
UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Davi Hirafuji Neiva

# **Dynamic Translation between Sign Languages:** A Deep Learning Approach

Recife

2022

Davi Hirafuji Neiva

# **Dynamic Translation between Sign Languages:** A Deep Learning Approach

Tese apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco como requisito parcial à obtenção do título de Doutor em Ciência da Computação.

**Área de Concentração**: Inteligência computacional

**Orientador (a)**: Cleber Zanchettin

Recife

2022

Catalogação na fonte
Bibliotecária Monick Raquel Silvestre da S. Portes, CRB4-1217

N417d Neiva, Davi Hirafuji
Dynamic translation between sign languages: a deep learning approach / Davi Hirafuji Neiva. – 2022.
145 f.: fig., tab.

Orientador: Cleber Zanchettin.
Tese (Doutorado) – Universidade Federal de Pernambuco. CIn, Ciência da Computação, Recife, 2022.
Inclui referências e apêndice.

1. Inteligência computacional. 2. Aprendizagem profunda. 3. Redes neurais. I. Zanchettin, Cleber (orientador). II. Título.

006.31 CDD (23. ed.) UFPE - CCEN 2023-34

**Davi Hirafuji Neiva**

**“DYNAMIC TRANSLATION BETWEEN SIGN LANGUAGES: A DEEP LEARNING APPROACH”**

Tese de Doutorado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Doutor em Ciência da Computação. Área de Concentração: Inteligência Computacional

Aprovado em: 11/03/2022.

_____

**Orientador: Prof. Dr. Cleber Zanchettin**

**BANCA EXAMINADORA**

_____

Prof. Dr. Adriano Lorena Inacio de Oliveira
Centro de Informática / UFPE

_____

Prof. Dr. Leandro Maciel Almeida
Centro de Informática / UFPE

_____

Prof. Dr. João Fausto Lorenzato de Oliveira
Escola Politécnica de Pernambuco/UPE

_____

Prof. Dr. André Câmara Alves do Nascimento
Departamento de Computação/UFRPE

_____

Prof. Dr. Francisco Carlos Monteiro Souza
Coordenação de Engenharia de Software/ UTFPR

## ACKNOWLEDGEMENTS

One of my greatest rides in amusement parks is the roller-coaster. At first you rise up in the sky, excited to see the world that surrounds you where you gaze upon endless possibilities ... then you fall, fall hard. Between twists, turns and loops, you will rise a couple of times just enough to catch a breath to fall again. There will be people shouting, asking to leave, crying, laughing and a few indifferent. There will be a point where you will get tired of whatever emotions you had and think: "there is more?"or "when will this end?"but you can not get out and will have to endure it until you reach your starting point where you are free, a little shooked but free.

Looking back at this research, this analogy resembles the course of this Ph.d with not only me but with everyone next to me riding along. And to that I am most grateful. I thank my mother, who felt the hardest turn and fall, endured flawlessly and to my father, who sat next to her. To my brothers accompanying us all I thank you. I also thank my supervisor observing the ride and calmly waiting for it to end. And last but not least, saving the best for last. The most important person to thank is my wife, who encouraged, helped and sat next to me the whole time, although she thought that by just encouraging she would not have to ride along, how silly of her hehe. Close by or miles afar, she was right next to me the entire time and I thank her immensely.

At last we have arrived at the starting point, shooked and shocked but all alive as it was a hell of a ride. We all shouted, asked to leave, cried, laughed and acted indifferent. These emotions we now leave aside to our next ride.

As I walk away and look back, I ask myself was it worth it? The child in me says: "awesome! look there's another one!"but my body says "easy there boy, you are not as young as you think you are anymore". I can't deny that the knowledge and experience acquired were worth it and for that I am sticking with the child within.

I thank you all!

## ABSTRACT

Sign languages are used by deaf people worldwide to communicate with others. By using body movements, especially the hands, a deaf person can express him/herself. However, Sign language is not universal, which means that an American deaf person may not be able to communicate with a Brazilian deaf person properly. Software-based solutions use machine learning algorithms to recognize Sign language gestures and translate the oral and written language to Sign language, but to our knowledge, no works tackle translation between Sign languages. We propose a mobile application integrated with a server to translate Sign languages that use a smartphone's camera to capture Sign language gestures through video, send to the server, and display the translation back to the user. We use a combination of state-of-the-art Deep learning algorithms such as Mask-RCNN, CNN, and Transformers to perform background removal, feature extraction, Sign language gesture recognition, and translation. We also propose a word-based sentence dataset to recognize continuous Sign language videos. Moreover, we present a webpage to host different sign languages, view gestures individually or in sentences, and create customized hand masks. Using two publicly available Sign language datasets (PHOENIX14-T German Sign language and V-Librasil Brazilian Sign language), our approach improved Word Error Rate (WER) accuracy result by 4% on the PHOENIX14-T while on the V-Librasil dataset, and we achieved a 21.7% WER and a 5% WER using our word-based sentence dataset.



**Keywords**: gesture recognition; deep learning; translation between gestures; computer vision; artificial neural network.

## RESUMO

Língua de sinais são usadas por pessoas surdas ao redor do mundo para se comunicar com outras pessoas surdas. Ao usar movimentos do corpo, especificamente as mãos, uma pessoa surda consegue se expressar. Contudo, língua de sinais não são universais o que significa que uma pessoa surda americana pode não conseguir se comunicar apropriadamente com uma pessoa surda brasileira. Soluções baseadas em software usam algorítmos de aprendizagem de máquina para reconhecer os sinais em uma língua de sinais e tranduzir uma língua falada ou escrita para uma língua de sinais, mas até a escrita desta tese, não identificamos nenhum trabalho que tenha proposto uma tradução entre língua de sinais. Para isto, nós propomos uma aplicação móvel integrada a um servidor que utiliza a sua câmera para capturar vídeos de uma língua de sinais, enviar este vídeo para um servidor e mostrar sua tradução em outra língua de sinais. Nós usamos uma combinação do estado-da-arte em aprendizado profundo como Mask-RCNN, CNN e Transformers para realizar as tarefas de remoção de plano de fundo, extração de características, reconhecimento e tradução de sinais. Também propomos uma base de dados composta somente de palavras para reconhecer sentenças em língua de sinais. Além disso, nós propomos uma página web para hospedar diferentes língua de sinais para visualizar sinais individualmente ou em sentenças e criar máscaras customizadas das mãos. Utilizando dois datasets públicos (PHOENIX14-T, uma base alemã de língua de sinais e V-Librasil, uma base brasileira), nossa proposta melhorou a taxa de erro de palavra (WER) em 4% na base alemã enquanto que na V-Librasil nós atingimos um WER de 21.7% e 5% para palavras e sentenças utilizando nossa base de palavras.



**Palavras-chaves**: reconhecimento de gestos; apredizagem profundo; tradução entre língua de sinais; visão computacional; redes neurais artificiais.

## LIST OF FIGURES

## LIST OF TABLES

## LIST OF ABBREVIATIONS AND ACRONYMS

| | |
|---|---|
| **AI** | Artificial Intelligence |
| **AR** | Augmented Reality |
| **ArSL** | Arabic Sign Language |
| **ASL** | American Sign Language |
| **BMU** | Best Matching Unit |
| **BoF** | Bag of Feature |
| **CCD** | Charge-CoupledDevice |
| **CNN** | Convolutional Neural Network |
| **COHST** | Combined Orientation Histogram and Statistical Measures |
| **CPU** | Central Processing Unit |
| **CSL** | Chinese Sign Language |
| **CTC** | ConnectionistTemporal Classification |
| **DCT** | Discrete Cosine Transform |
| **DGS** | Deutsche Gebärdensprache |
| **DL** | Deep Learning |
| **DoG** | Difference of Gaussian |
| **DTW** | Dynamic Time Warping |
| **DWT** | Discrete Wavelet Transform |
| **ELM** | Extreme Learning Machines |
| **EOH** | Edge orientation histogram |
| **FC** | Fully Connected |
| **FPGA** | Field Programmable Gate Array |
| **GAN** | Generative Adversarial Network |
| **GAP** | Global Average Pooling |
| **GMM** | Gaussian Mixture Model |

