0,93% de desvio padrão, enquanto que a VGG13 obtém 26% e 7,6% de acurácia e desvio padrão respectivamente e a VGG16 com o maior intervalo de desvio padrão correspondente a 7,83% seguido de 24,8% de acurácia. Ao aumentar a taxa de aprendizagem de 0,0001 para 0,001 é perceptível uma decrescente no desvio padrão de todos os modelos, mas em contrapartida há um aumento nas acurácias. Ou seja, o modelo se tornou capaz de classificar mais sinais corretamente. Mesmo com uma variância maior o ganho é positivo para os modelos VGG, visto que o ganho da acurácia é exponencialmente maior do que o desvio padrão.

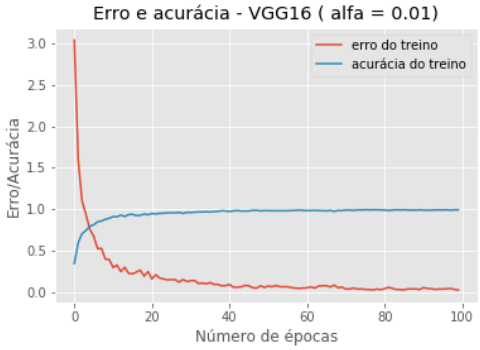

Figura 15 – Gráfico de erro x acurácia para a VGG 16 com a taxa de aprendizagem α 0,01

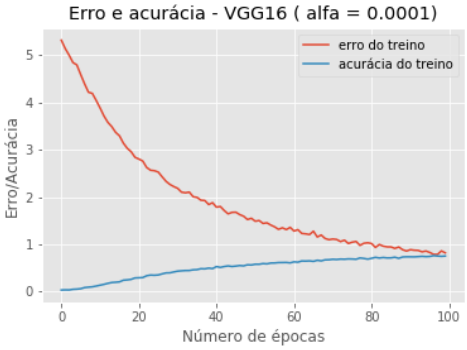

Figura 16 – Gráfico de erro x acurácia para a VGG16 com a taxa de aprendizagem α 0,0001

As Figuras 15 e 16 mostram os gráficos de erro x acurácia para o melhor ($\alpha = 0,01$) e pior ($\alpha = 0,0001$) treinamento com a VGG16, ou seja, estes gráficos lidam apenas com o conjunto de treino. É válido notar que o decaimento ocorre em uma rápida escala quando a taxa de aprendizagem é 0,01, o que dá ao algoritmo uma excelente acurácia já na 30ª época. Por outro lado, a acurácia caminha a passos lentos quando a taxa é 0,0001, o que requer um número de épocas muito maior do que as cem inseridas neste estudo.

Assim como na VGG16, as Figuras 17 e 18 demonstram os gráficos de erro x acurácia para a VGG13 com as mesmas taxas de aprendizagem. O comportamento acaba sendo bastante parecido com a VGG16, embora a convergência ocorra com menor velocidade. Caso o número de épocas fosse maior que cem, a VGG13 poderia obter resultados melhores com a taxa de aprendizagem em 0,0001.

Com respeito aos valores relacionados à FFCNN, as Figuras 19 e 20 mostram o erro e a acurácia obtidas no melhor resultado. A FFCNN consegue também aumen-

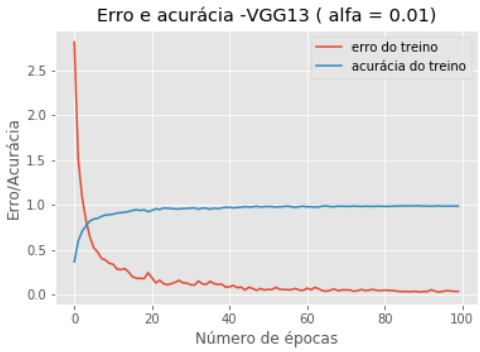

Figura 17 – Gráfico de erro x acurácia para a VGG 13 com a taxa de aprendizagem α 0,01

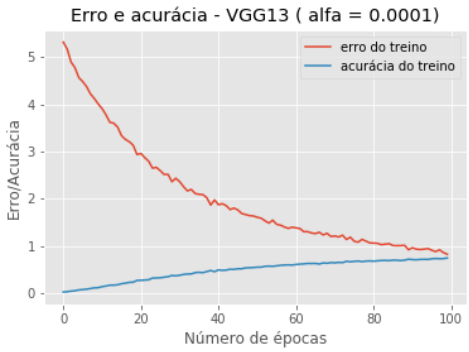

Figura 18 – Gráfico de erro x acurácia para a VGG13 com a taxa de aprendizagem α 0,0001

