UNIVERSIDADE ESTADUAL DA PARAÍBA

CAMPUS VII – GOVERNADOR ANTÔNIO MARIZ

CENTRO DE CIÊNCIAS EXATAS E SOCIAIS APLICADAS

CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

GIUAN ADAUTO DE SOUSA ARAÚJO

UMA ABORDAGEM DE RECONHECIMENTO DE GESTOS APLICADO À LÍNGUA BRASILEIRA DE SINAIS (LIBRAS)

PATOS – PB

2018

GIUAN ADAUTO DE SOUSA ARAÚJO

UMA ABORDAGEM DE RECONHECIMENTO DE GESTOS APLICADO À LÍNGUA BRASILEIRA DE SINAIS (LIBRAS)

Trabalho de Conclusão de Curso apresentado ao Curso de Bacharelado em Ciência da Computação da Universidade Estadual da Paraíba, em cumprimento à exigência para obtenção do grau de bacharelado em Computação.

Orientadora: Dra. Kézia de Vasconcelos Oliveira Dantas

PATOS – PB

2018

A658a Araujo, Giuan Adauto de Sousa.

Uma abordagem de reconhecimento de gestos aplicado à Língua Brasileira de Sinais (LIBRAS) [manuscrito] / Giuan Adauto de Sousa Araujo. - 2018.

46 p. : il. colorido.

Digitado.

Trabalho de Conclusão de Curso (Graduação em Computação) - Universidade Estadual da Paraíba, Centro de Ciências Exatas e Sociais Aplicadas , 2018.

"Orientação : Profa. Dra. Kézia de Vasconcelos Oliveira Dantas , Departamento de Computação - CCT."

1. Redes Neurais Convolucionais. 2. Inteligência Artificial. 3. Língua Brasileira de Sinais - Libras. 4. Reconhecimento de gestos. I. Título

21. ed. CDD 006.3

Elaborada por Camile de A. Gomes - CRB - 15/559 BC/UEPB

Giuan Adauto de Sousa Araújo

UMA ABORDAGEM DE RECONHECIMENTO DE GESTOS APLICADO À LÍNGUA BRASILEIRA DE SINAIS (LIBRAS)

Trabalho de Conclusão de Curso apresentado ao Curso de Bacharelado em Ciências da Computação da Universidade Estadual da Paraíba, em cumprimento à exigência para obtenção do grau de Bacharel em Ciências da Computação.

Aprovado em 28/11/2018

BANCA EXAMINADORA

Kézia de V. O. Dantas
Dra. Kézia de Vasconcelos Oliveira Dantas
(Orientador)

Rodrigo Alves Costa
Dr. Rodrigo Alves Costa
(Examinador)

Wellington C. Araújo
Prof. Dr. Wellington Candeia de Araújo
(Examinador)

RESUMO

Este trabalho tem como objetivo o desenvolvimento de uma abordagem para reconhecimento de gestos, utilizando a Língua Brasileira de Sinais como caso de uso, de maneira não intrusiva, através do uso redes neurais artificiais. O reconhecimento de gestos é o foco de pesquisa em diversos trabalhos, uma das principais desafios técnicos é fazer tal reconhecimento de maneira não intrusiva e natural, de modo que não seja preciso nenhuma alteração no ambiente ou uso de equipamentos por parte do usuário. Fotos isoladas das mãos também são geralmente utilizadas para este fim, acompanhadas de padrões de distância e iluminação, atrapalhando na naturalidade das imagens manuseadas para o reconhecimento. O uso de redes neurais artificiais é fundamental neste trabalho, já que tal tecnologia tem sido bastante usada para processamento de imagens. Uma arquitetura previamente desenvolvida por outro trabalho, chamada de *Convolutional Pose Machines*, foi utilizada a fim de alcançar a solução do problema aqui proposto. Os resultados gerais foram bastante positivos, tendo sido obtida uma taxa média de acertos de 88.60%, chegando à conclusão de que a abordagem aqui proposta é válida e uma boa alternativa para o reconhecimento de gestos.

Palavras-Chave: Redes Neurais Convolucionais; Inteligência Artificial; Língua Brasileira de Sinais - Libras; Reconhecimento de gestos.

ABSTRACT

This work aims to develop an approach for gesture recognition, using the Brazilian sign language as a non-intrusive use case, through the use of artificial neural networks. The recognition of gestures is the focus of research in several works, one of the main difficulties is to make such recognition in a non-intrusive and natural way, so that there is no need to change the environment or the use of equipment by the user. Isolated photos of the hands are also generally used for this purpose, accompanied by patterns of distance and illumination, disrupting the naturalness of the images handled for recognition. The use of artificial neural networks is fundamental in this work, such technology has been widely used for image processing. An architecture previously developed by another work, called Convolutional Pose Machines, was used in order to achieve the solution of the problem proposed. The general results were quite positive, an average rate of correct answers of 88.60% was obtained, reaching the conclusion that the approach proposed is a valid and a good alternative for the recognition of gestures.

Keywords: Convolutional Neural Networks; Artificial intelligence; Brazilian Language of Signals - Libras; Gesture recognition.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

RNA	Redes Neurais Artificiais
RNC	Redes Neurais Convolucionais
CPM	Convolutional Pose Machine
SIFT	Scale-invariant Feature Transform
HOG	Histogram of Oriented Gradients

SUMÁRIO

1. INTRODUÇÃO

1.1. Cenário Técnico Científico

Segundo Kendon (1986), pessoas usam gestos frequentemente para se comunicar. Gestos são usados para tudo, de apontar para uma pessoa para chamar sua atenção até levar uma informação sobre espaço e características temporais.

Pesquisadores do mundo todo enfatizam na interação homem-computador (HCI, do inglês *Human-Computer Interaction*), que tem como foco conectar pessoas com máquinas, permitindo que usuários se comuniquem com qualquer dispositivo eletrônico. A interação entre máquina e computador era majoritariamente baseada em ações com periféricos, como mouse e teclado, que, apesar de acessíveis, são bastante limitados em questão de naturalidade e intuitividade.

Alguns meios de comunicação baseados em H2H (do inglês *Human to Human*) foram criados com o objetivo de uma interação mais natural, entre os mais populares estão o reconhecimento de fala e de gestos. A interação através de gestos mostrou-se bastante eficiente, principalmente para usuários menos experientes ou que possuem algum tipo de deficiência.

Reconhecimento de gestos pode ser conduzido com técnicas de visão de computador e processamento de imagens (SULTANA; RAJAPUSPHA, 2012). Segundo Kamal, Pareek e Sandhya (2013), reconhecimento de gestos é um fenômeno da engenharia e tecnologia de linguagem com o objetivo de interpretar gestos humanos através de algoritmos matemáticos. Segundo Kham e Ibraheem (2012 apud BASTOS; ANGELO; LOULA, 2015), recentemente, foi noticiado um grande aumento nos trabalhos relacionados ao reconhecimento de gestos. Muitos desses trabalhos utilizam linguagens de sinal como um aplicação prática para as técnicas desenvolvidas em seus trabalhos.

As linguagens de sinais, que são o principal meio de comunicação entre deficientes auditivos, não são universais, elas possuem sua própria estrutura de país para país e diferem até mesmo de região pra região de um mesmo país,

dependendo da cultura daquele determinado local para construir suas expressões ou regionalismos (ARAÚJO, 2010). A língua brasileira de sinais (Libras) é o meio de comunicação oficial empregado entre deficientes auditivos, e é tido como 2ª língua oficial no país.

Segundo Panwar (2012), nas últimas décadas, com a popularidade do reconhecimento de gestos, muitas técnicas foram desenvolvidas para o rastreamento e reconhecimento de vários gestos com as mãos. Em seu trabalho, são utilizadas características geométricas, como orientação, centro de massa, estado dos dedos (*status of fingers*), polegar (levantado ou dobrado) e a localização dos dedos na imagem, para o reconhecimento dos gestos com a mão. A abordagem desenvolvida é totalmente dependente dos parâmetros de forma da mão, não levando em consideração outros fatores, como cor da pele ou textura. Foram testadas 450 imagens, com uma taxa de acerto de 94%.

Já em Bastos, Angelo e Loula (2012), a abordagem desenvolvida consiste na combinação de dois descritores de forma, Histograma de Gradientes Orientados (HOG) e Momentos de Zernike (ZIM) para a extração de informações relacionadas a bordas e formas das mãos, técnicas de processamento de imagens, como detecção de pele para melhorar a significância dos dados extraídos com os descritores, e uma rede neural de dois estágios para o reconhecimento de gestos. Uma base contendo 9600 imagens representando 40 gestos diferentes em Libras foi criada para a avaliação desta abordagem, mostrando uma taxa de acertos de 96.77%.

Em Lin, Hsu e Chen (2014), foi utilizada uma Rede Neural Convolucional (RNC) para fazer o reconhecimento de sete gestos, antes do treinamento e classificação dos gestos com a RNC, foi aplicado o algoritmo *Gaussian Mixture model* (GMM) para detecção de pele, seguido de uma calibração da posição da mão que procura rotacionar a mão para uma posição neutra, que conseguiu uma taxa de acerto de 95.96%.

Já em Alani et al. (2018), que também faz a utilização de uma RNC para seu trabalho, 3750 imagens de sete gestos da Língua de Sinais do Peru (LSP) foram utilizados para o treinamento e classificação do método proposto, onde a imagem de entrada é transformada em escala de cinza, reduzindo o número de

parâmetros na primeira camada convolucional, para então enviar a imagem pré-processada para a RNC, que utiliza de uma arquitetura simples criada pelos próprios autores, nomeada *Adapted Deep Convolutional Neural Network* (ADCNN), o método proposto conseguiu uma taxa de acerto de 99.73%.

Uma limitação apontada nos trabalhos anteriores, é que ambos conseguem bons resultados em imagens centradas em torno da mão, onde o foco e objetivo da imagem é mostrar apenas as mãos, considerando distância e fundo padrão ou pouca variação na iluminação, logo, não conseguem reconhecer gestos em imagens que não atingem tais padrões, como imagens de meio corpo ou corpo inteiro em diferentes condições de fundo e iluminação.