| | |
|---|---|
| **GPS** | Global Positioning System |
| **GRU** | Gated Recurrent Unit |
| **HMM** | Hidden Markov Model |
| **HOG** | Histogram of Oriented Gradients |
| **HSR** | Hand Skeleton Recognition |
| **HSV** | Hue Saturation Value |
| **HTM** | Hierarchical Temporal Memory |
| **ILSVRC** | ImageNet Large-Scale Visual Recognition Challenge |
| **IMEn** | Intrinsic Mode Entropy |
| **KNN** | K-Nearest Neighbors |
| **LBP** | Local Binary Patterns |
| **LCS** | Local Contour Sequence |
| **LDA** | Linear Discriminant Analysis |
| **Libras** | Brazilian Sign language |
| **LSTM** | Long Short Term Memory |
| **Mask R-CNN** | Mask Region-Based Convolutional Neural Networks |
| **MD** | Minimum Distance |
| **MLP** | Multi-layer Perceptron |
| **NLP** | Natural Language Processing |
| **OHTM** | Optimized Hierarchical Temporal Memory |
| **OS** | Operational System |
| **PaSL** | Pakistan Sign Language |
| **PCA** | Principal Component Analysis |
| **PDF** | Probability Density Function |
| **PE** | Positional Encoding |
| **PoSL** | Polish Sign language |
| **RBF** | Radial Basis Function |

| | |
|---|---|
| **ReLU** | Rectified Linear Unit |
| **RGB** | Red Green Blue |
| **RNN** | Recurrent Neural Network |
| **ROI** | Region of Interest |
| **RPCA** | Recursive Principal Component Analysis |
| **SAE** | Sparse-Autoencoder |
| **SASL** | South African Sign Language |
| **SED** | Squared Euclidian Distance |
| **SF** | Shape Factor |
| **SGM** | Single Gaussian Model |
| **SIFT** | Scale-Invariant Feature Transform |
| **SMC** | Spatial Multi-cue |
| **SMS** | Short Messaging Service |
| **SOM** | Self-Organizing-Maps |
| **STMC** | Spatial-Temporal Multi-Cue |
| **SURF** | Speeded Up Robust Features |
| **SVM** | Support Vector Machine |
| **SWAB** | Sliding-window and Bottom-up Algorithm |
| **TDD** | Telecommunication Device for the Deaf |
| **TMC** | Temporal Multicue |
| **UML** | Unified Modeling Language |
| **VOIP** | Voice Over IP |
| **VR** | Virtual Reality |
| **WER** | Word Error Rate |
| **WHO** | World Health Organization |

## CONTENT

# 1 INTRODUCTION

Communication is an essential part of our daily lives. We use it in various aspects of our routine without noticing its importance. It has a social and emotional impact, and its absence may cause loneliness, isolation, and frustration (CIORBA et al., 2012). People with hearing loss have difficulties communicating verbally because they cannot hear all the sounds or even their voice. According to World Health Organization (WHO)[1], more than 5% of the world's population have hearing loss that can be mild, moderate, severe, or profound. This statistic means that approximately 430 million people struggle to communicate daily due to hearing problems.

Most people with hearing disabilities, especially those with moderate to profound disabilities, communicate using sign languages to convey meaning instead of acoustically conveyed sound patterns. By combining hands movement and facial expressions, an impaired person can communicate with others who know sign language. However, every country has its sign language and, for the communication to be established, both ends must know the same sign language. Technology plays an essential role in facilitating the communication of deaf persons, mainly through mobile applications such as video chatting. Mobile technology became people's personal carry-on computer augmenting the possibility of communications.

Cellphones are now "smart" phones, which can be seen as pocket-size computers nowadays. Cellphones had limited resources and provided limited interactions with the user, all done through a physical keyboard. Smartphones are enriched with cameras, internet connection, sensors, fast Central Processing Unit (CPU), memory, and a significant number of applications, thus the term "smart". Through the smartphones' hardware such as sensors, smartphones can perform biometric recognition such as facial, iris, specific hand movement in the air or on the screen, and fingerprint allowing interesting end-user applications. They provide engaging forms of interaction such as: pause/stop a video when detecting if the user is no longer looking at the screen and resume when looking again (iris detection); if a user is reading a book, the smartphone can flip the page when a specific hand gesture is performed (hand movement); unlock screen and make online purchases based on the face or fingerprint detection (facial or fingerprint recognition). Various other types of recognition are also performed, such as touch intensity, gesture drawing on the screen, facial expressions). Furthermore, through software,

[1] http://www.who.int/en/news-room/fact-sheets/detail/deafness-and-hearing-loss

mainly using Artificial Intelligence (AI) (WINSTON, 1992), smartphones can also perform language translation through voice recordings or written text which has had a significant impact on tourism since it enabled users to go to places where the language was an impediment (GIDUMAL, 2020). However, none are Sign language-based. The convenience of mobility and the available technological resources motivated this research on gesture recognition using these devices. Although gesture recognition technology is already available in high-end smartphones, our work focuses on Sign Language recognition and translation using any smartphone.

Nevertheless, gesture recognition is not an easy task especially involving sign language gestures, which can be static or dynamic, and involve complex (with heterogeneous elements) and dynamic (with moving elements) background in a real-world scenario. Moreover, in order for a computer to recognize gestures, it requires a lot of computing power, memory, and storage capacity features that a smartphone does not have as abundant as a desktop computer (i.e., Android devices limit memory usage to a percentage based on the available memory and Operational System (OS) being used). For example, neural networks are algorithms that can recognize gestures but are computationally demanding and, in a smartphone, would consume most of its resources (if not all), thus affecting its performance. With this difficulty in mind, we combined the computing power of a desktop/server computer with a smartphone application to allow deaf people from different countries to communicate.

This work focuses on sign language recognition and translation using state-of-the-art AI algorithms in Deep Learning (DL) (GOODFELLOW; BENGIO; COURVILLE, 2016) with the primary objective of facilitating communication between deaf people from different countries by proposing a translator between sign languages. Furthermore, our system will be able to display translated gestures both ways.

In previous works on this area, we achieved an overall classification rate of 61% accuracy in recognizing hand images from static gestures from Brazilian Sign Language - Libras using Extreme Learning Machines (ELM) (NEIVA; ZANCHETTIN, 2016). The current work aims to evolve this approach and tackle problems such as efficient static complex background removal, dynamic gestures, and gesture translation. To accomplish this, we used new approaches of Deep Learning algorithms such as CNN (ALBAWI; MOHAMMED; AL-ZAWI, 2017), Mask Region-Based Convolutional Neural Networks (Mask R-CNN) (HE et al., 2017b) and Transformers (VASWANI et al., 2017a).

## 1.1 SIGN LANGUAGE

Sign language can be understood as a means of communication through bodily movements, especially of the hands and arms that is used when spoken communication is impossible or not desirable. The practice is probably older than speech. Early body language consisted of as grimaces, shrugs, or pointings that evolved to complex fingers configurations. The first educator of Sign language, Charles-Michel, abbé de l'Epée, developed a system for spelling out French words with a manual alphabet and expressing whole concepts with simple signs. From l'Epée's system the French Sign Language (FSL)[2] was developed.

FSL was brought to the United States in 1816 by Thomas Gallaudet, founder of the American School for the Deaf in Hartford, Connecticut. The new sign language was combined with the various systems already in use in the United States to form ASL. In the same manner, FSL was brought to Brazil by Ernest Huet and combined with different communication system already in use[3].

Sign languages use a complex combination of movements and go beyond the mimic. There are two basic forms of representation in Sign languages: based on letters of the alphabet as in Figure 2 and based on a sequence of movements to represent words as in Figure 1.

Figure 1 – Sequence of hand movements to represent the word house in ASL.

Source: https://www.signingsavvy.com/search/house (2022)

Sign languages have their own grammar, with verbs, pronouns, adverbs, etc., but they do not follow the same grammar as spoken languages. For instance, verbs are not conjugated and there are no articles or prepositions (LIDDELL et al., 2003). Taking as an example "The boy threw the ball."in English. This sentence in sign language would be translated to "BOY THROW BALL". Another similar example: "The ball was thrown by the boy.". In sign language grammar this sentence would be translated to "BALL, BOY THROW".