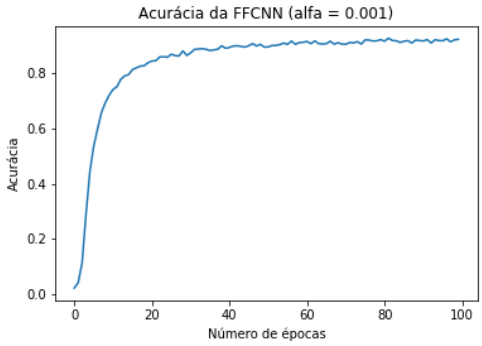

Figura 19 – Gráfico de acurácia para a FFCNN com a taxa de aprendizagem α 0,001

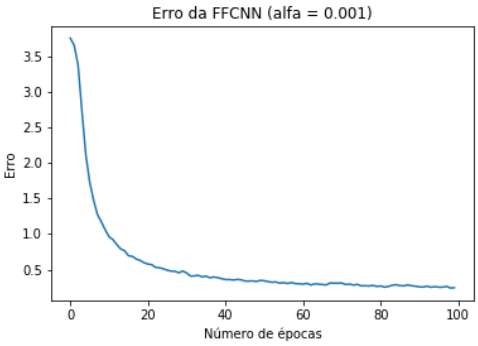

Figura 20 – Gráfico de erro para a FFCNN com a taxa de aprendizagem α 0,001

tar bastante sua acurácia e reduzir seu erro, podendo até obter resultados um pouco melhores se aumentar o número de épocas.

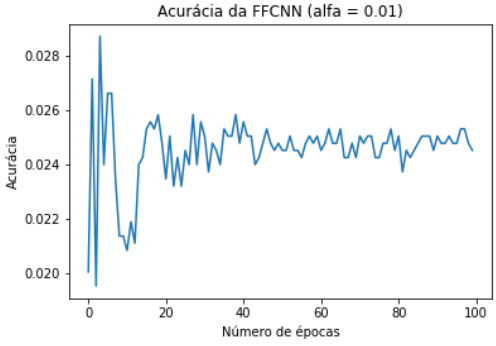

Figura 21 – Gráfico de acurácia para a FFCNN com a taxa de aprendizagem α 0,01

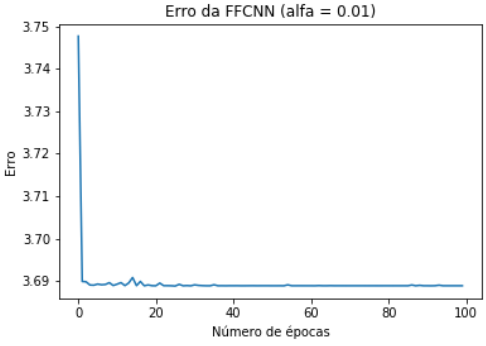

Figura 22 – Gráfico de erro para a FFCNN com a taxa de aprendizagem α 0,01

Considerando os valores relacionados à taxa de aprendizagem igual a 0,01, os resultados são demasiadamente diferentes, como mostram as Figuras 21 e 22. O erro e a acurácia não mudam após a 20ª época, ou seja, a taxa de aprendizagem

provavelmente irá gerar um modelo altamente enviesado com um alto erro (3,69) e baixa acurácia (cerca de 2,4%). Deste modo, essa configuração pode ser descartada visto que mesmo com um alto número de épocas, o resultado tende a permanecer igual. A Tabela 3 compara as acurácias obtidas nesta seção com as obtidas por (BASTOS, 2016)

Tabela 3 – Comparação da acurácia obtida pelos algoritmos neste estudo

Rede	**Melhor acurácia (%)**
(BASTOS, 2016)	96,77%
(CHEVTCHENKO et al., 2018)	98,12%
VGG13	99,09%
VGG16	99,45%

5.2 Precisão, Revocação e F1-score

A Tabela 6 demonstra os resultados obtidos pela VGG16, o algoritmo de melhor acurácia, utilizando a taxa de aprendizagem igual a 0,01.

É válido notar que, na média, todos os sinais obtiveram ao menos 95% de precisão e 94% de revocação, o que garantiu um F1-*score* alto, acima dos 96% em todos os casos. Os sinais correspondentes à “2”, “4”, “5”, “américa”, “avião”, “e”, “gasolina”,“identidade”, “junto”, “n”, “palavra”, “pedra”, “t”, “w” e “x” alcançaram 100% no F1-*score*, o que também caracteriza 100% nos índices de precisão e revocação destes sinais.