1.2. Problemática

Existem diversas abordagens para a resolução de problemas relacionados ao reconhecimento de gestos, entre elas, técnicas de reconhecimento envolvendo extração de atributos e classificação são discutidas, cada uma com suas vantagens e desvantagens, e aplicadas em vários sistemas de reconhecimento.

Alguns marcadores ou dispositivos baseados em luvas podem ser utilizados para a aplicação de técnicas de reconhecimento de gestos com a mão, como em Mohandes (2012), mas o uso de equipamentos podem ser inviáveis pelo seu custo e desconforto para o usuário, logo, o reconhecimento de gestos de mão baseado em visão de computador é bem mais viável, pois pode ser realizado de forma natural, tendo como único requisito uma câmera.

Diante desse contexto, como realizar reconhecimento de gestos da Linguagem Brasileira de Sinais de maneira não intrusiva e sem a necessidade de tantas condições para sua aplicação (fundo padrão, iluminação, imagens isoladas da mão com o gesto e etc.)?

1.3. Justificativa

No âmbito de reconhecimento de gestos, o desenvolvimento de abordagens intrusivas torna fácil a detecção de informações ou configurações do

ambiente, entretanto, tal uso pode trazer mais custos e desconforto para o usuário. Abordagens não intrusivas, sem uso de luvas ou qualquer outro dispositivo do tipo, pode ser evitado com a intenção de simplificar a interação do usuário com o sistema. Deste modo, a proposta é desenvolver uma abordagem eficiente e não intrusiva para o reconhecimento de gestos com o uso de algoritmos de aprendizado utilizando como caso de uso a linguagem Libras.

1.4. Objetivos

1.4.1. Objetivo Geral

O objetivo geral deste trabalho consiste em desenvolver uma abordagem para o reconhecimento de gestos aplicado na linguagem de sinais Libras utilizando visão computacional.

1.4.2. Objetivos Específicos

Para alcançar o objetivo geral desta pesquisa, serão necessários atingir os seguintes objetivos específicos:

- Elaborar *dataset*;
- Aplicar o *dataset* no método proposto por Wei et al. para obtenção de pontos-chave referentes ao gesto reproduzido em cada imagem;
- Aplicar pontos-chave extraídos em uma Rede Neural Artificial para a classificação do gesto;
- Analisar os resultados comparando com os trabalhos de Alani et al. (2018), Lin, Hsu e Chen (2014) e Bastos, Angelo e Loula (2012).

1.5. Metodologia

O trabalho iniciou com uma pesquisa sobre trabalhos relacionados à abordagens para reconhecimento de gestos com a intenção de elaborar um método que poderia contribuir para este meio. Uma pesquisa exploratória à

procura de tecnologias de visão de computadores também foi realizada para a avaliação de uma abordagem não intrusiva.

Alguns algoritmos de pré-processamento de imagens foram avaliados e chegou-se a conclusão que o uso de estimação de pose acompanhado de uma RNA é uma alternativa viável para o reconhecimento de gestos. Foram então desenvolvidos o tema, contexto, problematização e objetivos do trabalho, a partir desse momento iniciam-se as etapas que serão necessárias para o desenvolvimento deste trabalho.

Pesquisas acerca de Libras foram feitas a fim de elaborar um *dataset* contendo os gestos correspondentes a todas as letras do alfabeto, que foram usados para o treinamento e teste para a RNA, e que também serão disponibilizados publicamente.

Com o *dataset* completo, os dados foram divididos em grupos de treino e de teste, o treinamento do algoritmo foi efetuado com os dados de treino, já com os dados de teste foi feita uma análise dos resultados. O objetivo foi obter dados acerca da média final de acertos para uma análise de comparação com trabalhos na área.

2. FUNDAMENTAÇÃO TEÓRICA

2.1. Redes Neurais Artificiais (RNA)

RNAs podem aprender padrões a partir de várias amostras de maneira supervisionada ou não supervisionada através de interações com os dados disponibilizados previamente e, a partir daí, atribuir um rótulo para amostras nunca antes vistas.

Segundo Mishra e Srivastava (2014) o conceito de RNA é basicamente introduzido a partir do tema da biologia, onde a rede neural desempenha um papel importante e fundamental no corpo humano. No corpo humano o trabalho é feito com a ajuda da rede neural, que é uma teia de milhões e milhões neurônios interconectados.

RNAs são a base em vários algoritmos para inúmeras resoluções de problemas, as aplicações são incontáveis, classificação (imagens ou dados brutos), detecção de objetos, segmentação em imagens, tomada de decisão (aplicado a jogos ou máquinas) e etc.

Nesta seção será apresentado a arquitetura geral de uma RNA, o algoritmo *Feedforward* e detalhes importantes do processos utilizados em seu decorrer e o gradiente descendente e sua aplicação na otimização de RNAs.

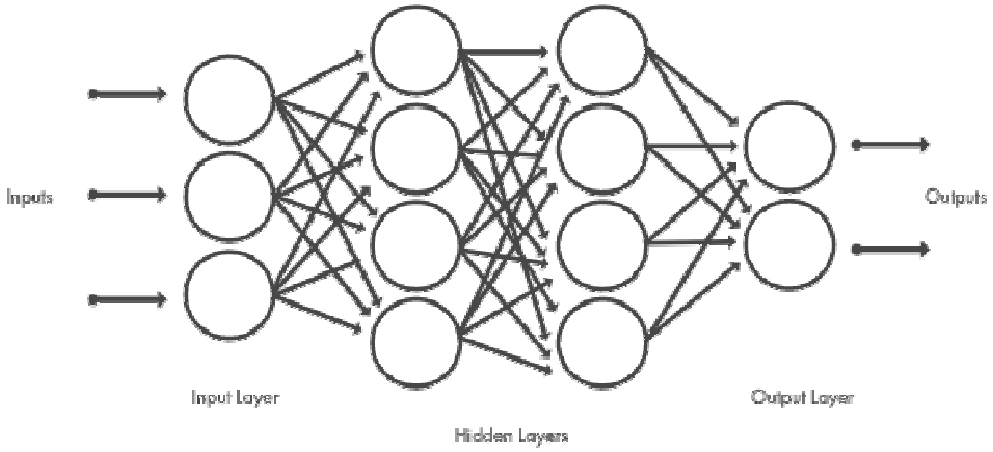
2.1.1. Arquitetura

De acordo com Schmidhuber (2014), Uma RNA padrão consiste em muitos processadores simples e conectados, chamados neurônios, cada um produzindo uma sequência de ativações. Neurônios da camada de entrada inicializam com os valores recebidos como entrada, outros neurônios são ativados através de conexões ponderadas de neurônios previamente ativos.

Uma RNA é formada por uma camada de entrada, *N* camadas intermediárias e uma camada de saída, onde todas as camadas são interligadas por pesos (do inglês, *weights*), que podem ser inicializados com valores

aleatórios. Cada camada é composta por vários neurônios, que é a unidade mais básica de uma RNA. Na figura 1, podemos observar como pode ser feita a estrutura geral de uma RNA.

Figura 1: Arquitetura geral de uma RNA



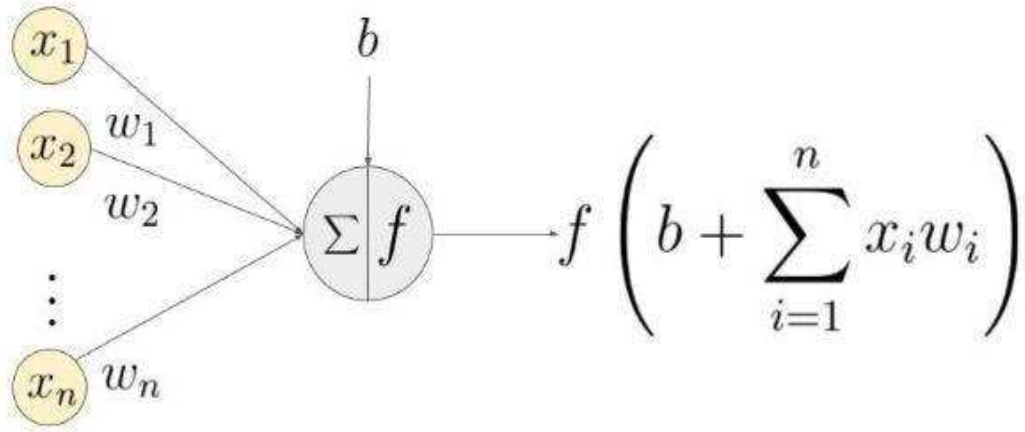
Fonte: https://www.mathworks.com/discovery/neural-network.html

2.1.2. Feedforward

Para obtermos um valor de saída, é necessário alimentar a RNA partir da camada de entrada até a camada de saída, essa alimentação é feita inicialmente com a combinação linear da camada de entrada e dos pesos que a interligam com a próxima camada, seguido da aplicação de uma função de ativação, este processo é feito de maneira contínua até que os valores da camada de saída sejam obtidos.

Na figura 2, podemos observar o processo de aquisição do valor para um neurônio da camada posterior a camada de entrada, onde cada neurônio recebe como valor da saída *y* da função de ativação *f(x)*, onde *x* é a somatória de todos os neurônios ligados ao neurônio em questão adicionado do bias. Tal processo é aplicado em todos os neurônios da RNA durante a execução do algoritmo *Feedforward*.