Moreover, in sign language, facial expressions are used to express both linguistic information and emotions. For example: eyebrow raise is necessary to mark general questions in most sign languages (NGUYEN; RANGANATH, 2012).

[2] https://www.britannica.com/science/deafness
[3] https://brasilescola.uol.com.br/educacao/lingua-brasileira-sinais-libras.htm

Figure 2 – Hand configurations for the Libras alphabet.


Source: https://www.ultratec.com/products/text-telephones/superprint-4425/ (2022)

## 1.2 MOTIVATION

There is a solid motivation to include people with disabilities (mental or physical) to benefit from the technological advancements in modern society. This is known as digital inclusion, the ability of individuals and groups to access and use information and communication technologies (BECKER et al., 2012). New computational resources expanded the possibility of how people with disabilities use technology, increasing their autonomy to communicate even with the absence of human interpreters. One of the first electronic devices to be adapted for people with hearing disabilities was the telephone which was called: Telecommunication Device for the Deaf (TDD)[4] (Figure 3). This device can work with a relay service where someone intermediates a call between a deaf and a hearing person. The function of the relay service is to say what the deaf person wrote in the TDD to a hearing person or to type what was said to the deaf person. Two deaf persons can call one another directly without using a relay service. In this case, texts are sent between them.

There are Sign language interpreters that help translate the oral language to sign language and vice-versa, but a deaf person will not be near an interpreter nor a TDD all the time, and that is when things become challenging for a deaf person to communicate because not everyone knows sign language.

With the uprising of smartphones, placing a call became a mere functionality amidst many others. Smartphones became, for the majority of people, an indispensable item. It brought new forms of communication such as voice messages, video and call conferences, Voice Over

[4] https://www.dialogic.com/glossary/telecommunication-device-for-the-deaf-tdd

Figure 3 – Telephone used by deaf people to communicate with others from Ultratec.


Source: https://www.ultratec.com/products/text-telephones/superprint-4425/ (2022)

IP (VOIP), real-time chats, social media, etc. It brought new forms of interaction with touch screens, cameras (AR and Virtual Reality (VR)), sensors (accelerometer and gyroscope), and Global Positioning System (GPS). Other characteristics that made smartphones so popular are their portability, easy interaction, and the most important, a vast amount of applications (apps) for many needs that a person may have.

However, there are a few communication apps that people with hearing disabilities can use. For example, a deaf person can use a smartphone to communicate via text messages using Short Messaging Service (SMS) (LÓPEZ-LUDEÑA et al., 2013) or any other messaging app (e.g., Messenger[5], Whatsapp[6], Line[7]) but they would have to use a written language or place video calls (but in this case the other person has to know the same sign language). To make things more complicated, some deaf people only know Sign language, they do not know how to write (SCHELP, 2009).

Most mobile applications developed for communication are not adequate for hearing impaired people. Even when new tools are developed, allowing the inclusion of deaf users in the communication process, they are still centered on the hearing person.

There are a couple of sign language-driven apps, but they focus on teaching basic sign language. For Android devices, there are: ASL American Sign Language[8], Sign Language for Beginners[9], Sign ASL[10] and JW Library Sign Language[11]. For Apple devices: ASL App[12], ASL

[5] https://play.google.com/store/apps/details?id=com.facebook.orca
[6] https://play.google.com/store/apps/details?id=com.whatsapp
[7] https://play.google.com/store/apps/details?id=jp.naver.line.android
[8] https://play.google.com/store/apps/details?id=tenmb.asl.americansignlanguagepro
[9] https://play.google.com/store/apps/details?id=com.sign.language.learn1234
[10] https://play.google.com/store/apps/details?id=com.signasl.signasl
[11] https://play.google.com/store/apps/details?id=org.jw.jwlibrary.signlanguage
[12] https://apps.apple.com/us/app/the-asl-app/id921030207

Kids - Sign Language[13], Marlee Signs[14] and PCS Sign Language Flash Cards[15]. Other apps translate spoken and/or written language to sign languages such as VLibras[16] and HandTalk Translator[17].

All of the applications above have a non-deaf perspective, meaning that the applications help a hearing person communicate with a deaf person by teaching sign language or translating text or spoken words into a sign language. By inverting this perspective to focus on the deaf person's needs, i.e., communication through gestures, things become more challenging. We need to make a machine "see" and recognize what is being seen just as well as humans do, performing gestures and facial recognition. In sign languages, this involves some aspects that need to be considered. First of all, gestures can be static or dynamic. For instance, numbers and some letters of the alphabet are static, meaning that there is no motion involved and can be understood using one image, but some words like "Hello", "Good morning" and "House" in American Sign Language (ASL), are dynamic gestures for which a person needs to move the hands in order to convey its meaning. Dynamic gestures must be represented by several images. Differing static and dynamic gestures can be a challenging task.

Another challenge involved is background removal (THEPADE et al., 2013). The background has much information for a computer that can detriment gesture recognition. Removing unwanted information increases the chances of having an accurate recognition. Nevertheless, this is a complex problem due to the inconsistency between images. If the background is static, meaning that nothing around the person performing the gesture changes except their hand, it is relatively straightforward to remove it, but background removal becomes complex when it involves dynamic lighting, constant background movement, and color change.

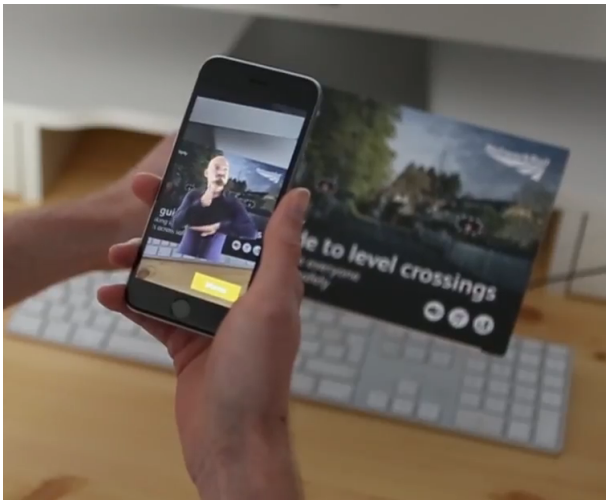
Finally, facial expressions play an important role in sign language because they convey the emotional state, intention, and grammatical formation of a sentence. However, this adds more complexity to Sign language recognition since it involves a combination of face, head, eye, eyebrow, and mouth recognition (KIMMELMAN et al., 2020).

We found two applications with a different approach by considering the deaf perspective. Marlle Keyboard[18] aims to substitute the keyboard comprised with alphabetical letters, with

[13] https://apps.apple.com/us/app/asl-kids-sign-language/id962456700
[14] https://apps.apple.com/br/app/marlee-signs/id566054855
[15] https://apps.apple.com/us/app/sign-language-flash-cards/id601324922
[16] https://apps.apple.com/us/app/vlibras/id1039641615
[17] https://play.google.com/store/apps/details?id=br.com.handtalk
[18] https://apps.apple.com/us/app/marlee-keyboard/id983473162

one containing sign language image gestures. Another app called Signly[19] uses AR, to display a person on a smartphone's screen translating a written text to sign language (Figure 4).

Figure 4 – Written text being displayed in sign language using AR from Signly




Source: https://signly.co/ (2022)

Although these apps are a great initiative with uncountable benefits to the deaf community, they are restricted to one sign language only. Therefore, we intend to consider the deaf perspective and use state-of-the-art AI algorithms to implement solutions that help communication between deaf people who express in different sign languages.

## 1.3 OBJECTIVES

Our main objective is to develop a solution to translate between sign languages, facilitating communication of deaf persons who know different sign languages through an integrated server and mobile technology. For example, a Brazilian and a German, each one using their native sign language, Brazilian Sign language (Libras) and DGS respectively, could communicate using our solution in the following manner: the Brazilian user uses a mobile device's camera to capture the gestures performed by the German. The system performs the background removal, recognition, and translation of DGS gestures and returns to the Brazilian user's mobile application the corresponding gesture in Libras. This way, the Brazilian user can understand DGS gesture and vice-versa.