Dentre os sinais classificados imprecisamente, o sinal correspondente à letra “s” obteve o menor índice, totalizando 95,71%. O sinal da letra “m” obteve 95,97% de precisão, correspondendo ao segundo menor índice. Em seguida, os sinais de “b” e “c” obtiveram valores de precisão iguais a 96,50% e 96,60%. Os valores de desvio padrão fornecem um razoável indicativo de como a escolha das imagens influencia no resultado do experimento. Em relação à precisão, os maiores desvios ocorrem nas letras “m” (8,21%), “g” (7,9%) e “s” (6,38%).

Em relação aos valores referentes à revocação, a menor média corresponde à letra “q”, com 94,40%, seguida pela letra “d” com 96% e a letra “f” com 96,67%. As letras “d” e “q” também lideram no quesito desvio padrão, com 12,65% e 12,08% respectivamente. Assim como ocorreu na precisão, a letra “g” alcançou um alto desvio, com valor igual à 7,15%.

Com respeito aos valores de F1-*score*, as piores médias ocorreram nas classificações das letras “q” e “g” e “d”, com 96,71%, 97,11% e 97,34%, respectivamente. Estas letras também possuem os mais altos índices de desvio padrão da métrica, correspondendo à mais de 5,5% em todos os casos.

Os valores das médias da precisão da VGG16 com α igual a 0,01, quando comparados aos valores do trabalho de (BASTOS, 2016), embasam a melhora da classificação do modelo VGG16 em relação à MLP com descritores. Dentre os 40 sinais, a VGG16 obteve melhor precisão em 28. A MLP obteve 4 sinais com melhor acerto e os oito sinais restantes obtiveram o mesmo valor (100,00%). Como destaque, os sinais das letras “i”, “b” e da palavra “pequeno” obtiveram as maiores taxas de melhora, aumentando a precisão em 10,44%, 9,83% e 8,67% respectivamente. Os quatro sinais em que o trabalho de (BASTOS, 2016) obteve o melhor resultado compreendem as letras “g” (1,66% melhor), “s” (1,20%), “m” (0,70%) e “p” (0,27%). Deste modo, é válido notar que as melhorias na precisão aplicadas pela VGG16 correspondem à uma maior escala do que os melhores resultados da MLP, visto que nenhum sinal em que a MLP obteve melhor resultado foi superior em mais de 2%.

5.3 Testes estatísticos

De modo à identificar a diferença entre os resultados apresentados, este trabalho aplica testes estatísticos para validar se um resultado é estatisticamente melhor do que outro. Dado que o teste t é consolidado pela literatura e aplicado em vários problemas que se adequam à uma distribuição normal, o teste t será aplicado neste trabalho.

A comparação será feita com os resultados obtidos nos conjuntos de teste treinadas pelas melhores configurações encontradas para cada rede: A VGG13 e a VGG16 com $\alpha = 0,01$ e a FFCNN com $\alpha = 0,001$. Assume-se que as médias encontradas nas repetições de cada rede seguem uma distribuição *t-student*, com n-1 graus de liberdade e que as amostras são independentes entre si. A distribuição *t-student* é considerada uma aproximação da normal pois utiliza o desvio padrão da amostra e não da população como a normal usa. Quanto mais graus de liberdade, mais próximo da normal ela está.

Além de assumir a normalidade, o teste t consiste de dois passos: verificar se variância das médias a serem comparadas são significativamente diferentes através do cálculo usando a distribuição F; e o cálculo da diferença entre amostras utilizando a distribuição *t-student*, verificando se uma amostra é diferente da outra. As comparações a serem feitas serão:

- VGG13 x VGG16
- FFCNN x VGG13
- VGG16 X FFCNN

5.3.1 Verificando se as variâncias são iguais

A aplicação do teste F de equidade de variâncias assume que as variâncias são retiradas de uma distribuição normal e que as amostras são independentes. A Eq. 5.1 (JOHNSON; KOTZ; BALAKRISHNAN, 1970) define o cálculo do teste. Os valores n_a e en_b correspondem ao número de amostras a e b, respectivamente, e indicam o número de graus de liberdade da distribuição. De modo similar, os valores S_a^2 e S_b^2 correspondem às variâncias das amostras a e b, dado que $S_a^2 > S_b^2$.

$$F_{n_a-1,n_b-1} = \frac{S_a^2}{S_b^2} \tag{5.1}$$

Nas comparações, todas as amostras têm tamanho 10, ou seja, a distribuição F sempre terá nove graus de liberdade para o numerador e denominador. O nível de confiança do teste é de 95%. Deste modo, a tabela F indica que, com 95% de confiança, $F_{9,9} = 3,179$, ou seja, se a razão entre variâncias estiver abaixo deste valor, não há como rejeitar que as variâncias são iguais. A Tabela 4 mostra a comparação entre as redes. Dado que em todos os casos o valor da razão das variâncias foi menor que 3,179, não há evidências estatísticas para rejeitar que as variâncias são iguais em todos os casos.