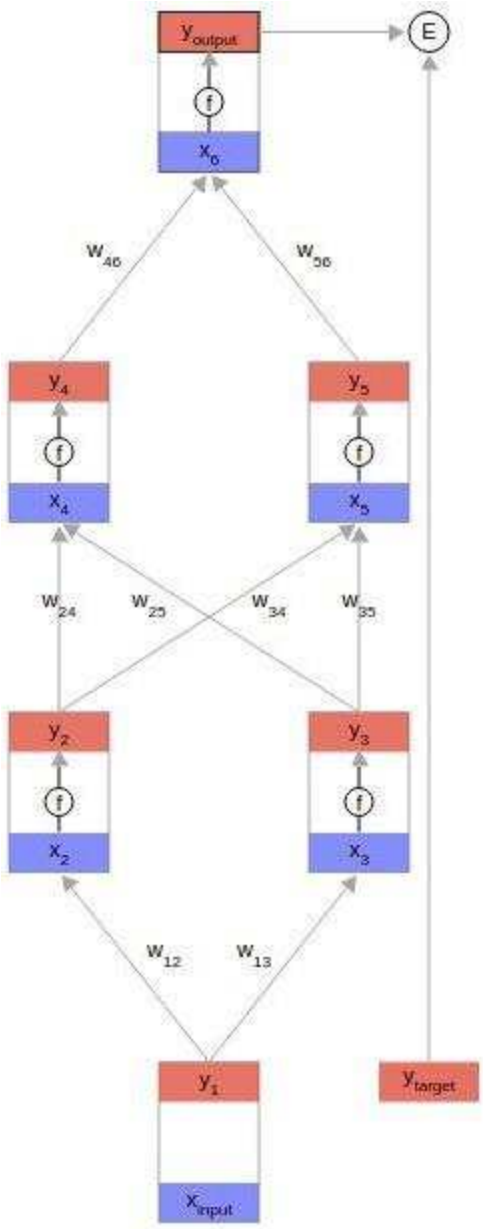
Figura 2: Combinação linear entre uma camada e seus pesos correspondentes, seguido da aplicação de uma função de ativação *f*



Fonte: https://www.learnopencv.com/understanding-feedforward-neural-networks (2017)

Na figura 3, uma RNA com duas camadas intermediárias é mostrada. Incluindo a camada de entrada, existem quatro camadas na estrutura. Ainda na figura 3, podemos observar como funciona o processo detalhado em um contexto geral, como são obtidos os valores de todos os neurônios da RNA até a obtenção de um resultado na camada de saída.

Figura 3: RNA



$$x_j = \sum_{i \in in(j)} w_{ij} \cdot y_i + b_j \quad (1)$$

$$y_j = f(x_j) \quad (2)$$

Fonte: https://google-developers.appspot.com/machine-learning/crash-course/backprop-scroll/ (2018)

2.1.2.1. Bias

O bias é um dos elementos mais importantes, não apenas para redes neurais, mas também para aprendizado de máquina no geral. Muitas vezes o bias pode ser adicionado ao criar um modelo de RNA qualquer.

Cada neurônio, depois da camada de entrada, também inclui mais um peso chamado bias associado a ele. O bias é um neurônio extra que guarda o valor um

e que não é influenciado pelas camadas anteriores, seu propósito é prover cada neurônio com uma constante treinável, ou seja, que pode ser otimizada durante o processo de otimização. Basicamente, se queremos montar uma RNA para aproximar a função f(x) = mx + c, o bias representaria o c da função e daria mais liberdade ao modelo para aprender os dados de treino.

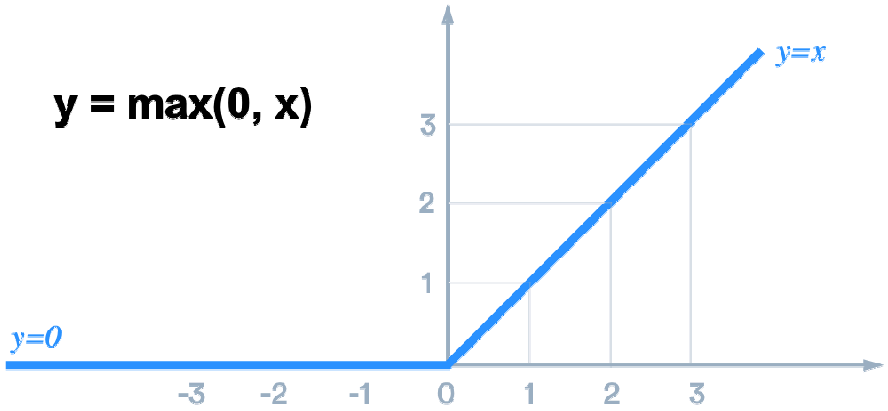
2.1.2.2. Funções de Ativação

Vários pesquisadores caracterizaram a função de ativação sob a qual RNAs podem atuar como aproximadores universais (LESHNO et al., 1992). Segundo Mhaskar e Micchelli (1994), funções de ativação adicionam não linearidade para a RNA, fazendo com que a rede possa se aproximar de praticamente qualquer função, sendo capaz de resolver problemas linearmente separáveis e não separáveis. Diversas funções de ativação são utilizadas na construção de RNAs, entre as mais populares estão sigmoid, tanh, ReLU, softmax.

Nas camadas intermediárias, a preferência é pelo uso de funções não saturadas, que evitam o problema do desaparecimento do gradiente e aceleram a convergência da RNA. Na camada de saída, a escolha depende do problema que está sendo resolvido, se o problema consiste em, por exemplo, retornar uma probabilidade para cada classe possível a partir de uma entrada *x*, a função de ativação softmax pode ser indicada.

Para a resolução do problema proposto, a RNA será implementada utilizando a função de ativação ReLU (Rectified Linear Unit) nas camadas intermediárias, por acelerar a convergência no ciclo de treinamento e, por ser uma função menos complexa em termos de operações, ter uma melhor performance quando comparado à funções como sigmoid ou tanh (KRIZHEVSKY et al., 2012).

Figura 4: Função de ativação ReLU



Fonte: https://www.tinymind.com/learn/terms/relu (2018)

A função softmax será aplicada na camada de saída, bastante utilizada em problemas de classificação, que nos retornará distribuição de probabilidades para cada sinal a partir do conjunto de coordenadas de entrada.

Figura 5: Função de ativação softmax

$$\text{softmax}(x)_i = \frac{\exp(x_i)}{\sum_j \exp(x_j)}$$

Fonte: https://datascienceplus.com/mnist-for-machine-learning-beginners-with-softmax-regression/ (2018)

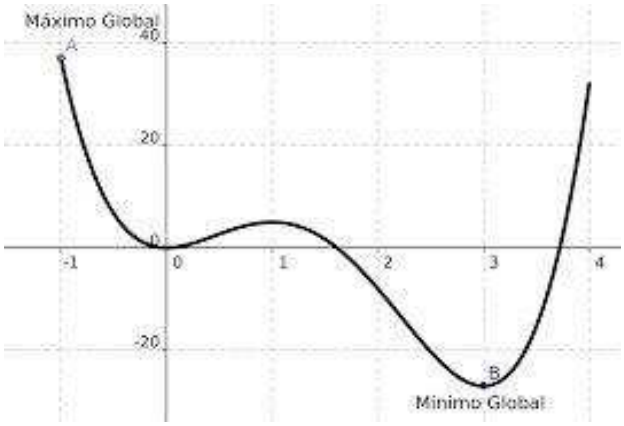
2.1.2.3. Funções de Perda

A partir da função de perda, o erro pode ser calculado, desta forma, avaliando o atual desempenho da RNA em relação aos dados de treino. O ideal é que o valor calculado pela função de perda esteja o mais próximo possível de zero, indicando que a rede está classificando de maneira correta todos os dados de treino.

O *y* da função de perda inicia em um local aleatório no gráfico, já que seus pesos são iniciados de forma aleatória, e o ciclo de treinamento (descrito na

sessão 2.1.3.1) vai atualizar esses pesos de forma que o *y* da função vá de encontro ao mínimo global. Na figura 6, podemos observar um exemplo de como um gráfico de função de perda pode se comportar.

Figura 6: Máximo e mínimo global de uma função de perda



Fonte: https://www.ime.unicamp.br/~valle/Teaching/MA111/Aula13.pdf (nao sei)

Uma RNA entra em convergência quando alcança seu mínimo global, que nem sempre pode ser atingido, existem casos em que a função de perda permanece bloqueada em mínimos locais, ou até mesmo, não chegam no mínimo global quando seu *y* inicialmente está muito longe do mínimo global. Alguns fatores influenciam bastante na entrada da RNA em convergência e também na sua velocidade, como as funções de ativação utilizadas e o método de otimização empregado.

2.1.3. Gradiente Descendente

Segundo Goh (1995), a rede neural aprende modificando os pesos dos neurônios em resposta aos erros entre os valores de saída da RNA e os valores de saída reais. Isto é feito através do gradiente descendente da função de perda, onde a mudança nos pesos é dada pelo negativo da derivada da função de perda em relação a cada peso. O intuito desta técnica é minimizar o valor da função de perda para que fique o mais próximo possível de zero, quando a rede entra em

convergência. É importante ressaltar que esta optimização só pode ser aplicada se as funções de ativação e de perda forem diferenciáveis.

Em Ruder (2016), o gradiente descendente é apontado como um dos algoritmos mais populares para realizar a otimização e de longe a maneira mais comum de otimizar as redes neurais.

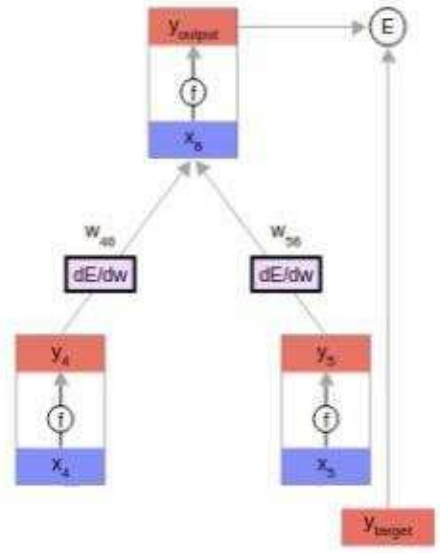
Ainda em Ruder, o gradiente é uma maneira de minimizar uma função objetiva J parametrizada por um modelo de parâmetros $W \in \mathrm{R}$ atualizando os parâmetros na direção oposta do gradiente da função objetiva ∇W em relação aos parâmetros. A taxa de aprendizagem α determina o tamanho do passos que é tomado para alcançar um mínimo (local).

$$\nabla w_j = \frac{\partial J}{\partial w_j} \qquad (3)$$

$$w_j = w_j - \alpha \cdot \nabla w_j \qquad (4)$$

Quando se trata de otimização de RNAs, a função objetiva J é função de perda E, e o gradiente ∇W é obtido através da derivada E' da função em relação aos parâmetros W. Será usada a regra da cadeia para dividir a equação em partes mais gerenciáveis, que nos permite trabalhar com derivadas nas camadas da RNA.