This work has the following specific objectives:

- Review the literature focusing on sign language in a mobile context;

[19] https://signly.co/

- Propose a novel background removal approach to leave just the hand in an image;
- Develop a computational infrastructure for capturing and translating sign languages;
- Perform continuous sign language recognition from videos;
- Develop user interface for web and mobile technologies hosted in a server.

In this work we use the DGS dataset to validate our recognition approach and use Libras and ASL to validate our translation approach.

## 1.4 RESEARCH QUESTIONS

Based on the literature about Sign languages recognition, we elaborated the following research questions:

- Question 1: Is it possible to perform translation between sign languages using a deep learning NLP architecture ?
- Question 2: Can a word-based sign language dataset be used to recognize sign language sentences?

These research questions are better contextualized in Chapter 3 in the light of our literature review.

## 1.5 PUBLICATIONS

Three papers were derived from this research:

1. Neiva, D. H., & Zanchettin, C. (2022). Translating between Sign Languages using Transformers. Expert Systems with Applications. Under review.
2. Neiva, D. H., & Zanchettin, C. (2018). Gesture recognition: A review focusing on sign language in a mobile context. Expert Systems with Applications, 103, 159-183 (NEIVA; ZANCHETTIN, 2018).

3. Neiva, D. H., & Zanchettin, C. (2016). A Dynamic Gesture Recognition System to Translate between Sign Languages in Complex Backgrounds. In 2016 5th Brazilian Conference on Intelligent Systems (BRACIS) (pp. 421-426). IEEE. (NEIVA; ZANCHETTIN, 2016).

## 1.6 THESIS STRUCTURE

Chapter 2 describes the DL algorithms we used, explaining why we chose them in this research. Next, Chapter 3 discusses related works giving an overview of the techniques used, datasets and results. In this chapter, we present our overall thoughts on the literature presented, and based on them, we discuss possible future research goals to improve this area. In Chapter 4 we describe the datasets we used and the methodology applied to perform training and testing. Chapter 5 we introduce sAIgns, our translation system, as well as its architecture. Then, we describe how every module is organized and used and detail sAIgns' web and mobile interfaces and functionalities. Next, Chapter 6 presents our results using the proposed solution. And lastly, we conclude our research on Chapter 7 by presenting a summary of what was achieved and future research directions.

# 2 BACKGROUND

This chapter discusses the main techniques used that comprises our architecture, experiments and justify our choices. Our architecture uses DL algorithms such as Mask R-CNN, CNN, and Transformers to accomplish our research goal: to translate between sign languages. Deep learning is a subset of machine learning, which is essentially a neural network with three or more layers. These neural networks attempt to simulate the behavior of the human brain allowing it to "learn" from large amounts of data. Deep learning eliminates some of data preprocessing that is typically involved with machine learning. These algorithms can ingest and process unstructured data, like text and images, and automate feature extraction, removing some of the dependency on human experts (GOODFELLOW; BENGIO; COURVILLE, 2016).

## 2.1 CONVOLUTIONAL NEURAL NETWORK - CNN

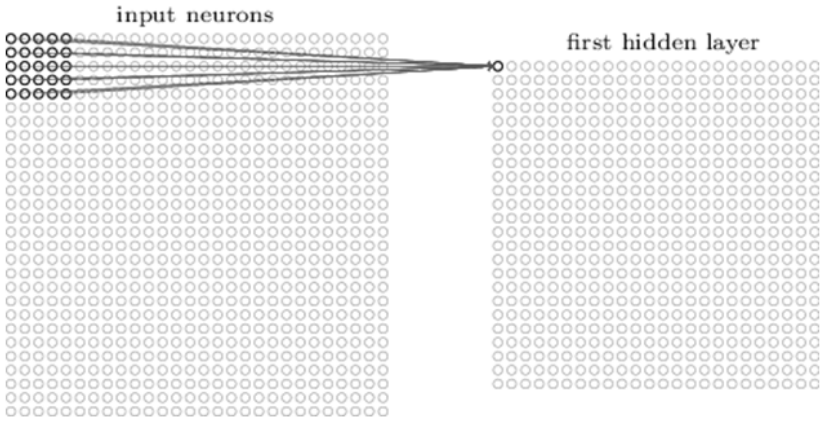
CNN is a Deep Learning network achieving remarkable results in image classification. This is due to its ability to extract features using a series of windows (kernels or neurons) that slide through an image. The core of a CNN is comprised of a combination of layer-based operations such as:

1. Convolution
2. Rectified Linear Unit (ReLU)
3. Pooling
4. Dropout
5. Fully Connected Layer - FC

Convolution is comprised of various kernels (or neurons) of fixed equal size that correspond to one receptive field when convolved (Figure 5). This results in an activation map of dot products (Figure 6b). Figure 6a represents the result of a convolution with 5 kernels of size *5 x 5 x 3* in a *32 x 32 x 3* image. This results in a activation map of size *28 x 28 x 5*.

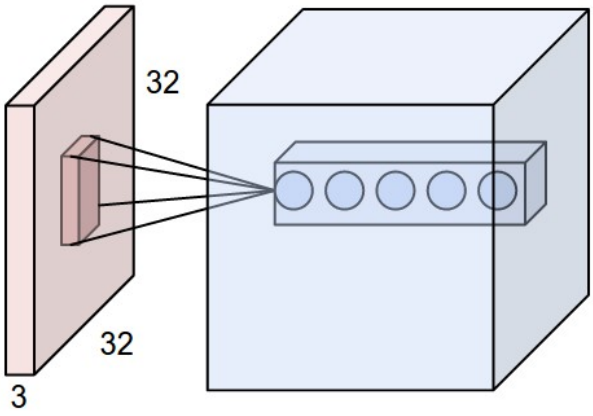
ReLU is the most commonly used activation function in deep learning models (RASAMOELINA; ADJAILIA; SINČÁK, 2020). The function returns 0 if it receives any negative input, but for any positive value $x$ it returns that value back.

Figure 5 – Convolution step representation. It can be understood as a window of fixed size that produces one value. At every step this window is shifted throughout the data until it reaches the end.

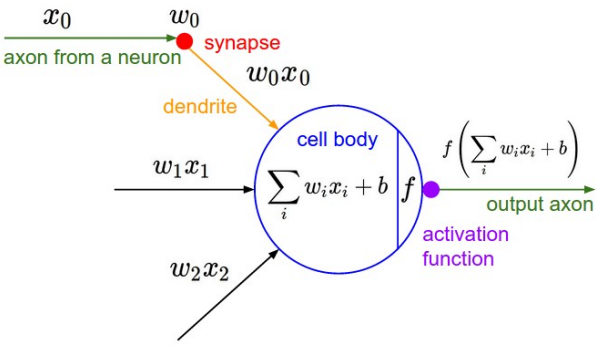



Source: https://adeshpande3.github.io/assets/ActivationMap.png (2022)

Figure 6 – Convolution detailed. a) Once the window has shifted through the entire data, it produces a smaller matrices. The number of filters represents the number of layers. b) The math behind to get all values of a window to generate one value at a given step.




(a) 5 filters with size 5x5 result



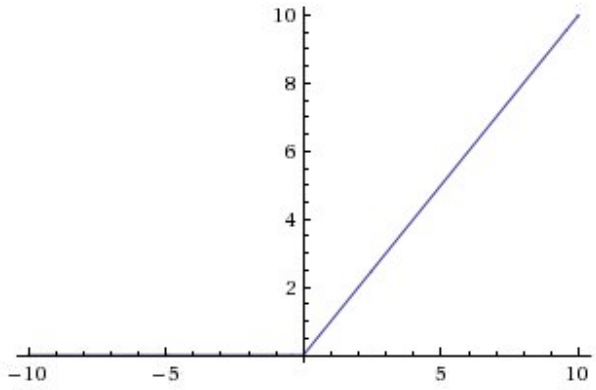

(b) Activation function

Source: http://cs231n.github.io/convolutional-networks/ (2022)

$$f(x) = max(0, x) \tag{2.1}$$

This activation function has become widely used because it is more computationally efficient than Sigmoid-like functions (exponential operations) since it just needs to pick *max(0,x)* and it tends to have a better convergence performance and avoids vanishing gradients. Vanishing gradients commonly occur in Sigmoid and Tanh functions because when the inputs grow extremely small or extremely large, the sigmoid function saturates at 0 and 1, while the tanh function saturates at -1 and 1. When inputs have large negative values, these functions saturates at 0 and at 1 for large positive values and when the gradient is close to 0, weights are barely updated. As opposed to vanishing gradients, there are exploding gradients which

Figure 7 – Graphical representation of ReLU.