Tabela 4 – Comparação das variâncias entre as redes.

Comparação	Razão entre variâncias	Variâncias iguais(<3,179)
VGG13 x VGG16	2,51	Sim
FFCNN x VGG13	1,15	Sim
VGG16 x FFCNN	2,89	Sim

5.3.2 Verificando se as médias das amostras são iguais

Dado que as variâncias das amostras são iguais, o próximo passo é verificar de fato se as médias das amostras podem ser consideradas iguais. O teste t é definido por:

- H_0: as médias das amostras são iguais (hipótese nula)
- H_1: as médias das amostras são diferentes (hipótese alternativa)

Como as variâncias foram iguais, o cálculo do teste t para a hipótese nula é dado pela Eq. 5.2.

$$t_0 = \frac{\mu_a - \mu_b}{\sqrt{S_p^2(\frac{1}{n_a} + \frac{1}{n_b})}} \tag{5.2}$$

Os valores μ_a e μ_b correspondem às médias das amostras, com $\mu_a > \mu_b$. Por outro lado, o valor de S_p^2 corresponde à média ponderada entre as variâncias amostrais S_a^2 e S_b^2, com $n_a + n_b - 2$ graus de liberdade, definido pela Eq 5.3. Vale salientar que a distribuição t calculada pela Eq. 5.2 também possui $n_a + n_b - 2$ graus de liberdade.

$$S_p^2 = \frac{(n_a - 1)S_a^2 + (n_b - 1)S_b^2}{n_a + n_b - 2} \tag{5.3}$$

Dado que as amostras têm tamanho dez, a distribuição t possuirá 18 graus de liberdade. De.ste modo, com 95% de confiança, se o valor de t_0 for maior que 2,101, é possível rejeitar a hipótese nula, ou seja, que as médias das amostras são iguais. A Tabela 5 mostra a comparação. Em todos os casos, a média do algoritmo mais à esquerda (de maior valor) é diferente do algoritmo da direita, o que configura a rejeição da hipótese nula. Portanto, podemos dizer com significância estatística, que a acurácia da FFCNN é menor que a da VGG13, assim como a acurácia desta última é menor que a da VGG16.

Tabela 5 – Teste t aplicado na comparação entre amostras

Comparação	t0 com 18 graus de liberdade	Médias diferentes (>2,101)
VGG16 x VGG13	2,48	Sim
VGG13 x FFCNN	5,02	Sim
VGG16 x FFCNN	8,28	Sim

5.4 Discussão

Com base nos resultados obtidos, a VGG13 e a VGG16 convergiram rapidamente com uma taxa de aprendizagem mais alta, diferente da FFCNN. Este fato ocorre devido a menor quantidade de camadas de convolução na FFCNN, uma vez que a FFCNN possui duas camadas de convolução em cada CNN e as VGG13 e VGG16 possuem 10 e 13, respectivamente. Desse modo, a FFCNN cria uma estimativa com um grande erro estatístico e sem convergir para a redução do erro. Este fenômeno é chamado de *overshooting*, ou seja, quando a taxa de aprendizagem é tão alta que o modelo dá passos muito grandes em cada iteração e não consegue reduzir o erro.

No momento em que a taxa de aprendizagem reduz para 0,001, a FFCNN consegue convergir para uma boa acurácia, uma vez que as CNNs conseguem extrair mais características e dar passos menores à cada iteração. Por outro lado, à medida que a taxa de aprendizagem cai, a VGG chega à resultados piores devido aos passos menores dados a cada iteração, o que se configura um *overshooting*. Entretanto, como mostrado nas Figuras 16 e 18, um número muito alto de épocas iria resultar num valor

razoável de acurácia, mas como visto anteriormente, é possível chegar num resultado mais rápido aumentando a taxa de aprendizagem.

A VGG16 obtém resultados melhores que a VGG13 levando em consideração o número de camadas convolutivas. Neste estudo, não houve sobreajuste uma vez que as acurácias de teste e treino foram parecidas. Havia um risco da VGG16 resultar num sobreajuste devido à sua maior complexidade, mas não foi isso que ocorreu.

Os testes estatísticos aplicados confirmaram que a rede VGG16 possui maior acurácia entre as redes apresentadas. Embora possuísse um valor muito próximo da VGG13, o valor do teste foi superior ao limiar de 2,101 dado um intervalo de confiança de 95% e 18 graus de liberdade. O teste também destacou a diferença entre os resultados das redes VGG e FFCNN, com valores altos na comparação (5,02 entre VGG13 e FFCNN e 8,28 entre VGG16 e FFCNN), o que indica que as redes VGG obtiveram um resultado significativamente superior em relação à FFCNN.