Figura 7: Gradiente descendente em RNA



$$\nabla w_{ij} = \frac{\partial E}{\partial w_{ij}} \quad (5)$$

Aplicando a regra da cadeia, obtemos

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial y_i} \cdot \frac{\partial y_i}{\partial x_j} \cdot \frac{\partial x_j}{\partial w_{ij}} \quad (6)$$

Depois, atualizamos os pesos

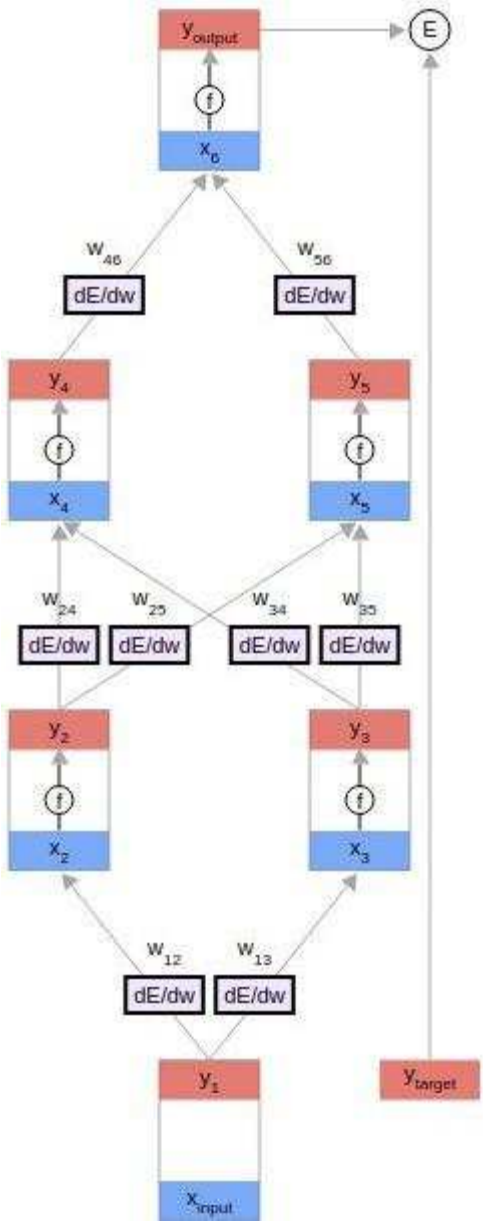
$$w_{ij} = w_{ij} - \alpha \cdot \nabla w_{ij} \quad (7)$$

Fonte: https://google-developers.appspot.com/machine-learning/crash-course/backprop-scroll/ (2018)

2.1.3.1. Backpropagation

Backpropagation é a implementação do gradiente descendente em uma RNA. Já que a mesma regra se aplica em cada camada da rede neural de forma recursiva, o erro pode ser propagado de forma reversa, a partir da camada de saída, até a camada de entrada, atualizando todos os pesos.

Figura 8: Gradiente descendente em RNA



Fonte: https://google-developers.appspot.com/machine-learning/crash-course/backprop-scroll/ (2018)

Alguns problemas podem atrapalhar ou prolongar o convergência da RNA, como a utilização de uma taxa de aprendizado muito alta ou baixa, ou o desaparecimento/explosão dos gradientes, que acontece quando os gradientes desaparecem/aumentam à medida que vão sendo calculados.

Este processo é chamado de ciclo de treinamento, ou época (do inglês, *epoch*). O treinamento da rede é dado repetindo a época várias vezes com os dados de treinamento, onde a atualização dos pesos é feita no fim de cada ciclo, procurando sempre reduzir o valor da função de perda. O pseudocódigo (WERBOS, 1994) para uma RNA de três camadas é apresentado a seguir.

Inicializar pesos da RNA (frequentemente, pequenos valores)
faça :
    para cada exemplo de treino, nomeado ex:
        predição = saída-da-RNA(RNA, ex) // feedforward
        saída-real = saída-real(ex)
        compute o erro (predição - saída-real) nas unidades de saída // ou qualquer outra função de perda
        compute ∇W para todos os pesos da camada intermediária para a camada de saída
        compute ∇W para todos os pesos da camada intermediária à camada oculta
        atualize os pesos da RNA
até que todos os exemplos classificados corretamente ou outro critério de parada satisfeito

2.2. Redes Neurais Convolucionais

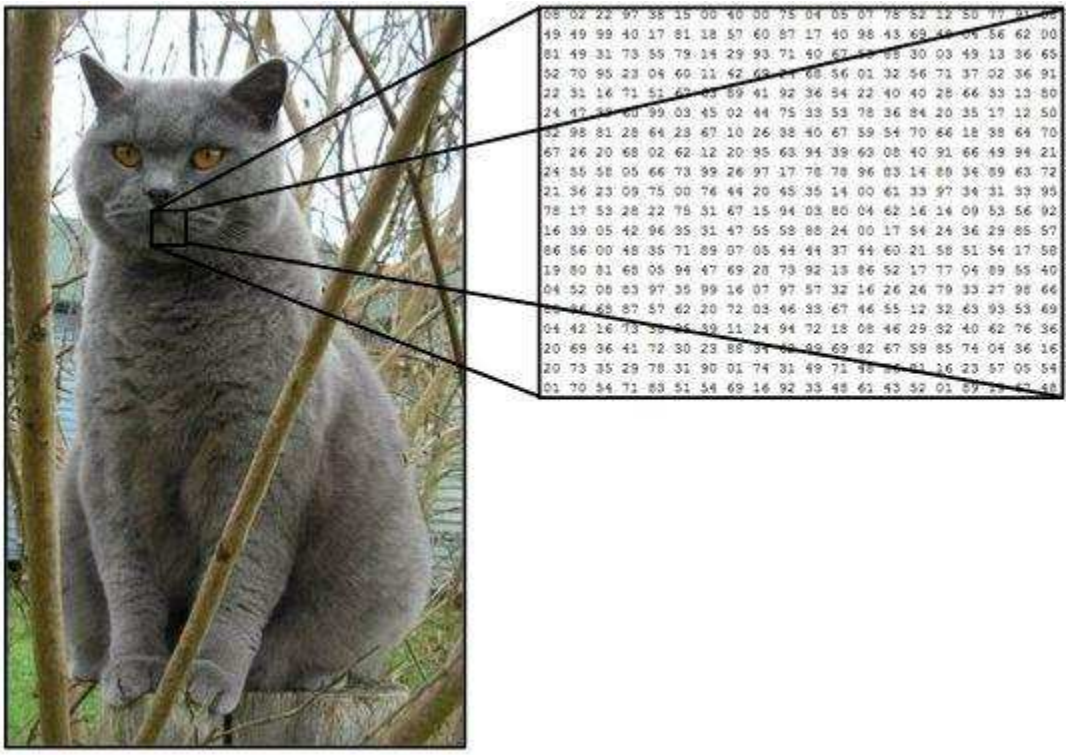
Rede Neural Convolucional (RNC) é um tipo de RNA bastante eficiente para processamento de imagens, muito utilizado para reconhecimento de padrões e processamento de imagens.

Segundo Liu et al. (2015), uma RNC é uma RNA que tem um design especial para identificação de padrões em imagens bidimensionais. Possui algumas camadas adicionais: camada de entrada, camada de convolução, camada de sampling e camada de saída. Em uma RNC, existem dois processos principais, convolução e sampling.

Para um computador, uma imagem é representada por uma matriz de três dimensões, altura x largura x canais, no caso de imagens que adotam o sistema

RGB (vermelho, verde e azul, do inglês *red, green and blue*) o número de canais é três, em imagens preto e branco esse valor é representado por um. Cada elemento da matriz é composto por números entre 0 e 255. É nessa representação onde as operações de convolução e downsampling são realizadas.

Figura 9: Representação de uma imagem para o computador



fonte: http://cs231n.github.io/classification/ (2018)

