UNIVERSIDADE FEDERAL DE VIÇOSA
CAMPUS DE RIO PARANAÍBA
SISTEMAS DE INFORMAÇÃO

LUIGI MILAGRES DE MIRANDA 5181
THIAGO BORGES SILVA 6364

ANÁLISE DE LINGUAGEM DE SINAIS - TRABALHO DE VISÃO COMPUTACIONAL

RIO PARANAÍBA
2019

LUIGI MILAGRES DE MIRANDA 5181
THIAGO BORGES SILVA 6364

ANÁLISE DE LINGUAGEM DE SINAIS - TRABALHO DE VISÃO COMPUTACIONAL

Trabalho desenvolvido na Universidade Federal de Viçosa como parte das exigências para a aprovação na disciplina Visão Computacional

Orientador: João Fernando Mari

RIO PARANAÍBA

2019

Sumário

1 Introdução

A comunicação é parte essencial para a vivência do ser humano em sociedade. Através dela nós transmitimos o conhecimento, expressamos sentimentos e nos socializamos, seja pela fala, pela realização de gestos ou expressões faciais, por palavras escritas em livros, entre inúmeros outros meios, ou seja, o ato de se comunicar é um aspecto extremamente presente em nosso dia-a-dia.

Contudo, para uma pequena parcela da sociedade, a comunicação se torna complicada, na grande maioria das vezes, por ser dificultada por algum obstáculo, deficiência, como a surdez e a mudez. Por isso, múltiplos métodos foram encontrados para diminuir essa lacuna na comunicação para tal parcela, como a linguagem em sinais. Porém, os métodos utilizados, como por exemplo, intérpretes, podem se tornar opções caras para aqueles que não tem condições de adquirir tal solução.

Uma aproximação diferente, envolvendo a visão computacional é o aprendizado de máquina, que vem sendo estudado e adotado para poder solucionar muito problemas diferentes que são enfrentados diariamente, um deles é o problema de integração dos surdos e mudos à sociedade, uma vez que a grande maioria das pessoas não sabem interpretar linguagem de sinais.

Essa linguagem consiste, basicamente, em diferentes posições da mão, gestos corporais e expressões faciais, que podem ou não depender de um contexto para possuir algum significado. Por isso, estão sendo empregados sistemas com algoritmos capazes de receber, analisar, interpretar e classificar a linguagem de sinais, dando como resultado o significado de algum gesto ou expressão, facilitando a mensagem que algum indivíduo queira passar através desses sinais.

1.1 Objetivo Geral

O objetivo desse trabalho é poder criar um modelo de classificação que possa reconhecer, através da classificação de imagens, com a maior precisão possível, o alfabeto básico da linguagem de sinais de A a Z. É necessário, porém, o movimento da mão para as letras J e Z serem representadas e, como o movimento não pode ser reconhecido em apenas uma imagem, nosso *dataset* contém a angulação feita para executar o gestos dessas letras podem haver como classificar e diferenciar das demais.

2 Trabalhos Relacionados

O estudo da classificação de gestos da linguagem de sinais não é algo novo, muito pelo contrário ele esta cada vez mais ganhando seu espaço. Por isso, há uma gama de estudos no qual podemos nos apoiar. (DHIMAN, 2017), trata esse problema de várias maneiras diferentes, usando diversas metodologias. Com base na revisão do estudo dele, a metodologia que tornou capaz atingir a maior precisão foram os seguintes algoritmos de classificação: Máquina de Vetores de Suporte (*SVM*), juntamente com o Histograma de Gradientes Orientados (*HoG*) e as Redes Neurais Convolucionais (*CNN*), atingindo uma precisão máxima de de 81,15% usando *SVM+HoG*.

Na pesquisa em questão, foram utilizados o *dataset* de *ASL (American Sign Language)* e *ISL (Indian Sign Language)*. (DHIMAN, 2017) relata, no seu trabalho, que uma precisão maior poderia ter sido atingida se estivesse usado um *dataset* maior.

A maioria dos estudos relacionados ao reconhecimento e classificação de gestos da linguagem de sinais envolvem o uso de ferramentas bastante avançadas para o tipo de problema que visamos resolver, como sensores de movimento (RIVERA-ACOSTA et al., 2017) (TAO; LEU; YIN, 2018), e a *Data Glove* (OZ; LEU, 2005), que é uma luva que registra os gestos feitos com as mãos, tanto estáticos quanto dinâmicos. Contudo, não temos acessos a essas ferramentas, logo, o treinamento do modelo deve ser realizado da maneira convencional.