Em relação às estimativas, a VGG16 errou em sinais muito parecidos, como entre “a” e “s”, onde 16% das letras “s” foram classificadas como “a” e entre “l” e “i”, onde 10% das letras “l” foram classificadas como “i”. A rede errou em 4,5% dos sinais de “pedra”, classificando-os como “junto”. As letras “q”, “g” e “d” obtiveram os piores índices de F1-*score* e maiores desvios padrões, o que indica que a variação de imagens referentes à estes sinais podem causar erros em maior escala do que em outros sinais, ou seja, há maior dificuldade da VGG16 em classificar estes sinais caso estejam com maior ruído ou estejam em posições diferentes do habitual.

No entanto, todos os sinais atingiram um F1-*score* maior que 96,71%, tendo 15 deles chegando aos 100% e 18 a, no mínimo, 99%. Isso demonstra que embora a VGG16 não possua descritores como os momentos invariantes de Zernike ou filtros de Gabor, ela consegue atingir resultados precisos já que as imagens do *dataset* não possuem tanta variação ou os ruídos destas imagens (sombra, interruptores, entre outros) não ocasionam diferenças em relação às imagens limpas. Deste modo, para este problema em específico, a VGG16 conseguiu se equiparar e até superar propostas que usam filtros antes de inserir a imagem na rede propriamente dita.

Tabela 6 – Precisão, Revocação e F1-Score da VGG16

Sinal	**Precisão**		**Revocação**		**F1-Score**	
	Média	Desvio	Média	Desvio	Média	Desvio
1	100,00%	0,00%	98,80%	3,79%	99,36%	2,02%
2	100,00%	0,00%	100,00%	0,00%	100,00%	0,00%
4	100,00%	0,00%	100,00%	0,00%	100,00%	0,00%
5	100,00%	0,00%	100,00%	0,00%	100,00%	0,00%
7	100,00%	0,00%	99,07%	2,07%	99,52%	1,06%
9	99,63%	1,17%	100,00%	0,00%	99,81%	0,60%
a	98,49%	2,5%	99,56%	1,37%	99,00%	1,34%
adulto	100,00%	0,00%	99,65%	1,09%	99,82%	0,55%
américa	100,00%	0,00%	100,00%	0,00%	100,00%	0,00%
avião	100,00%	0,00%	100,00%	0,00%	100,00%	0,00%
b	96,50%	1,09%	100,00%	0,00%	99,82%	0,55%
c	96,60%	1,20%	100,00%	0,00%	99,79%	0,64%
casa	100,00%	0,00%	99,78%	0,68%	99,89%	0,35%
d	99,69%	0,99%	96,00%	12,65%	97,34%	7,86%
e	100,00%	0,00%	100,00%	0,00%	100,00%	0,00%
f	98,71%	4,05%	96,67%	1,05%	99,14%	2,17%
g	97,50%	7,9%	97,37%	7,15%	97,11%	5,66%
gasolina	100,00%	0,00%	100,00%	0,00%	100,00%	0,00%
i	99,6 %	1,02%	100,00%	0,00%	99,84%	0,52%
identidade	100,00%	0,00%	100,00%	0,00%	100,00%	0,00%
junto	100,00%	0,00%	100,00%	0,00%	100,00%	0,00%
l	99,69%	0,99%	99,69%	0,99%	99,68%	0,67%
lei	100,00%	0,00%	100,00%	0,00%	100,00%	0,00%
m	95,97%	8,21%	100,00%	0,00%	97,77%	4,62%
n	100,00%	0,00%	100,00%	0,00%	100,00%	0,00%
o	100,00%	0,00%	97,24%	4,44%	98,55%	2,35%
p	99,73%	0,83%	100,00%	0,00%	99,87%	0,42%
palavra	100,00%	0,00%	100,00%	0,00%	100,00%	0,00%
pedra	100,00%	0,00%	100,00%	0,00%	100,00%	0,00%
pequeno	99,50%	1,58%	99,56%	1,37%	99,52%	1,01%
q	100,00%	0,00%	94,40%	12,08%	96,71%	7,12%
r	99,37%	1,98%	99,46%	1,71%	99,40%	1,27%
s	95,71%	6,38%	99,44%	1,76%	97,42%	3,35%
t	100,00%	0,00%	100,00%	0,00%	100,00%	0,00%
u	98,55%	3,11%	99,70%	0,96%	99,10%	1,89%
v	99,67%	1,05%	100,00%	0,00%	99,83%	0,53%
verbo	100,00%	0,00%	98,73%	2,70%	99,34%	1,40%
w	100,00%	0,00%	100,00%	0,00%	100,00%	0,00%
x	100,00%	0,00%	100,00%	0,00%	100,00%	0,00%
y	100,00%	0,00%	99,13%	0,27%	99,54%	1,44%

6 Conclusão

A Libras, embora seja a segunda língua oficial do país, ainda não tem o devido espaço no cotidiano de nossas ações na sociedade, o que diminui a inclusão de pessoas surdas no contexto social. Deste modo, estudos que busquem facilitar esta inclusão auxiliam tanto no debate de ações para a melhoria da inclusão dos surdos quanto para demonstrar novos resultados e/ou técnicas para auxiliar este processo.