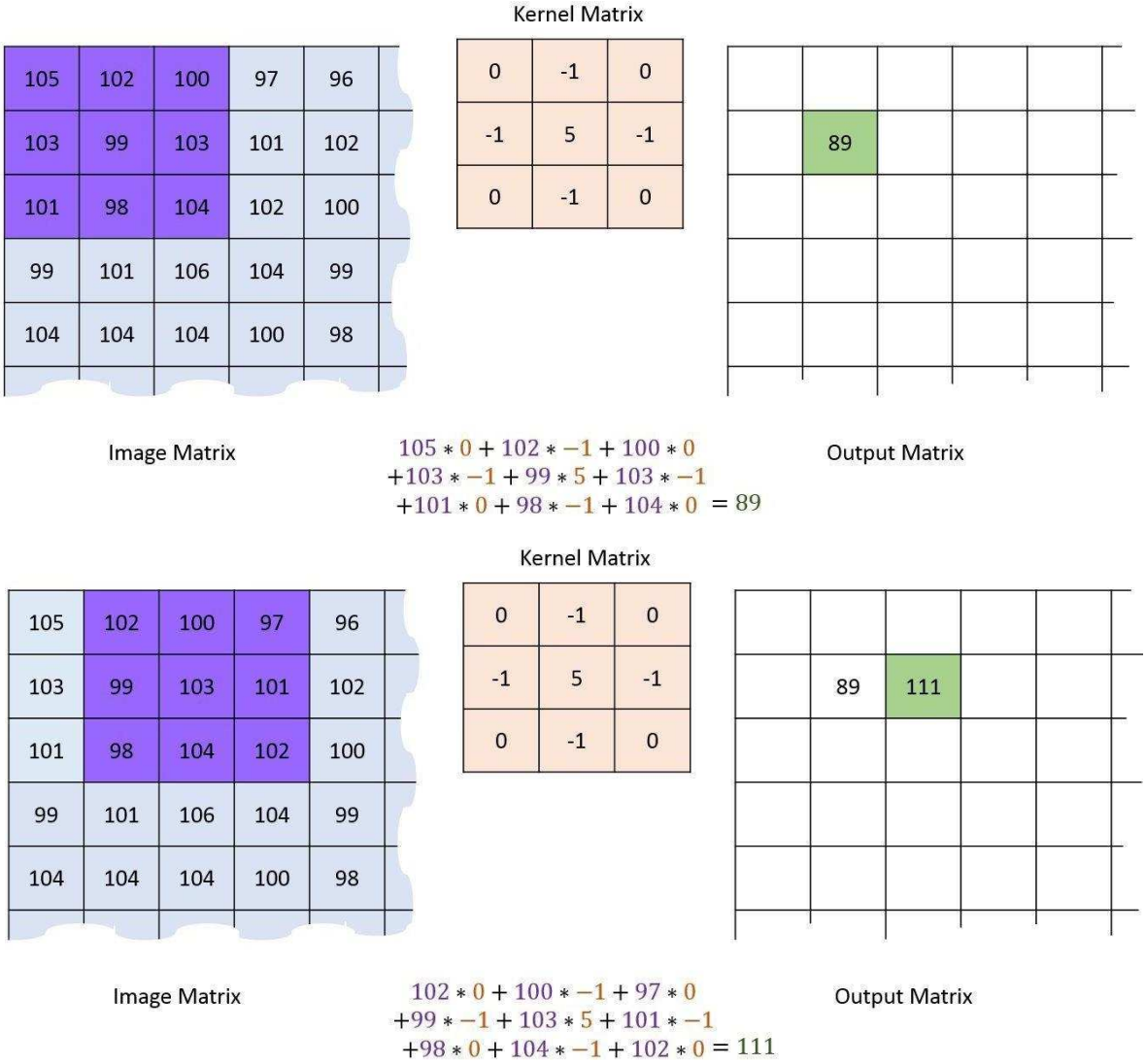
2.2.1. Convolução

O processo de convolução é dado por uma simples operação matemática de fusão entre um filtro, ou kernel, e uma imagem. Na RNC esses filtros são equivalentes aos pesos da RNA, ou seja, serão modificados em cada ciclo de treinamento, através do backpropagation, a fim de reduzir o valor obtido pela função de perda. O resultado dessa operação é uma nova imagem com número de canais equivalente ao número de filtros aplicados, chamado *feature map*.

Em processamento de imagens, um kernel é uma pequena matriz de tamanho *N x N*. É utilizado para aplicar filtros e transformações em imagens por

meio da convolução. A figura 10 demonstra com detalhes o processo de convolução, onde o kernel percorre toda a imagem por *S* pixels, e para cada localidade do kernel, uma multiplicação de matrizes é feita, e a somatória dessa multiplicação resulta em um pixel na nova imagem de saída.

Figura 10: Processo de convolução, kernel 3x3, S=1



fonte:
http://machinelearninguru.com/computer_vision/basics/convolution/image_convolution_1.html (2018)

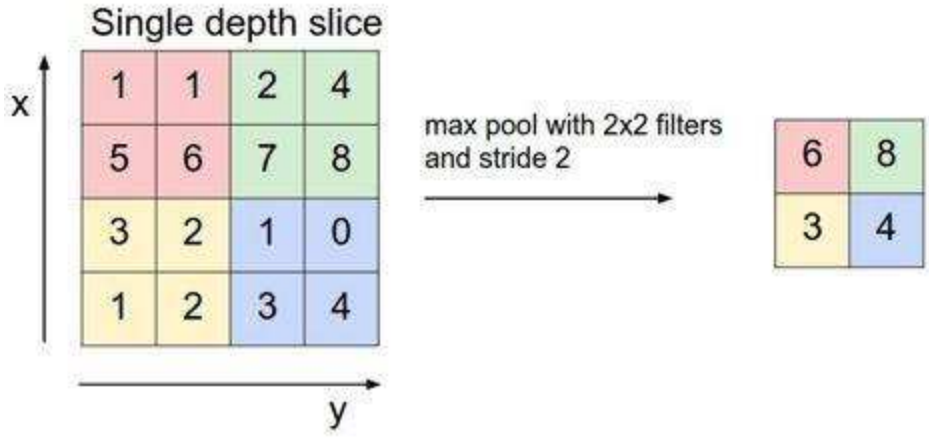
2.2.2. Downsampling

Segundo Jaswal, Vishvanathan e Soman (2014), a camada de downsampling vem depois da camada convolucional. O objetivo dessa camada é diminuir o tamanho do *feature map*, reduzindo o número de parâmetros para a RNC. A camada de downsampling preserva as informações relativas entre os recursos e não a relação exata.

Em uma RNC, o processo de downsampling é feito, de preferência, por *pooling*, entretanto, convoluções com *strides* maiores também podem ser usados para tal objetivo. O processo de *pooling* requer que uma janela de tamanho *N x N* deslize pela imagem por *S* pixels, assim como no processo de convolução. Em cada janela, podemos escolher o maior pixel (*max pooling*), o menor pixel (*min pooling*) ou a média dos pixels (*mean pooling*), entre outros métodos.

Na figura 11, podemos observar com detalhes o processo de *maxplooling*, onde cada localidade da janela de tamanho 2x2 é indicada por uma cor diferente na imagem de entrada.

Figura 11: Processo de *max pooling*, N=2, S=2



fonte: http://cs231n.github.io/convolutional-networks/ (2018)

Depois do processo de *pooling*, o feature map resultante deve ser bem menor que o anterior, as novas dimensões de altura e largura podem ser calculadas da seguinte maneira:

$$Nova\ altura = \frac{(Altura\ anterior - N)}{Stride - 1} \quad (8)$$

$$Nova\ largura = \frac{(Largura\ anterior - N)}{Stride - 1} \quad (9)$$

2.2.3. Design e Arquitetura

RNCs são basicamente formadas por três tipos de camadas, já definidas neste trabalho, são elas: Camadas convolucionais, camadas de *pooling* (*downsampling*) e RNA, a combinação e uso repetitivo destas camadas formam diferentes tipos de arquiteturas para a RNC.

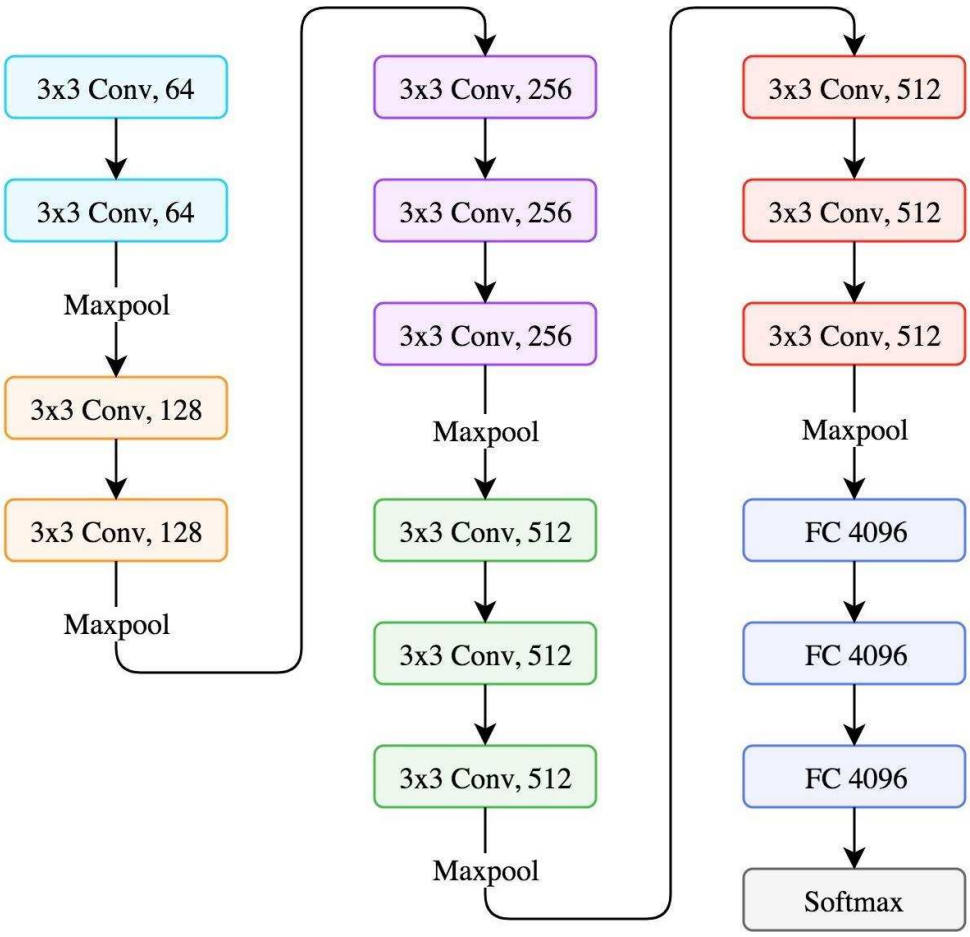
Em Ahire (2018), pode ser encontrado um exemplo de uma simples RNC para classificação, com a arquitetura [INPUT - CONV - RELU - POOL - RNA]. Em mais detalhes:

- INPUT [32x32x3] mantém os valores brutos de pixel da imagem, neste caso uma imagem de largura 32, altura 32 e com três canais de cores R, G, B.
- A camada CONV calcula a saída de neurônios conectados a regiões locais na entrada, cada um computando o produto entre seus pesos e uma pequena região à qual estão conectados no volume de entrada. Isso pode resultar em volume como [32x32x12] se decidirmos usar 12 filtros.
- A camada RELU aplica uma função de ativação elementar, como o limiar máximo (0, x) em zero. Isso deixa o tamanho do volume inalterado ([32x32x12]).
- A camada POOL executa uma operação de redução de resolução ao longo das dimensões espaciais (largura, altura), resultando em um menor volume, como [16x16x12].
- A camada RNA calcula as pontuações da classe, resultando em um volume de tamanho [1x1x10], em que cada um dos 10 números corresponde a um escore de classe. Tal como acontece com as

redes neurais comuns e como o nome indica, cada neurônio nesta camada será conectado a todos os números do volume anterior.

Existem arquiteturas bem mais robustas (VGG, Resnet, Inception e etc.) e com resultados melhores para datasets mais diversificados, como COCO dataset ou ImageNet. Estruturas como essas são comumente utilizadas em algoritmos de classificação, detecção, segmentação e estimação de pose. Abaixo temos a ilustração de uma versão da arquitetura VGG.