A vasta quantidade de pesquisas na área trás inúmeros métodos de aproximação para o problema em questão, soluções podem ser encontradas utilizando *CNNs*, *HMMs (Hidden Markov Models)* (KOLLER et al., 2018), *MN (Moore Neighborhood)*, *k-NN (k-Nearest Neighbors)*, entre outros métodos.

Neste trabalho o problema será abordado usando as *CNNs*. Nos apoiando em trabalhos que apresentam uma aproximação similar (MAKAROV et al., 2019), (GARCIA, 2016) e (SAHA, 2018), chegamos à conclusão de que fazer uso das *CNNs* é um método relativamente simples, convencional, eficiente e confiável para a realização desse tipo de tarefa.

3 Metodologia

3.1 Apresentação do *Dataset*

Em nosso trabalho, será feito o uso do *dataset "ASL Alphabet"*, comitado por Akash na plataforma *Kaggle*. O conjunto de imagens se subdivide em 29 classes rotuladas, onde três delas significam "Enter", "Delete"e "Nada". Desconsiderando essas três classes que não fazem parte do alfabeto de linguagem de sinais, ficamos com 78000 imagens no total, 3000 por classe.

3.2 Materiais e Métodos

Como dito anteriormente, será feito o uso das *CNNs* para a construção do modelo de classificação. Para isso será necessário a realização das seguintes tarefas:

- Preparar o conjunto de dados;
- Montar o modelo de rede neural;
- Treinar o modelo;
- Avaliar o modelo;
- Verificar os resultados e medir a acurácia;

O código havia sido inicialmente feito no *Jupyter Lab*, mas por motivos de armazenamento tivemos que mudar o local de processamento. Para montamos o arquivo de treino foram utilizadas as bibliotecas ***os***, ***OpenCV*** e ***tdqm***. Fez-se o usos de caminhos de diretórios para acessar as imagens e logo foram definidos os rótulos das classes e suas chaves.

Todas as imagens que eram encontradas nas pastas passavam por um processo de conversão para escalas de cinza e posteriormente por uma etapa de redução para o tamanho 50x50 com o uso do ***OpenCV***, já que as imagens originais eram grandes e pesaria ainda mais o processamento desses dados e o treino do modelo. À medida que eram processadas, as imagens foram colocadas no arquivo de treino todas embaralhadas, para evitar o enviesamento do modelo. Para acompanhar o procedimento da montagem do arquivo, o ***tdqm*** proporcionou uma barra de progresso.

Para rodar o modelo, foi utilizado a plataforma *Google Colab*, uma vez que máquinas com alto poder de processamento não estavam ao alcance para a realização do trabalho. Assim, fez-se o *upload* do arquivo de treino para o *Drive*, e foi usado o módulo ***drive*** da biblioteca do *Google Colab* para se ter o acesso aos dados na plataforma.

Para a montagem do modelo de *CNN*, foi utilizada a biblioteca ***Pytorch***, já que ela é considerada por muitos mais fácil de usar do que os demais métodos populares, como o *TensorFlow*, mas não deixa de ser tão poderosa quanto. A Figura 1 ilustra de forma básica o funcionamento do modelo desenvolvido.