Source: https://i.stack.imgur.com/CA4Vd.jpg (2022)

occur when large error gradients accumulate, resulting in extremely large updates to neural network model weights during training. As a result, the model is unstable and incapable of learning from training data. To overcome these problems, low weights initialization such as Xavier initialization can be used.

The idea behind the Pooling layer is to reduce the size of an activation map, thus reducing the number of parameters and controlling overfitting. There are a couple of pooling types (max, average, and normalization), but the most commonly used is the max and average (Figure 8).

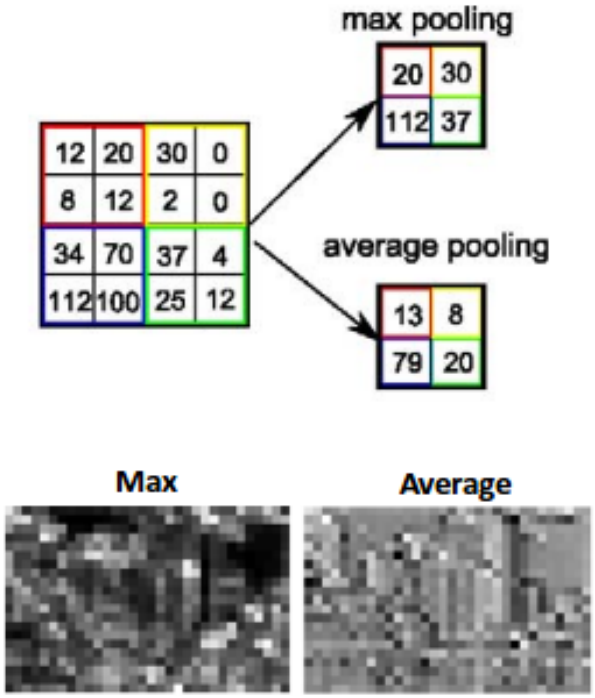
Max pooling extracts the most important features like edges, whereas average pooling extracts features smoothly. Therefore, Max pooling is better for extracting the extreme features, while average pooling sometimes can not extract good features because it takes all values into account, resulting in an average value that may or may not be important for object detection type tasks.

The dropout layer disables randomly selected neurons during the forward pass, and their weights are not updated in the backward pass. By disabling specific neurons, the network can avoid overfitting by having a better generalization from learning from other parts.

The last layer is a fully connected layer that connects all of the neurons of the previous layer to all neurons of the FC layers. Finally, the FC layer is a Multi-layer Perceptron where backpropagation is used to update all of the weights of the kernels in the convolution operation.

An alternative to FC is Global Average Pooling - GAP. It is an approach to decrease the overfitting problem. As we already mentioned, the pooling layer reduces the number of parameters. In this case, the pooling layer will be the size of the activation map, which will be reduced to a single number (Figure 9). Using GAP decreases overfitting and reduces the

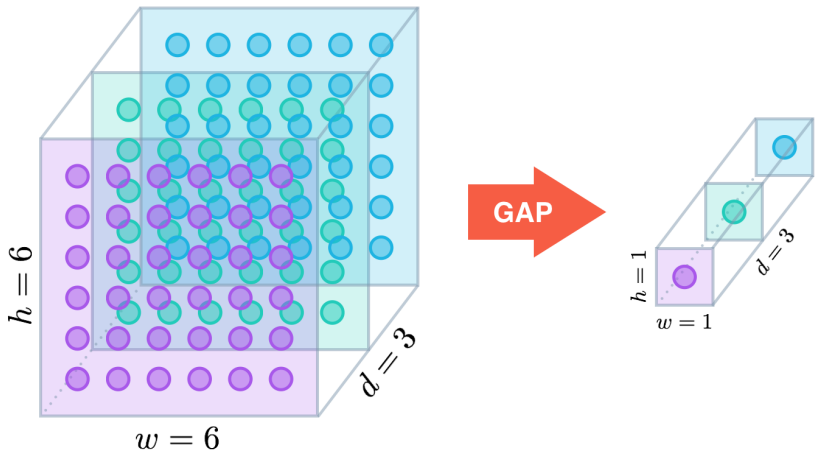
Figure 8 – Max and Average pooling difference.




Source: https://cdn-images-1.medium.com/max/800/1*C0EwU0aknuliOsGktK6U0g.png (2022)

computational cost compared to using an FC layer.

Figure 9 – Global average pooling representation.

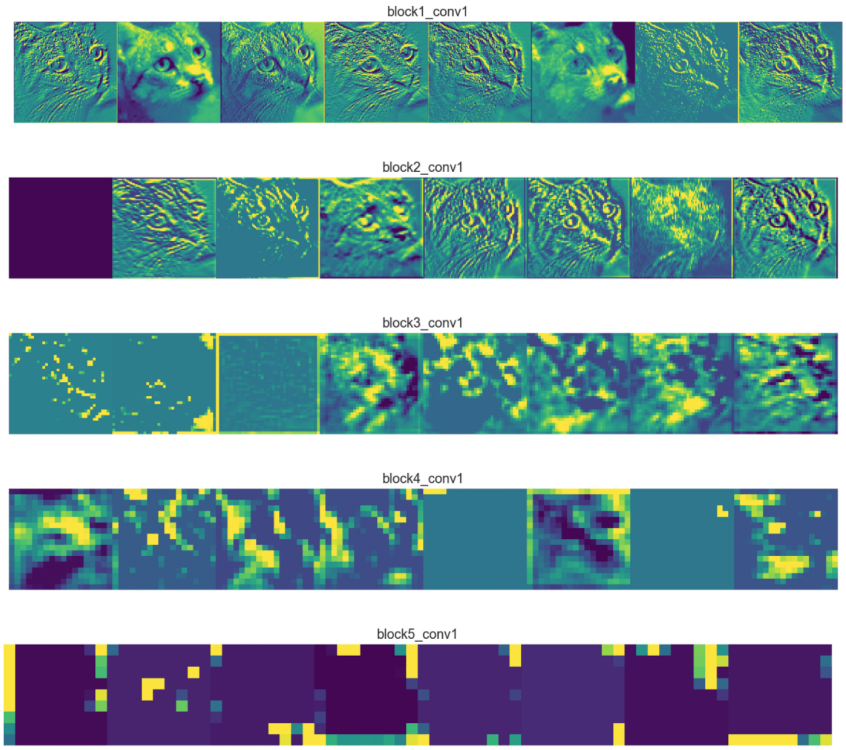



Source: https://alexisbcook.github.io/2017/global-average-pooling-layers-for-object-localization/ (2022)

One important thing to mention is the ability of a CNN to extract features. For example, using hundreds or thousands of learning weights through convolutions, a CNN can represent an image the way a human can by identifying certain patterns.

As explained before, the convolution calculation involves various neurons/kernels/filters of fixed equal size that, when convolving through an image, represent one receptive field (Figure 5). By updating the weights via backpropagation, convolutional weights are updated and can extract different features the deeper they go. By analyzing Figure 10 taken from a VGG CNN convolutional layer, we can see that the filters are capable of extracting particular contours of the image as seen in the convolutional "block1_conv1"and "block2_conv1". As we use more filters or go deeper into the network, the feature maps look less like the original image and more like an abstract representation. For example, as seen in "block3_conv1"the cat is somewhat visible, but after that, it becomes unrecognizable.

Figure 10 – Visualization of feature maps from CNN. The more layers there are, more abstract representation of data is captured.