Figura 12: Arquitetura VGG-16

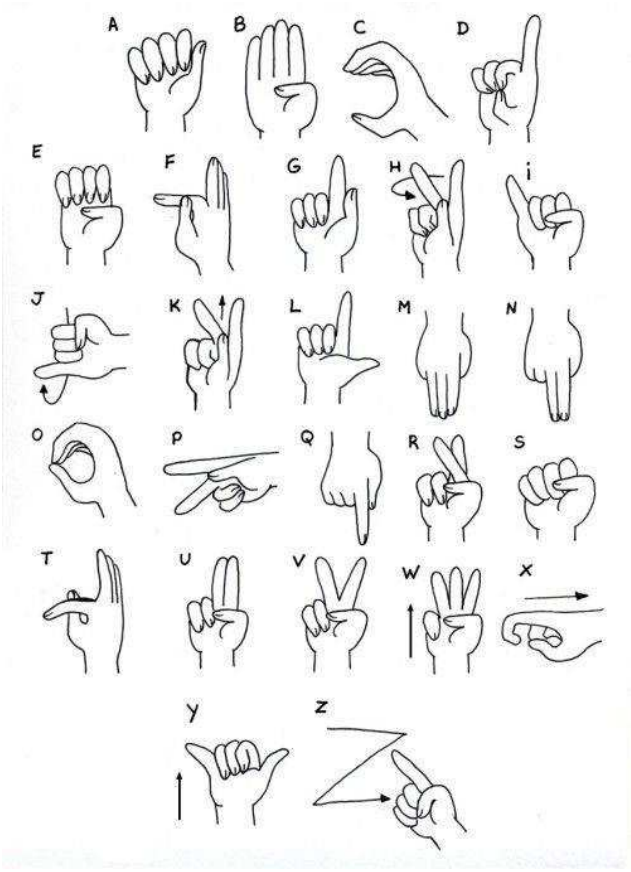


Fonte: https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2 (2017)

2.3. Libras

As línguas de sinais são utilizadas pela maioria das pessoas surdas no mundo. No Brasil, existem duas línguas de sinais: a Língua Kaapor e a Língua Brasileira de Sinais (Libras), que é utilizada nos centros urbanos. A língua portuguesa, no caso dos surdos brasileiros, é considerada uma segunda língua (UZAN; OLIVEIRA; LEON, 2008). A figura 4 ilustra o alfabeto de Libras.

Figura 13: Alfabeto de Libras



Fonte: http://historiadodesign-blog.tumblr.com/post/3915299150/alfabeto-de-libras-o-alfabeto-de-libras-l%C3%ADngua (2011)

2.4. Visão de computadores

Segundo Kaiser (2017), visão de computadores é uma área de inteligência artificial e ciências da computação que tem por objetivo dar a computadores um conhecimento visual do mundo. O objetivo de visão de computadores é emular a

visão humana utilizando imagens digitais através de três processos principais, aquisição de imagens, processamento de imagens e análise e compreensão.

O uso de descritores é muito comum no processamento de imagens. Segundo Pham (2016), descritores tem um papel fundamental em vários problemas de visão de computadores, como na comparação de imagens e reconhecimento de objetos. Descritores são usados para extrair características únicas de uma imagem, como uma assinatura, a utilização dessa assinatura para posterior localização em uma outra imagem é a solução mais popular para o problema de correspondência de imagens. Dois descritores bastante populares são descritos abaixo.

2.4.1. Histogram of Oriented Gradients (HOG)

Segundo Gritti et al. (2008), a idéia essencial do descritor HOG é descrever a aparência e forma de um objeto com base na distribuição de intensidade de gradientes. O método utiliza histogramas locais normalizados a partir da orientação de gradientes como características de uma imagem. Na prática, as características do HOG são extraídas dividindo a imagem em pequenas regiões chamadas “células”. Para cada célula, um histograma de 1-D é computado levando em conta as direções e intensidade dos gradientes sobre os pixels da célula. A combinação dos histogramas formam uma representação final da imagem.

2.4.2. Scale-invariant Feature Transform (SIFT)

Lowe (1999) propôs um novo método para geração de características em imagens, que combina um detector de região invariante de escala e um descritor baseado na distribuição de gradiente nas regiões detectadas. Sua obtenção pode ser feita a partir dos seguintes passos:

Detecção de extremos (máximos e mínimos): No primeiro estágio, o algoritmo procura em todas as escalas e localizações das imagens. A função DoG (do inglês *difference-of-gaussian*) está sendo usada para identificar pontos de interesse invariáveis à escala e rotação.

Localização dos pontos-chave: Em cada ponto de interesse, um modelo detalhado é ajustado para determinar a localização e escala. Pontos-chave são selecionados baseados nas medidas de estabilidade.

Atribuição de orientação: Uma ou mais orientações são designadas para cada ponto-chave com base nas direções do gradiente de imagem local. Todas as futuras operações são realizadas em dados da imagem transformados em relação a orientação designada, escala e localização de cada característica. Desta maneira se obtém invariância a estas transformações.

Descritor dos pontos-chave: Os gradientes da imagem local são medidos na escala selecionada na região em torno de cada ponto-chave. São transformados em uma representação que permite níveis significativos de distorção da forma local e mudança na iluminação.

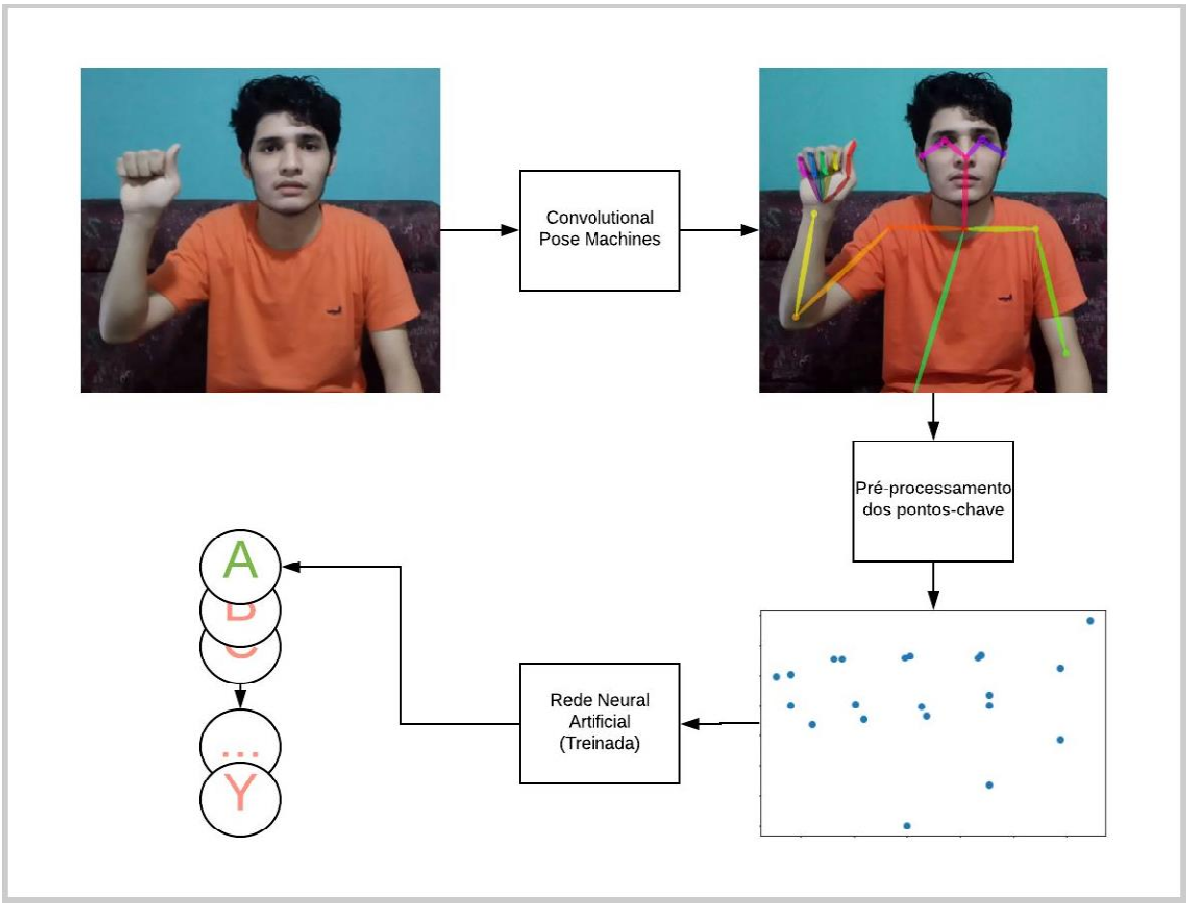
3. DESENVOLVIMENTO

Neste capítulo, serão apresentados detalhes do processo de desenvolvimento, de modo que fique claro o método apresentado durante este trabalho.

Foram considerados vinte e três sinais de Libras, alguns sinais não foram levados em consideração por serem dinâmicos, o foco deste trabalho foi o reconhecimento de sinais que não precisam de movimentos para serem interpretados, são eles: A, B, C, D, E, F, G, I, K, L, M, N, O, P, Q, R, S, T, U, V, W, X e Y.

A abordagem proposta para o reconhecimento dos gestos tem quatro passos fundamentais, a extração de vinte e um pontos-chave da mão, pré-processamento dos pontos, o treinamento de uma RNA e a aplicação dos pontos-chave no algoritmo para classificação. O processo de classificação pode ser visualizado na figura 14.

Figura 14: Processo de classificação do método proposto



Fonte: Próprio autor

3.1. Estimativa de Pose da mão

A estimativa de pose é um passo fundamental no processo, nesse procedimento, o sistema recebe como entrada uma imagem de corpo inteiro ou meio corpo de uma pessoa reproduzindo algum sinal de Libras, a partir desta entrada, vinte e um pontos-chave (coordenadas de um plano cartesiano) são extraídos das mãos utilizando o método proposto por Wei et al. (2016), *Convolutional Pose Machines* (CPM). Para este trabalho, apenas a mão direita está sendo levada em consideração.