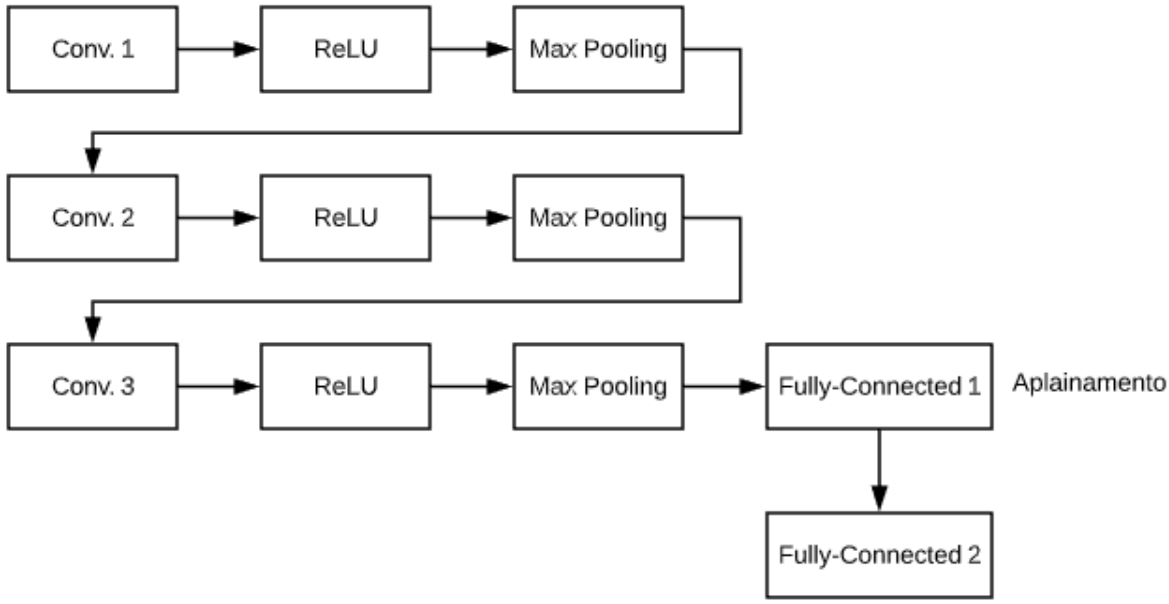

Figura 1 – Exemplo do funcionamento do modelo da *CNN*

Fonte: Autoria Própria

Para treinar o modelo, os dados de treino foram divididos em *features* e *labels*, e depois 10% desses dados foram reservados para validação. Em seguida, definiu-se os *features* e *labels* de treino e validação. Depois, foi decidido utilizar o otimizador de perdas Adam e foi definido o passo de aprendizagem do modelo de 0.001.

Os dados de treino foram agrupados em pacotes (*batches*) de 1000 imagens e o treinamento ocorreu, inicialmente, com 15 épocas (*epochs*). Ao fim da cada pacote os gradientes da função de classificação eram zerados, para que no próximo pacote eles pudessem ser definidos de novo, sem qualquer influência dos demais. Em seguida, o otimizador era atualizado para minimizar a perda de dados do modelo. Ao fim de todas as épocas, o treinamento foi finalizado.

A validação simplesmente funciona em contar quantos acertos o algoritmo teve

usando os 10% de dados que foram separados para validação, fazendo a comparação se a classe da imagem correspondente era a mesma apontada pelo algoritmo.

Enquanto o modelo é treinado, é realizado o teste das acurácias *in-sample* (dentro do conjunto de dados de treinamento) e *out-of-sample* (fora do conjunto de dados de treinamento). Com esses testes, é possível verificar se os parâmetros como o número de épocas estão gerando o *overfitting*. Esse problema se resume em treinar demais o modelo, ao ponto que ele não aprende realmente, mas sim decora o padrão das imagens de treino e, ao ser apresentado uma nova imagem, ele não sabe distingui-la.