Source: https://towardsdatascience.com/applied-deep-learning-part-4-convolutional-neural-networks-584bc134c1e2 (2022)

This is important because we want to extract as much relevant information as possible from a gesture. Since we want just the features, networks such as Inceptions and VGGs made their learned convolutional weights available. This can be helpful because these weights were trained extensively throughout thousands of different images, and we can fine-tune our network to our dataset to obtain features faster.

### 2.1.1 Why CNN?

CNN has been used in various domains, from music to Natural Language Processing - NLP. For instance, according to (DONG, 2018), CNN was able to achieve human-level accuracy when classifying musical genres. In image classification, GoogleLenet, a CNN network, achieved 99% accuracy in the ImageNet 2014 competition. In the NLP domain, according to (KIM, 2014), CNN achieved 90% accuracy when classifying sentences. CNN has even been used to learn how to play computer games like Checkers and Go. This shows that CNN has excellent potential in various domains (especially in image classification), which motivated us to use this feature extraction technique. Furthermore, using consolidated architecture such as EfficientNet (TAN; LE, 2019) can help us extract more relevant features. Figure 67 shows our EffientNet model to train and extract features from the PHOENIX-14T and V-Librasil dataset.

This shows that CNN has excellent potential in various domains (especially in image classification). By using convolutions to automatically extract features, we wanted to see if the extracted features would give us interesting results. Using learned weights from already consolidated architecture, instead of random initial weights, can help us extract more relevant features faster.

It is worth mentioning that CNNs or any other Deep Learning Network requires a large and highly representative dataset to generalize appropriately. This was another motivation to see how it can perform in small and low representative Sign language datasets.

## 2.2 MASK REGION-BASED CNN - MASK R-CNN

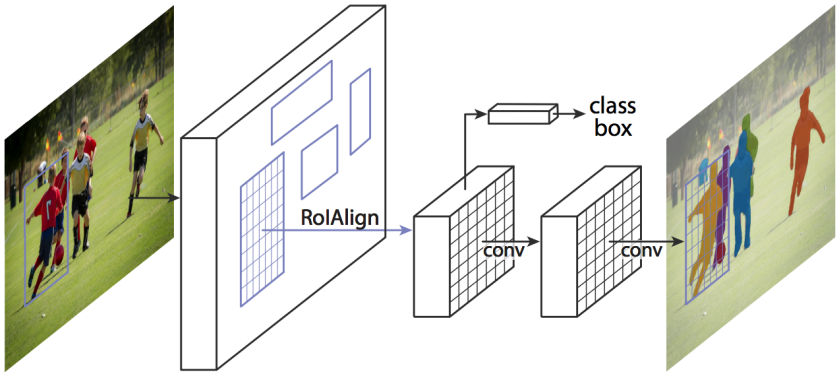
Mask R-CNN (HE et al., 2017b) is a CNN architecture used in image segmentation problems. This type of CNN is used to create masks surrounding multiple objects as well to classify them (Figure 11).

Its architecture is based on the Faster-RCNN (F-RCNN) but with added convolution layers to predict the masks. It is comprised of multiple stages in order to generate these masks. The first stage is gathering features from images using a regular CNN network. Next, region proposal Networks (RPN) are used to obtain regions that might contain an object. Then, RoIPool is used to extract features and classify what object the RoI is from these regions. Finally, in parallel to the last step, Mask-RCNN uses a Fully Convolutional Network (FCN) combined with RoIAlign to predict precise pixel-level masks.

Figure 11 – Objects mask being generated by Mask R-CNN. Every detected object in the image has its mask representation.


Source: https://adeshpande3.github.io/assets/ActivationMap.png (2022)

Figure 12 – Mask-RCNN architecture adapted from (HE et al., 2017a).




Source: HE et al. (2017a)

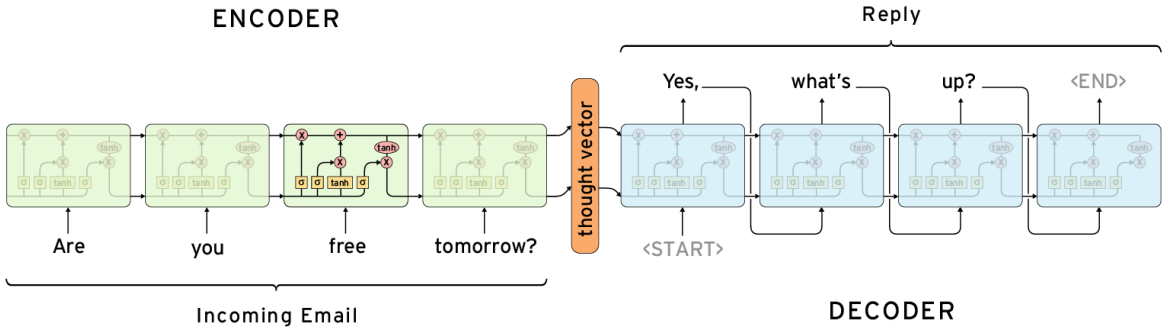
### 2.2.1 Why Mask-FRCNN?

Mask-RCNN is the state-of-the-art in image segmentation (TIAN; SHEN; CHEN, 2020). Our goal is to remove the background to leave just the hand in the image. Using this network, we will be able to obtain a precise mask of the hands using simple image manipulation, and we can easily obtain our image with just the hand by subtracting the hand mask from Mask-RCNN from the original image.

## 2.3 SEQUENCE TO SEQUENCE ARCHITECTURE

Sequence to sequence (seq2seq) architecture is used to solve voice recognition, chatbot applications, and language translation problems. When used with LSTM and GRU, it achieved promising results through an encoder and decoder architecture (ZEYER et al., 2019). This architecture is used when there is a variable-length input with a variable-length output (Figure 13). The encoder maps a variable input sequence to a vector, while the decoder maps back the vector to a variable-length sequence, thus their names (CHO et al., 2014).

Figure 13 – Seq2Seq architecture comprised of stacks of encoders and decoders.




Source: https://towardsdatascience.com/nlp-sequence-to-sequence-networks-part-2-seq2seq-model-encoderdecoder-model-6c22e29fd7e (2022)

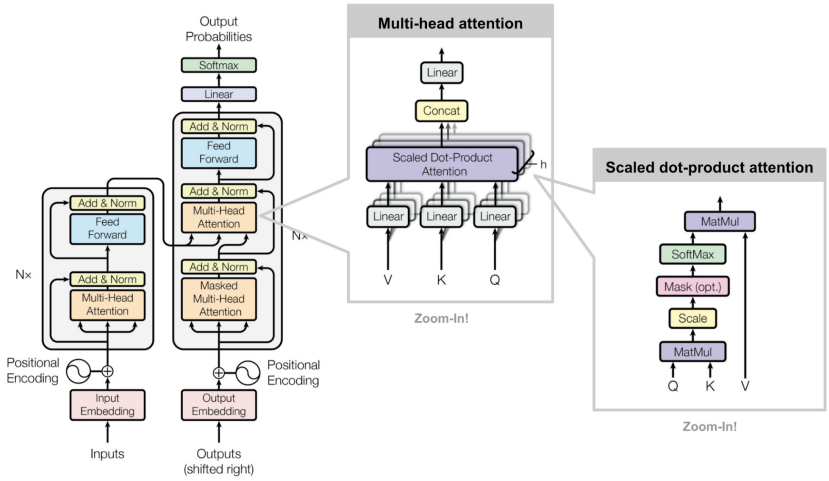
For time-series problems, Long Short-Term Memory (LSTM) (YU et al., 2019) and Gated Recurrent Unit (GRU) (DEY; SALEM, 2017) have long been used in academia and industry for issues such as Natural Language Processing (NLP) (THAKKER et al., 2019; HUANG; FENG, 2019; KAMATH; LIU; WHITAKER, 2019), voice recognition (CHERNYKH; PRIKHODKO, 2017), and data prediction (KIM et al., 2017), achieving state-of-the-art results (MERITY; KESKAR; SOCHER, 2017). However, they present problems such as lack of parallelization, thus resulting in long training time and context loss in long inputs (VASWANI et al., 2017a). Transformers were introduced to solve these problems and have been surpassing state-of-the-art LSTM and GRU results (VASWANI et al., 2017a; ZEYER et al., 2019).