De modo a dar visibilidade à Libras, o presente trabalho apresentou três algoritmos de reconhecimento de imagens, sendo um deles com variação em suas camadas (VGG13 e VGG16) e comparou a acurácia destes em um *dataset* de 40 sinais de Libras. A hipótese consistia que a FFCNN obtivesse o melhor resultado, o que não ocorreu, uma vez que a VGG16 obteve a melhor acurácia corroborada pelo teste estatístico.

A FFCNN proposta por (CHEVTCHENKO et al., 2018) obteve uma grande taxa de acerto (98%) porém obteve uma baixa acurácia com a taxa de aprendizagem mais alta, visto que o modelo CNN é bem menos complexo do que a VGG13 e VGG16 estudadas neste trabalho. Por outro lado, a convergência da FFCNN é mais rápida com uma taxa de aprendizagem menor, o que destaca que a solução com filtros (Gabor e Zernike) e redes CNN menores consegue atingir resultados acurados, enquanto que a VGG por ter muito mais camadas ainda demora a atingi-los como taxas de aprendizagem menores. Os testes estatísticos confirmaram que a VGG16 obteve resultados significativamente melhores que as demais redes, inclusive relacionados à VGG13, que poderia obter resultado semelhante com menor número de camadas.

Embora este trabalho não seja focado em segmentação, reconhecimento em tempo real e sinais dinâmicos, ele contribui com uma comparação entre algoritmos presentes na literatura com um *dataset* criado apenas em Libras. Este estudo pretende dar maior visibilidade à Libras e servir como base para trabalhos que busquem classificar sinais de Libras em determinados cenários, como em imagens e gestos estáticos.

6.1 Trabalhos Futuros

Futuramente, pretende-se aplicar mais algoritmos de reconhecimento neste *dataset*, como (NETO; MENEZES et al., 2018). Além disso, tentar estender a base de dados de (BASTOS, 2016) para inserir sinais e aumentar a variação dos sinais existentes, inserindo mais ruído. Por último, estudar técnicas de segmentação em vídeos para detecção de sinais de Libras em tempo real.

6.2 Dificuldades encontradas

O ramo da visão computacional não foi estudado pela autora durante a graduação, por não fazer parte da grade curricular do curso, o que acarretou em uma extensa carga de conteúdo durante todo o trabalho e de forma mais abundante no início, onde sua curva de aprendizado era bem mais lenta. No entanto, devido a uma grande admiração e convivência com a comunidade surda, trabalhar em uma temática de acessibilidade motivou a pesquisa desde a fase de ideação.

No decorrer desse trabalho, encontrar um *dataset* de Libras, como reforçado por (TEODORO, 2015) e (BASTOS, 2016) não é uma tarefa simples. Logo, durante a pesquisa, houve a criação de um *dataset* de Libras no intuito de dispor à literatura um *dataset* que pudesse ser utilizado por outros trabalhos como feito por (BASTOS, 2016). Infelizmente, o processo de criação do *dataset* demandou um tempo muito maior do que o estimado. A primeira versão continha 60 imagens de 38 gestos feitos por duas pessoas, incluindo alguns gestos que não estavam presentes em (BASTOS, 2016) como “cartão”, “cogumelo”, “cruz”, “eu te amo” e “parar”. Entretanto, diferentes arquiteturas da VGG não conseguiam classificar acuradamente os gestos de uma pessoa quando treinados por gestos da outra.

Deste modo, o *dataset* precisava ser melhorado para evitar os erros do algoritmo. Porém, o processo de criação do *dataset* necessitava de mais imagens e uma metodologia mais concisa. Portanto, devido ao tempo requerido para concluir o trabalho, foi usado apenas o *dataset* de (BASTOS, 2016) para efetuar comparações entre os algoritmos apresentados.

Referências

ARAÚJO, F. H. et al. Redes neurais convolucionais com tensorflow: Teoria e prática. *SOCIEDADE BRASILEIRA DE COMPUTAÇÃO. III Escola Regional de Informática do Piauí. Livro Anais-Artigos e Minicursos*, Sociedade Brasileira de Computação, v. 1, p. 382–406, 2017. Citado 2 vezes nas páginas 7 e 26.