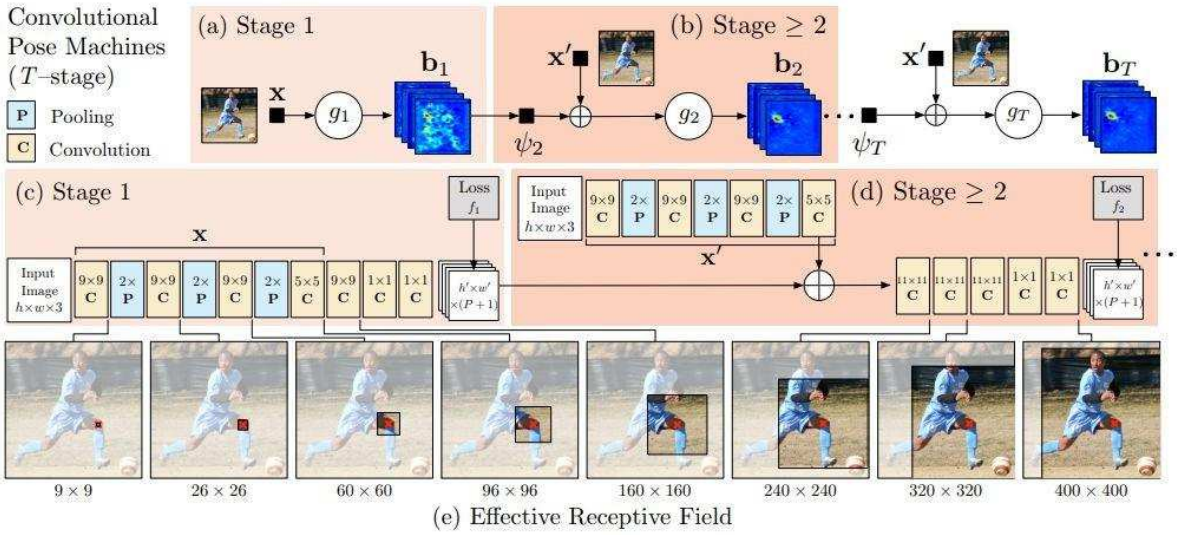
Para a aplicação do CPM, foi utilizada a implementação desenvolvida por seus próprios autores, nomeada *OpenPose*. O *OpenPose* foi desenvolvido utilizando a linguagem c++, mas implementações em outras linguagens podem ser facilmente encontradas no site de versionamento e controle de projetos *Github*.

3.1.1. Convolutional Pose Machines

No trabalho de Wei et al. (2016) foi proposto um método, nomeado *Convolutional Pose Machines* (CPM) para estimar a pose a partir de uma imagem 2D de maneira não intrusiva, ou seja, sem o uso dispositivos ou alteração do ambiente. O método foi baseado no trabalho de Ramakrishna et al. (2014), *Pose Machines*, combinando o mesmo com as vantagens do uso de arquiteturas convolucionais: a capacidade de aprender representações de características para ambos imagem e contexto espacial diretamente dos dados; uma arquitetura diferenciável que permite o treinamento com backpropagation; e a capacidade de lidar eficientemente grandes conjuntos de dados de treinamento.

O método consiste em uma série de redes convolucionais sequenciais. Em cada estágio, a rede neural recebe como entrada um mapa de confiança (*belief maps*) fornecido como saída na rede anterior. O CPM é treinado para aprender características das imagens e modelos espaciais dependentes de imagem. Cada passo corresponde a um refinamento sequencial. No primeiro estágio, a rede neural convolucional é aplicada para obter o mapa de confiança a partir de uma evidência local usando um pequeno campo receptivo. Sucessivos estágios usam múltiplas camadas para alcançar largos campos receptivos para capturar complexas correlações entre as partes. O modelo pode ser treinado a partir do zero e tem como saída dezoito pontos-chave do corpo, que são coordenadas cartesianas bidimensionais.

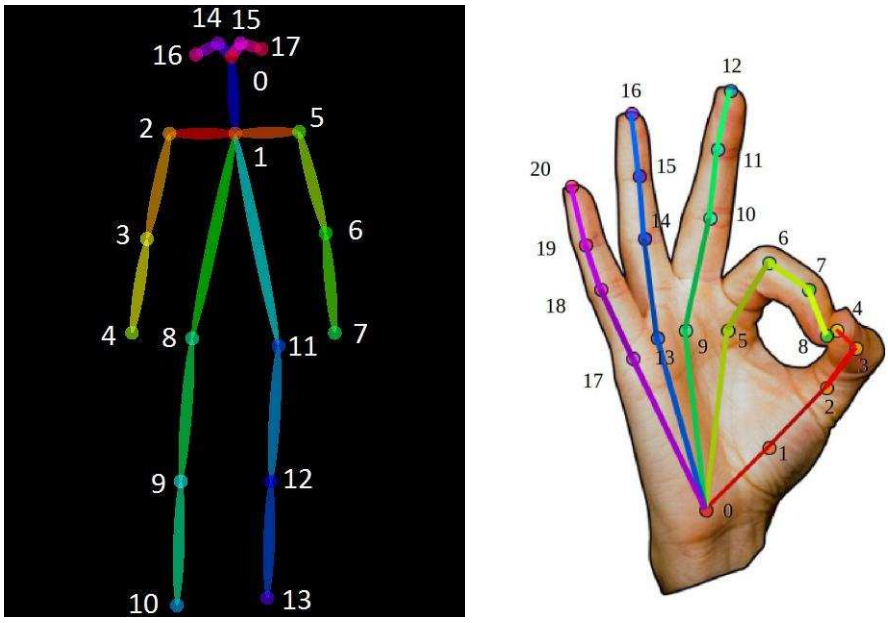
Figura 15: Arquitetura CPM



Fonte: https://arxiv.org/pdf/1602.00134.pdf (2016)

Na figura 15, é mostrada a arquitetura convolucional e os campos receptivos em camadas para um CPM com qualquer T estágios. O método *Pose Machine* é mostrado nas inserções (a) e (b), e as redes convolucionais correspondentes são mostradas nas inserções (c) e (d). Inserções (a) e (c) mostram que a arquitetura, no primeiro estágio, opera apenas na imagem entrada. As inserções (b) e (d) mostram a arquitetura das etapas subseqüentes, que operam tanto na imagem de entrada como nos mapas de confiança dos estágios anteriores. As arquiteturas em (b) e (d) são repetidas para todos os estágios subsequentes (2 a T). A rede é supervisionada localmente após cada estágio, usando uma camada intermediária que evita o desaparecimento do gradiente durante o treinamento. Abaixo, na inserção (e), é mostrado o campo receptivo efetivo em uma imagem (centrada no joelho esquerdo) da arquitetura, onde o grande campo receptivo permite ao modelo capturar dependências espaciais de longo alcance, como aquelas entre a cabeça e os joelhos. O processo tem como saída vinte e um pontos-chave, conforme mostrado na Figura 16.

Figura 16: Estimação de Pose do corpo e da mão

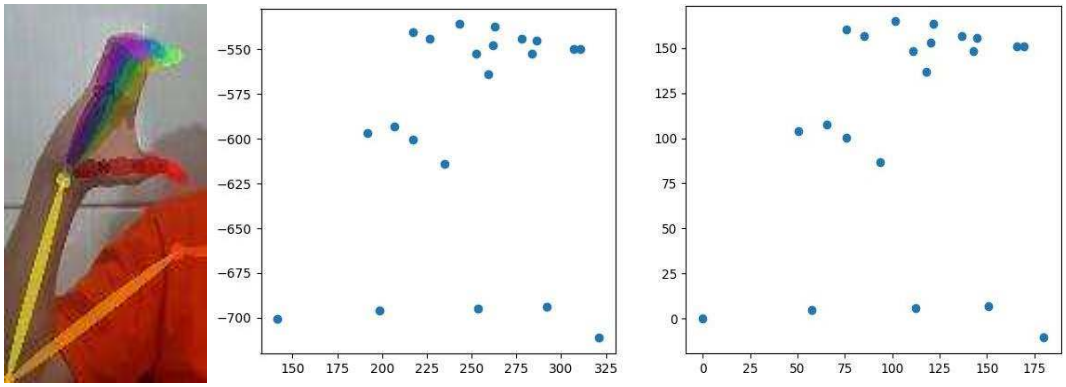


Fonte: https://github.com/CMU-Perceptual-Computing-Lab/openpose/blob/master/doc/output.md (2017).

3.2. Pré-processamento dos pontos-chave

Depois da extração, é feita uma translação dos pontos para um ponto inicial (0, 0). Este processo leva a RNA a entrar em convergência mais rápido, reduzindo bastante o tempo do processo de treinamento. Na figura 17, é ilustrado o processo de translação nos

Figura 17: Processo de translação



Fonte: Próprio autor

3.3. Estrutura da RNA

Para o desenvolvimento da RNA, foi utilizada a biblioteca *PyTorch*, que tem ganhado bastante popularidade entre pesquisadores pela sua facilidade de uso e transparência nos detalhes dos vários processos que ocorrem no ciclo de uma RNA (*Feedforward*, *Backpropagation* e etc.).

Foi desenvolvida uma RNA com quarenta e dois neurônios na camada de entrada, que recebe os vinte e um pontos-chave extraídos pelo *OpenPose*, um total de quarenta e dois pontos flutuantes. Foram utilizadas duas camadas intermediárias com 30 neurônios cada, por fim, na camada de saída, foram utilizados vinte e três neurônios, que representam as possíveis saídas.

Nas camadas intermediárias, foi utilizada a função de ativação ReLU, já na camada de saída, a função Log de Softmax foi empregada, nos dando como resultado o logaritmo das probabilidades das vinte e três possíveis saídas, o neurônio com maior valor é apontado como escolhido.

A função de perda aderida foi o negativo do log probabilidade (do inglês, *Negative Log-likelihood*), ilustrada na figura 18.