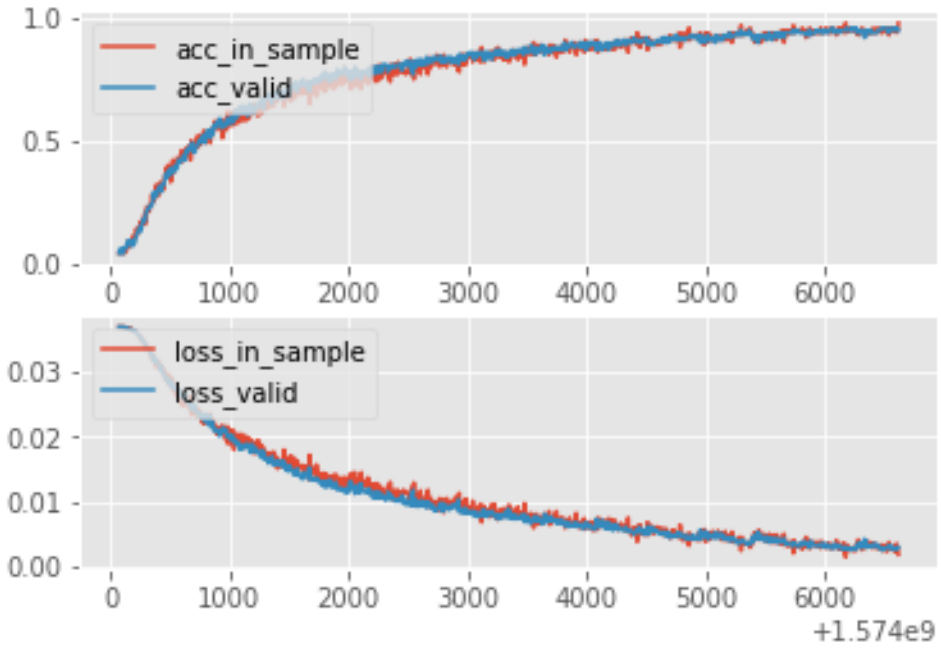

Figura 2 – Acurácia e Perda *in-sample* e *out-of-sample* X tempo de execução

Fonte: Autoria Própria

Um sinal de *overfitting* seria uma diferença considerável entre a acurácia *in-sample* e a acurácia de validação, fora do intervalo entre 5% e 10%. Como visto na Figura 2 isso não ocorre em momento algum durante o treino do modelo.

4 Resultados

Após treinar a rede neural, fez-se o teste de validação e foi encontrado uma acurácia de 94.9%. Os erros e acertos foram plotados em uma matriz de confusão feita usando a biblioteca **Scikit-plot**. Os resultados podem ser visualizados na Figura 3.