### 2.3.1 Transformers

Transformers is a relatively new seq2seq architecture in Deep learning, and it has been achieving state-of-art results in Natural Language Processing (NLP). Transformers network is an encoder/decoder model that tackles a couple of problems mentioned previsouly that

RNNs have: lack of parallelization (RNN/LSTM and GRU process word by word sequentially resulting in long training time); and difficulty to establish relationships between words far apart. Transformers can parallelize inputs and retain relationships between words using multi-headed attention, and by not using recurrent layers, its calculations are a lot faster (VASWANI et al., 2017a).

Figure 14 – Overview of the Transformers architecture. From left to right: Encoder layer; Decoder layer; Multi-head attention layer; and Scaled dot-product layer.




Source: https://deepfrench.gitlab.io/deep-learning-project/ (2022)

As we can see in Figure 14, the Transformers architecture is based on stacks of encoders and decoders where each is comprised of a multi-head attention mechanism and a feed-forward layer. Because it does not use recurrences and convolutions, information regarding the position of the input in the encoder and decoder is added using the following equation:

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{model}})$$

$$PE_{(pos,2+i)} = cos(pos/10000^{2i/d_{model}})$$

It is through attention layers, more specifically multi-head self-attention layers, that Transformers can retain context-related information. The concept of multi-head self-attention is to calculate the attention score for every word. For this, the input is split across multiple heads and in every head and goes through a series of linear layers to obtain its output. This is better visualized in the "zoom-in"figures in Figure 14.

This architecture was adapted to be used in input image classification problems, with the name Vision Transformers (DOSOVITSKIY et al., 2020) and surpassed consolidated CNN architectures results. However, this architecture required fixed-length input and 16x16 images

to be adequately trained, which did not work for our needs since we need variable-length input and images with size 480x320.

### 2.3.2 Why Transformers?

Transformers has been achieving state-of-the-art results surpassing LSTM and GRU seq2seq Natural language Processing (NLP) results (VASWANI et al., 2017a). As we mentioned before, works from (CAMGOZ et al., 2020a) and (YIN; READ, 2020) used Transformers for sign language recognition achieving interesting results. Considering its approach of using attention calculations instead of recurrence as in Recurrent Neural Network (RNN)s, we wanted to verify its performance when applied to video sequence recognition, especially in Libras sign language recognition.

# 3 LITERATURE REVIEW

This literature review investigates the techniques used for gesture and facial (including gaze) recognition in the context of sign languages in recent years (2009 to 2021). We focused on this recent period due to the advancement of technology providing faster hardware, better cameras, and new promising algorithms for recognition. We cover static and dynamic gestures, simple and complex backgrounds, facial and gaze expressions, and the use of special hardware, particularly in a mobile context. The questions addressed in this review are: which techniques are used to perform feature extraction and image classification of static and dynamic sign language gestures? Do these techniques consider facial expressions and body language? What are the recognition rates obtained? What type of hardware is being used? What type of image background is addressed (simple or complex)? A total of 50 papers were analyzed and compared.

In section 3.1, we describe the search and selection process. Then, we give an overview of how the papers analyzed answered our research questions. Next, section 3.2 discusses the papers, including the use of special hardware, the context of sign languages, facial expression recognition, and publicly available datasets. Lastly, section 3.3 presents our final thoughts and section 3.4 points to the suggested future research directions.

## 3.1 OVERVIEW OF TECHNIQUES AND RESULTS

Firstly we searched IEEE and Science Direct using the string "Sign language recognition AND mobile devices". This automatic search was manually complemented to include more techniques and approaches. Although the main focus of this review is the mobile context (due to the potential application of using the proposed systems in real-world scenarios of sign language communication), papers on relatively recent techniques such as CNN (LECUN et al., 1998a) and Hierarchical Temporal Memory (HTM) (HAWKINS; BLAKESLEE, 2007) were included even though they are not developed based on mobile technologies but dealt with gesture recognition for sign languages. These works were included due to the relevance and potential of such techniques for the field. Recognition of facial expressions, which is an important aspect of sign languages, is still a smaller field when compared to gesture recognition, and the papers found were also not applied to the mobile context.

Some of the papers analyzed did not recognize gestures from a real sign language but mentioned it as a possible application field. These papers were included because their techniques for gesture recognition can potentially be adjusted for sign languages.

Theses, dissertations and other long works, literature reviews, and papers not available in English were not considered. In addition, papers related to sign language but did not present feature extraction or classification techniques were also not considered. Such papers include the game design for sign language learning, usability testing for sign language applications, mobile application to teach sign language to non-deaf persons, description of methodologies used in Human-Robot Interaction, and finger direction detection.

For gesture recognition, 25 different techniques (Table 1) were used to extract information for the training/classification process. The most popular techniques were sensor and skin segmentation. For a machine to recognize an object properly, unwanted information must be removed from the image. This is crucial because unwanted information can influence the classification result negatively.

Table 2 displays the techniques used to recognize static gestures. The most commonly used were brute force comparison (comparison of a given pattern against saved patterns), SVM (CORTES; VAPNIK, 1995) and KNN (ALTMAN, 1992). Some papers did not mention which technique was used to classify static gestures or did not classify this type of gesture.

Papers that recognized dynamic gestures used 12 different techniques as shown in Table 3: Brute force, Multi-layer Perceptron (MLP) (ROSENBLATT, 1961), KNN, HTM and Mahalanobis (MAHALANOBIS, 1936) and Euclidean distance (BREU et al., 1995). Eight papers achieved a classification result higher than 90% and one paper informed that had results between 70% and 97%.

Table 4 displays an overview of the papers regarding their classification results, according to the type of recognition and size of the dataset. The numbers show that most works obtained a high accuracy, but they only recognize a few gestures, which does not correspond to a real sign language context. It is hard to make a more precise comparative analysis of the results due to a lack of standardization of techniques (Tables 1, 2 and 3) and datasets. For instance, it is hard to compare the results from (VINTIMILLA et al., 2016), which achieved a 90% accuracy with backpropagation but only recognized one letter (A) and (THALANGE; DIXIT, 2016) with 98% accuracy recognizing numbers from 0 to 9 using Statistical Measures and Orientation Histograms. They have different datasets, different test images, and different classification techniques. In addition, for some works, results varied within a fairly large range (for example,

(PAULSON; CUMMINGS; HAMMOND, 2011) report results from 51.7% to 96.9%). It is also important to note that, although most works used the word "gesture", not all gestures recognized were from real sign languages (see sections 3.2.1.2 and 3.2.2.2). Table 4 indicates the works that specified the exact type of gesture recognized (e.g., pointing gestures, number of fingers raised, trajectories, sentences, etc.). Four papers ((RAHEJA; SUBRAMANIYAM; CHAUDHARY, 2016), (HAKKUN; BAHARUDDIN et al., 2015), (YANG, 2013) and (MADHURI; ANITHA; ANBURAJAN, 2013)) did not inform classification results and therefore were not included in this table. A total of 9 papers used a publicly available dataset, 21 created their dataset, and 11 did not inform their dataset. This represents 23%, 53%, and 28% of the total of papers here analyzed, respectively.

Table 5 displays the various types of hardware that were used. The most commonly used was some glove mechanism with sensors connected via Bluetooth to a mobile device followed by Kinect[1] sensor camera from Microsoft[2]. On the other hand, a couple of words such as (WADHAWAN; KUMAR, 2020), (COSTER; HERREWEGHE; DAMBRE, 2020a), (BARROS et al., 2017) and (KO; SON; JUNG, 2018a) did not use any special hardware, which makes features extraction a lot harder because unnecessary information will have to be removed by software.

Table 6 shows which papers used a simple or complex background. The type of background interferes with the feature extraction process because complex background introduces much information that needs to be removed. Eighteen papers did not inform the type of background.

As for facial expression recognition, two techniques were used: Recursive Principal Component Analysis (RPCA) (VOEGTLIN, 2005) and Ada-boost (FREUND; SCHAPIRE, 1995) feature selector. MLP and Random Forest (HO, 1995) were used to classify.