Figura 18: Histórico da função de perda durante o treinamento

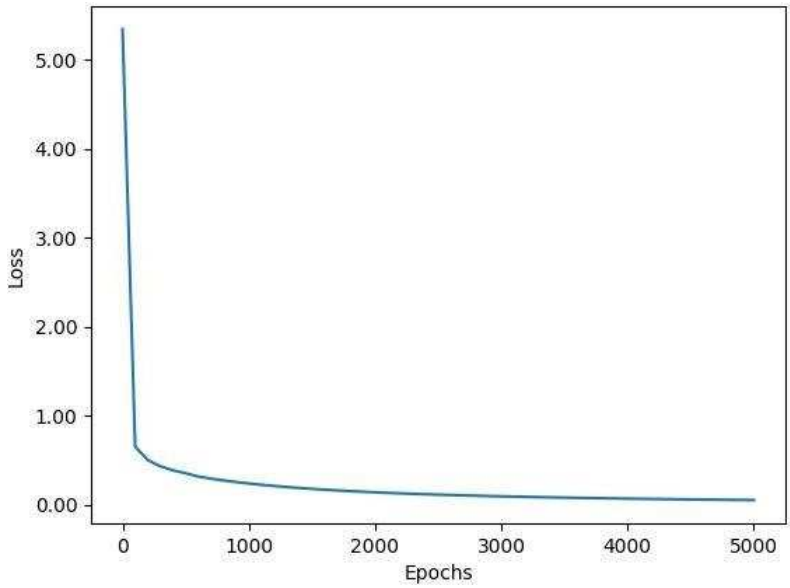
$$\mathrm{NLL}(\theta) = -\sum_{i=1}^{N} \log p(y_i \mid \mathbf{x}_i, \theta)$$

Fonte: https://www.quantstart.com/articles/Maximum-Likelihood-Estimation-for-Linear-Regression (sem data)

3.4. Treinamento

Antes que a RNA possa fazer alguma predição, o algoritmo precisa ser alimentado com amostras já classificadas, para isso, cinco mil épocas foram suficiente para a RNA entrar em convergência. O algoritmo SGD foi utilizado para a otimização da RNA.

Figura 18: Histórico da função de perda durante o treinamento



Fonte: Próprio autor

3.5. Classificação

O método recebe como entrada uma imagem e, aplicando os processos de estimação de pose e pré-processamento, é feita a extração dos pontos-chave, que então são utilizados como entrada na RNA, definida previamente, já treinada, a camada de saída então classifica o gesto que foi reproduzido na imagem de entrada.

4. RESULTADOS

O método proposto foi aplicado, 5077 imagens foram utilizadas para este trabalho, das quais 80% foram utilizadas para treinamento e as outras 20% para teste. As imagens foram reproduzidas pelo próprio autor, dada a falta de qualquer *dataset* de Libras com condições específicas para o estudo de caso aqui aplicado (gestos com corpo inteiro ou meio corpo).

Para a elaboração do *dataset*, vinte e três vídeos foram feitos, um para cada gesto, em que o modelo gesticulou sinais de várias maneiras e ângulos possíveis, que foram então quebrados em quadros, formando o *dataset* utilizado neste trabalho. É importante ressaltar que as imagens perdem um pouco de sua qualidade quando obtidas a partir de vídeos, logo, podemos conseguir resultados melhores se o *dataset* for composto por fotografias.

Os resultados da presente abordagem foram calculados em termos da taxa de acerto, foi obtida uma taxa média de 88.60%, alguns gestos conseguiram taxas bem melhores que outras. O resultado pode ser consultado na tabela 1.

Tabela 1: Taxa de reconhecimento por gesto

Gesto	Taxa de Acerto
A	98%
B	83%
C	96%
D	87%
E	96%
F	80%
G	94%
I	90%
K	92%
L	95%

M	92%
N	94%
O	81%
P	100%
Q	100%
R	85%
S	95%
T	89%
U	85%
V	90%
W	78%
X	89%
Y	50%

Fonte: Próprio autor

O problema de reconhecimento de gestos de maneira não intrusiva e sem restrições de fundo com meio corpo ou corpo inteiro foi resolvido, foi alcançada uma taxa de acertos satisfatória, comparando com os resultados obtidos em Alani et al. (2018), Lin, Hsu e Chen (2014) e Bastos, Angelo e Loula (2012), que conseguiram 99,73%, 95,96%, 96,77%, consecutivamente.

5. CONSIDERAÇÕES FINAIS

O objetivo do presente estudo, foi a proposta de uma abordagem para reconhecimento de gestos aplicado à Língua Brasileira de Sinais (Libras) com padrões mais aceitáveis quando se procura fazer o reconhecimento de maneira não intrusiva e de forma natural (imagens de meio corpo ou corpo inteiro, sem fundos padrões), a abordagem inclui o uso do *framework Convolutional Pose Machines* para a obtenção de pontos-chave da mão, pré-processamento dos pontos extraídos, seguido de uma Rede Neural Artificial para a classificação dos gestos.

Foi levantado um *dataset* para a avaliação da abordagem proposta neste trabalho, retiradas a partir de quadros de vídeos produzidos pelo autor. Como os resultados sugeriram, foram obtidos resultados satisfatórios, atingindo, em média, 88.60% de taxa de acerto, demonstrando ser uma boa alternativa para reconhecimento de gestos quando comparado com os resultados obtidos em Alani et al. (2018), Lin, Hsu e Chen (2014) e Bastos, Angelo e Loula (2012), que conseguiram 99.73%, 95.96%, 96.77%, consecutivamente.

Para melhorias e trabalhos futuros, pretende-se fazer o reconhecimento de gestos com movimentos, que representam maior parte da comunicação em Libras. Dada a importância do assunto, também é esperado fazer o reconhecimento dos gestos em tempo real, acompanhado de uma plataforma para aplicação da abordagem.

6. REFERÊNCIAS

A. SULTANA, T. RAJAPUSPHA. A vision based Hand gesture recognition. **IJCSET**, 2012.

BASTOS, Igor L.o.; ANGELO, Michele F. and LOULA, Angelo C. Recognition of Static Gestures Applied to Brazilian Sign Language (Libras). **2015 28th SIBGRAPI Conference on Graphics, Patterns and Images**, 2015.

GRITTI, Tommaso; SHAN, Caifeng; JEANNE, Vincent; *et al.* Local features based facial expression recognition with face registration errors. **2008 8th IEEE International Conference on Automatic Face and Gesture Recognition**, 2008.

VYAS, Kamal; PAREEK, Amita, VYAS, Sandhya. Gesture Recognition and Control. **International Journal on Recent and Innovation Trends in Computing and Communication**, 2013.

KENDON, Adam. **Special issue: Approaches to gesture.** [s.l.]: Mouton De Gruyter, 1986.

LOWE, David G. Distinctive Image Features from Scale-Invariant Keypoints. **International Journal of Computer Vision**, vol. 60, no. 2, p. 91–110, 2004.

Língua Brasileira de Sinais (LIBRAS). InfoEscola. Disponível em: <https://www.infoescola.com/portugues/lingua-brasileira-de-sinais-libras>.

Mohandes, Mohamed A. Recognition of Two-Handed Arabic Signs Using the CyberGlove. **Arabian Journal for Science and Engineering**, vol. 38, no. 3, 2012, pp. 669–677., doi:10.1007/s13369-012-0378-z.

PANWAR, Meenakshi. Hand gesture recognition based on shape parameters. **2012 International Conference on Computing, Communication and Applications**, 2012.

UZAN, A. J. S.; OLIVEIRA, M. R. T.; LEON, I. O. R.. A importância da língua brasileira de sinais – (libras) como língua materna no contexto da escola do ensino fundamental. **XII Encontro Latino Americano de Iniciação Científica e VIII Encontro Latino Americano de Pós-Graduação – Universidade do Vale do Paraíba**, 2008.

WEI, Shih-En; RAMAKRISHNA, Varun; KANADE, Takeo; *et al.* Convolutional Pose Machines. **2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**, 2016.

What is Computer Vision? Hayo. Disponível em: <https://hayo.io/computer-vision>.

LIN, Hsien-i; HSU, Ming-hsiang; CHEN, Wei-kai. Human Hand Gesture Recognition Using a Convolution Neural Network. **IEEE International Conference on Automation Science and Engineering (CASE)**, 2014.

ALANI, Ali; COSMA, Georgina; TAHERKHANI, Aboozar; *et al.*. Hand Gesture Recognition Using an Adapted Convolutional Neural Network with Data Augmentation. **4th IEEE International Conference on Information Management**, 2018.

MISHRA, Manish; SRIVASTAVA, Monika. A View of Artificial Neural Network. **IEEE International Conference on Advances in Engineering & Technology Research (ICAETR)**, 2014.

SCHMIDHUBER, Jürgen. Deep Learning in Neural Networks: An Overview. **Neural Networks**, 61: 85–117, 2015.

LESHNO, Moshe; LIN, Vladimir Ya.; PINKUS, Allan; *et al.* Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. **Neural Networks**, vol. 6, no. 6, p. 861–867, 1993.

H. N. MHASKAR; C. A. MICCHELLI. How to Choose an Activation Function. **Neural Information Processing Systems**, vol. 6, p. 319–326, 1993.

KRIZHEVSKY, Alex; SUTSKEVER, Ilya and HINTON, Geoffrey E. ImageNet classification with deep convolutional neural networks. **Communications of the ACM**, vol. 60, no. 6, p. 84–90, 2017.

GOH, A.t.c. Back-propagation neural networks for modeling complex systems. **Artificial Intelligence in Engineering**, vol. 9, no. 3, p. 143–151, 1995.

RUDER, Sebastian. An overview of gradient descent optimization algorithms. **arXiv preprint arXiv:1600.04747**, 2016.

LIU, Tiany; FANG, Shuangsang; ZHAO, Yuehui; WANG, Peng; ZHANG, Jun. Implementation of Training Convolutional Neural Networks. **arXiv preprint arXiv:1506.01195**, 2016.

JASWAL, Deepika; V., Sowmya; K.P.SOMAN. Image Classification Using Convolutional Neural Networks. **International Journal of Scientific & Engineering Research**, 2014.

RAMAKRISHNA, Varun; MUNOZ, Daniel; HEBERT, Martial; *et al.* Pose Machines: Articulated Pose Estimation via Inference Machines. **Computer Vision – ECCV 2014 Lecture Notes in Computer Science**, p. 33–47, 2014.

WERBOS, Paul John. **The roots of backpropagation: from ordered derivatives to neural networks and political forecasting**, 1994.