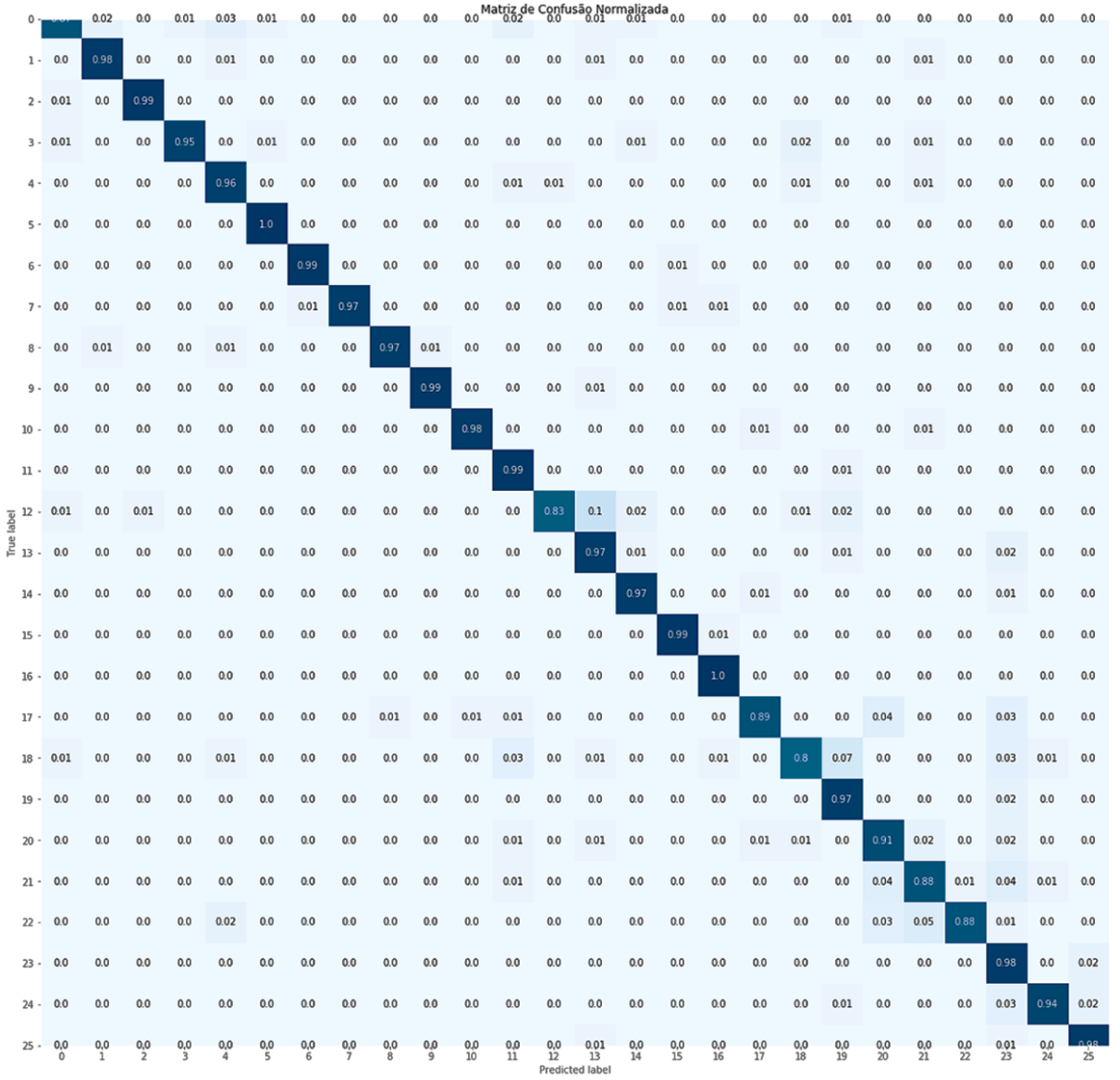

Figura 3 – Matriz de Confusão

Fonte: Autoria Própria

Os índices da matriz representam cada letra do alfabeto no conjunto de dados. Sendo as chaves e seus valores:

A: 0, B: 1, C: 2, D: 3, E: 4, F: 5, G: 6, H: 7, I: 8, J: 9, K: 10, L: 11, M: 12, N: 13, O: 14, P: 15, Q: 16, R: 17, S: 18, T:19, U:20, V: 21, W: 22, X: 23, Y: 24, Z: 25.

Com esses dados, é possível analisar que a classe 18, que seria a letra "S", foi a que teve mais erros por parte do algoritmo, onde 80% foram acertos. E a classe 16, letra "Q", teve uma classificação perfeita com 100% de acertos. Uma taxa de acertos relativamente boa se comparado a outros métodos utilizados nos estudos referenciados.

5 Conclusão

O problema de interpretação de linguagem de sinais está cada vez mais perto de ser resolvido e a barreira social enfrentada pelos deficientes está sendo destruída aos poucos. As *CNNs* têm bastante impacto neste aspecto, em vista que, no futuro, provavelmente todos terão a capacidade de interpretar a linguagem de sinais suplementado pelo auxílio de algoritmos de classificação.

Ao realizar este trabalho e analisar os resultados, pode-se podemos observar que as *CNNs* realmente são ferramentas poderosas para a tarefa de classificação de imagens e objetos, visto que através a acurácia obtida satisfez todas nossas especulativas para com as redes.

Referências

DHIMAN, M. **SIGN LANGUAGE RECOGNITION**. 2017. Disponível em: <https://edu.authorcafe.com/academies/6813/sign-language-recognition>.

GARCIA, B. Real-time American Sign Language Recognition with Convolutional Neural Networks Sigberto Alarcon Viesca Stanford University. 2016. ISSN 23254262.

KOLLER, O. et al. Deep Sign: Enabling Robust Statistical Continuous Sign Language Recognition via Hybrid CNN-HMMs. **International Journal of Computer Vision**, Springer US, v. 126, n. 12, p. 1311–1325, 2018. ISSN 15731405. Disponível em: <https://doi.org/10.1007/s11263-018-1121-3>.

MAKAROV, I. et al. American and russian sign language dactyl recognition. p. 204–210, 2019.

OZ, C.; LEU, M. C. Recognition of finger spelling of american sign language with artificial neural network using position/orientation sensors and data glove. **Lecture Notes in Computer Science**, v. 3497, n. II, p. 157–164, 2005. ISSN 03029743.

RIVERA-ACOSTA, M. et al. American sign language alphabet recognition using a neuromorphic sensor and an artificial neural network. **Sensors (Switzerland)**, v. 17, n. 10, 2017. ISSN 14248220.

SAHA, D. **A very simple CNN project**. 2018. Disponível em: <https://github.com/EvilPort2/Sign-Language>.

TAO, W.; LEU, M. C.; YIN, Z. American Sign Language alphabet recognition using Convolutional Neural Networks with multiview augmentation and inference fusion. **Engineering Applications of Artificial Intelligence**, Elsevier Ltd, v. 76, n. February, p. 202–213, 2018. ISSN 09521976. Disponível em: <https://doi.org/10.1016/j.engappai.2018.09.006>.