AZEVêDO, M. *Redes Convolucionais Aplicadas em Visão Computacional*. 2016. <https://www.cin.ufpe.br/~cabm/visao/Aula08_CNN.pdf>. Acessado em 30 de junho de 2019. Citado 2 vezes nas páginas 7 e 25.

BASTOS, I. L. O. Reconhecimento de sinais da libras utilizando descritores de forma e redes neurais artificiais. Instituto de Matemática. Departamento de Ciência da Computação, 2016. Citado 15 vezes nas páginas 7, 15, 29, 30, 31, 32, 33, 34, 35, 36, 38, 43, 44, 49 e 50.

BERKE, J. *Robert Weitbrecht's Life and Legacy*. 2018. <https://www.verywellhealth.com/robert-weitbrecht-inventor-of-the-tty-1049384>. Acessado em 02 de maio de 2018. Citado 2 vezes nas páginas 7 e 14.

BEZERRA, E. *Refinamento de Algoritmos para Aprendizagem de Máquina*. 2018. <https://slideplayer.com.br/slide/12445128/>. Acessado em 30 de junho de 2019. Citado 2 vezes nas páginas 7 e 19.

CHEVTCHENKO, S. F. et al. A convolutional neural network with feature fusion for real-time hand posture recognition. *Applied Soft Computing*, Elsevier, v. 73, p. 748–766, 2018. Citado 11 vezes nas páginas 7, 15, 16, 26, 27, 30, 31, 32, 33, 43 e 49.

CI2017VIL, C. *LEI Nº 8.213, DE 24 DE JULHO DE 1991*. 1991. <http://www.planalto.gov.br/ccivil_03/leis/l8213cons.htm>. Acessado em 11 de abril de 2019. Citado na página 13.

CIVIL, C. *LEI Nº 13.146, DE 6 DE JULHO DE 2015.* 2015. <http://www.planalto.gov.br/ccivil_03/_ato2015-2018/2015/lei/l13146.htm>. Acessado em 09 de abril de 2019. Citado na página 13.

DAKE, C. *A simplified view of an artifical neural network*. 2005. <https://commons.wikimedia.org/wiki/File:Neuralnetwork.png>. Acessado em 30 de junho de 2019. Citado 2 vezes nas páginas 7 e 21.

DALAL, N.; TRIGGS, B. Histograms of oriented gradients for human detection. In: IEEE COMPUTER SOCIETY. *international Conference on computer vision & Pattern Recognition (CVPR'05)*. [S.l.], 2005. v. 1, p. 886–893. Citado na página 33.

DOUGHERTY, J.; KOHAVI, R.; SAHAMI, M. Supervised and unsupervised discretization of continuous features. In: *Machine Learning Proceedings 1995*. [S.l.]: Elsevier, 1995. p. 194–202. Citado na página 20.

ESPOTE, R.; SERRALHA, C. A.; SCORSOLINI-COMIN, F. Inclusão de surdos: revisão integrativa da literatura científica. *Psico-USF*, Universidade São Francisco, v. 18, n. 1, p. 77–87, 2013. Citado na página 18.

FACURE, M. *Aprendizado de Máquina Essencial*. 2018. <https://matheusfacure.github.io/AM-Essencial/>. Acessado em 30 de junho de 2019. Citado 2 vezes nas páginas 7 e 20.

GESSER, A. Do patológico ao cultural na surdez: para além de um e de outro ou para uma reflexão crítica dos paradigmas. *Trabalhos em linguística aplicada*, v. 47, n. 1, p. 223–239, 2008. Citado na página 18.

HANSEN, L. K.; SALAMON, P. Neural network ensembles. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, IEEE, n. 10, p. 993–1001, 1990. Citado na página 21.

IBGE. *Cartilha do Censo 2010 Pessoa com Deficiência*. 2012. <https://www.pessoacomdeficiencia.gov.br/app/sites/default/files/publicacoes/cartilha-censo-2010-pessoas-com-deficienciareduzido.pdf>. Acessado em 16 de abril de 2019. Citado na página 13.

JAIN, A. K.; FARROKHNIA, F. Unsupervised texture segmentation using gabor filters. *Pattern recognition*, Elsevier, v. 24, n. 12, p. 1167–1186, 1991. Citado na página 26.

JOHNSON, N. L.; KOTZ, S.; BALAKRISHNAN, N. *Continuous univariate distributions*. [S.l.]: Houghton Mifflin Boston, 1970. v. 1. Citado na página 45.

JONES, J. P.; PALMER, L. A. An evaluation of the two-dimensional gabor filter model of simple receptive fields in cat striate cortex. *Journal of neurophysiology*, v. 58, n. 6, p. 1233–1258, 1987. Citado na página 27.

KHAN, R. Z.; IBRAHEEM, N. A. Survey on gesture recognition for hand image postures. *Computer and information science*, Canadian Center of Science and Education, v. 5, n. 3, p. 110, 2012. Citado na página 29.

KOROISHI, G. O.; SILVA, B. V. L. Reconhecimento de sinais da libras por visão computacional. *Mecatrone*, v. 1, n. 1, 2015. Citado 2 vezes nas páginas 29 e 31.

LAMA, R. D. *Neurônio artificial*. 2016. <https://commons.wikimedia.org/wiki/File:Sadssa.png>. Acessado em 30 de junho de 2019. Citado 2 vezes nas páginas 7 e 22.

LIEBENBERG, M.; LOTRIET, H. An exploration of deaf telecommunication processes and associated social issues in south africa. *South African Computer Journal*, Sabinet, v. 2010, n. 45, p. 11–17, 2010. Citado na página 14.

LOSCHI, M. *Pessoas com deficiência: adaptando espaços e atitudes*. 2019. <https://agenciadenoticias.ibge.gov.br/agencia-noticias/2012-agencia-de-noticias/noticias/16794-pessoas-com-deficiencia-adaptando-espacos-e-atitude>. Acessado em 15 de abril de 2019. Citado na página 13.

MARENGONI, M.; STRINGHINI, S. Tutorial: Introdução à visão computacional usando opencv. *Revista de Informática Teórica e Aplicada*, v. 16, n. 1, p. 125–160, 2009. Citado na página 29.

MCCARTHY, J.; MINSKY, M.; ROCHESTER, N. *Artificial intelligence*. [S.l.], 1959. Citado na página 15.

MEC. *META 4 - INCLUSÃO*. 2019. <http://pne.mec.gov.br/21-programas-e-metas/547-meta-4-inclusao>. Acessado em 19 de abril de 2019. Citado na página 13.

MITCHELL, T. et al. Machine learning. *Annual review of computer science*, Annual Reviews 4139 El Camino Way, PO Box 10139, Palo Alto, CA 94303-0139, USA, v. 4, n. 1, p. 417–433, 1990. Citado na página 19.

MONTEIRO, M. S. História dos movimentos dos surdos e o reconhecimento da libras no brasil. *ETD-Educação Temática Digital*, v. 7, n. 2, p. 292–305, 2006. Citado na página 13.

NAYAN, A. *Gabor Filtering for Fingerprint Image Enhancement*. 2016. <https://pt.slideshare.net/ankitnayan3/gabor-filtering-for-fingerprint-image-enhancement>. Acessado em 30 de junho de 2019. Citado 2 vezes nas páginas 7 e 27.

NETO, R.; MENEZES, G. et al. Reconhecimento de língua de sinais baseado em redes neurais convolucionais 3d. Universidade Federal do Maranhão, 2018. Citado 3 vezes nas páginas 30, 31 e 49.

OLIVEIRA, E. D. S. et al. Inclusão social: professores preparados ou não? *POLÊM! CA*, v. 11, n. 2, p. 314–323, 2012. Citado na página 13.

OMS. *A ONU e as pessoas com deficiência*. 2019. <https://nacoesunidas.org/acao/pessoas-com-deficiencia/>. Acessado em 16 de abril de 2019. Citado na página 13.

PONTI, M. A.; COSTA, G. B. P. da. Como funciona o deep learning. *arXiv preprint arXiv:1806.07908*, 2018. Citado 2 vezes nas páginas 7 e 24.

RAFAEL, C. G. et al. *Digital image processing using matlab*. [S.l.]: Tata McGraw-Hill, 2010. Citado na página 29.

SCIENCE, C. *Max Pooling / Pooling*. 2018. <https://computersciencewiki.org/index.php/Max-pooling_/_Pooling>. Acessado em 30 de junho de 2019. Citado 2 vezes nas páginas 7 e 25.

SIMONYAN, K.; ZISSERMAN, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. Citado 2 vezes nas páginas 28 e 34.

TEODORO, B. T. *Sistema de reconhecimento automático de Língua Brasileira de Sinais*. Tese (Doutorado) — Universidade de São Paulo, 2015. Citado 3 vezes nas páginas 30, 31 e 50.

ZERNIKE, F. v. Beugungstheorie des schneidenver-fahrens und seiner verbesserten form, der phasenkontrastmethode. *Physica*, Elsevier, v. 1, n. 7-12, p. 689–704, 1934. Citado na página 27.

ZHANG, W. et al. Shift-invariant pattern recognition neural network and its optical architecture. In: *Proceedings of annual conference of the Japan Society of Applied Physics*. [S.l.: s.n.], 1988. Citado na página 23.