From the three papers that recognized gaze gestures, all papers used HTM and their variants: traditional, extended, and optimized.

[1] https://www.xbox.com/en-US/xbox-one/accessories/kinect
[2] https://www.microsoft.com

Table 1 – Overall techniques used for feature extraction related to the authors that used it.

| Technique for feature extraction | Who used it? |
|---|---|
| Sensors (Flex, G, Accelerometer, Tilt, Inertial) | (BAJPAI et al., 2015); (SEYMOUR; TŠOEU, 2015); (KOSMIDOU; HADJILEONTIADIS, 2009); (KAU et al., 2015); (SHUKOR et al., 2015); (PAULSON; CUMMINGS; HAMMOND, 2011); (JUNKER et al., 2008); (DEVI; DEB, 2017) |
| Skin color segmentation | (MADHURI; ANITHA; ANBURAJAN, 2013); (LUO; WU; LIN, 2015); (ALMEIDA; GUIMARÃES; RAMÍREZ, 2014); (SHARMA; VERMA, 2015); (CARIDAKIS et al., 2010); (HARTANTO; KARTIKASARI, 2016) |
| Filters (Canny, Otsu) | (JIN; OMAR; JAWARD, 2016); (PATTANAWORAPAN; CHAMNONGTHAI; GUO, 2016) |
| Mahalanobis distance | (PARK; LEE, 2011); (SHANABLEH; ASSALEH, 2011) |
| Position of controller | (KHAMBADKAR; FOLMER, 2014) |
| Cellular Neural Nerwork | (YANG, 2013) |
| Binarization in HUV color space | (PRASUHN et al., 2014) |
| Gray scale image Binarization | (VINTIMILLA et al., 2016) |
| Morphological operations | (PRASUHN et al., 2014) |
| HOG | (PRASUHN et al., 2014); (SHAH et al., 2021) |
| Hand Centroid Calculation | (MADHURI; ANITHA; ANBURAJAN, 2013);(SANTOS; FERNANDES; BEZERRA, 2015) |
| Gabor feature, RAW and LBP | (LUO; WU; LIN, 2015) |
| SURF | (JIN; OMAR; JAWARD, 2016) |
| Viola and Jones Robust real-time object detection | (HAKKUN; BAHARUDDIN et al., 2015); (ELONS; AHMED; SHEDID, 2014) |
| Electromyogram | (KOSMIDOU; HADJILEONTIADIS, 2009) |
| Jones and Rehg's histogram color | (PARK; LEE, 2011) |
| Orientation Histogram and Statistical Measures | (THALANGE; DIXIT, 2016) |
| Wavelet features | (THALANGE; DIXIT, 2016) |
| Image subtraction | (AL-ROUSAN; ASSALEH; TALA'A, 2009) |
| Discrete Cosine Transform | (AL-ROUSAN; ASSALEH; TALA'A, 2009); (RAO; KISHORE, 2017) |
| Sparse-auto-encoder | (LI et al., 2015) |
| PCA | (LI et al., 2015) |
| Normalization | (WADHAWAN; KUMAR, 2020) |
| Hand Keypoints | (COSTER; HERREWEGHE; DAMBRE, 2020a); (XIE et al., 2015) (LEE et al., 2021) |
| Convexity approach | (BARROS et al., 2017) |
| CNN | (SHARMA; KUMAR, 2021); (BASNIN; NAHAR; HOSSAIN, 2021) |

Source: Created by the author (2022)

Table 2 – Overall techniques to recognize static gestures related to authors that used it.

| Technique for static gesture recognition | Who used it? |
|---|---|
| Brute force comparison | (KHAMBADKAR; FOLMER, 2014); (BAJPAI et al., 2015); (PRASUHN et al., 2014); (SHUKOR et al., 2015); (SHARMA; VERMA, 2015) |
| SVM | (SEYMOUR; TŠOEU, 2015); (PAN et al., 2016); (JIN; OMAR; JAWARD, 2016); (LI et al., 2015); (SHAH et al., 2021) |
| KNN | (HAKKUN; BAHARUDDIN et al., 2015); (PAULSON; CUMMINGS; HAMMOND, 2011) |
| Neural Network with Log-sigmoid activation function | (SEYMOUR; TŠOEU, 2015) |
| Neural Network with Symmetric Elliott activation function | (SEYMOUR; TŠOEU, 2015) |
| Difference comparison | (MADHURI; ANITHA; ANBURAJAN, 2013) |
| Hand Skeleton Recognition method with SVM (HSR + SVM) | (LUO; WU; LIN, 2015) |
| Feed-Forward Backpropagation Neural Network | (THALANGE; DIXIT, 2016); (HARTANTO; KARTIKASARI, 2016); (VINTIMILLA et al., 2016) |
| Convolutional Neural Network - CNN | (PIGOU et al., 2014); (NAGI et al., 2011); (WADHAWAN; KUMAR, 2020) |
| Discrete Wavelet Transform | (PATTANAWORAPAN; CHAMNONGTHAI; GUO, 2016) |
| Area Level Run Lengths | (PATTANAWORAPAN; CHAMNONGTHAI; GUO, 2016) |
| Softmax Classifier | (LI et al., 2015) |
| Hierarchal Temporal Memory | (KAPUSCINSKI, 2010) |
| RNN/GRU/LSTM/Transformers Network | (COSTER; HERREWEGHE; DAMBRE, 2020a) |
| CNN + LSTM | (BASNIN; NAHAR; HOSSAIN, 2021) |

Source: Created by the author (2022)

Table 3 – Overall techniques used to recognize dynamic gestures related to the authors that used it. ∗metric distance used as classifier.

| Techniques for dynamic gesture recognition | Who used it? |
|---|---|
| Brute force comparison | (KAU et al., 2015); (SHUKOR et al., 2015) |
| Multi-Layer Perceptron | (NAGI et al., 2011);(ELONS; AHMED; SHEDID, 2014) |
| Mahalanobis distance* | (KOSMIDOU; HADJILEONTIADIS, 2009); (RAO; KISHORE, 2017) |
| KNN | (SHANABLEH; ASSALEH, 2011) |
| Euclidean distance* | (VINTIMILLA et al., 2016) |
| Hierarchical Temporal Memory | (KAPUSCINSKI; WYSOCKI, 2009); (ROZADO; RODRIGUEZ; VARONA, 2012a) |
| Dynamic Time Warping* | (CELEBI et al., 2013);(BARROS et al., 2017) |
| 3D CNN | (HUANG et al., 2015); (SHARMA; KUMAR, 2021) |
| LSTM | (LIU; ZHOU; LI, 2016) |
| CNN + LSTM | (CUI; LIU; ZHANG, 2017) |
| LSTM + KNN | (LEE et al., 2021) |
| Hidden Markov Mode | (BARROS et al., 2017) |
| RNN/GRU/LSTM/Transformers Network | (KO; SON; JUNG, 2018a) |
| DTW + HMM Hybrid Network | (SANTOS; FERNANDES; BEZERRA, 2015) |

Source: Created by the author (2022)

Table 4 – Overall view of authors divided by the type of recognition performed, accuracy, dataset size, dataset ownership.

| Recognition of: | Accuracy(%) | Dataset Size | Dataset Ownership | Reference |
|---|---|---|---|---|
| Static Gestures | 74.69 - 98.17 | 10 | - | (THALANGE; DIXIT, 2016) |
| | 74.7 | 100 | - | (COSTER; HERREWEGHE; DAMBRE, 2020a) |
| | 75 | 36 | Own | (KHAMBADKAR; FOLMER, 2014) |
| | 77 - 92 | 8 | Own | (KAPUSCINSKI, 2010) |
| | 81.9 - 92.2 | 6 (number of fingers raised) | Own | (LUO; WU; LIN, 2015) |
| | 83 | 36 | Own | (BAJPAI et al., 2015) |
| | 89.38 | - | - | (PATTANAWORAPAN; CHAMNONGTHAI; GUO, 2016) |
| | 90 | 32 (numbers and letters) | - | (VINTIMILLA et al., 2016) |
| | 90.7 - 91.7 | 20 | Public | (PIGOU et al., 2014) |
| | 91.66 | 24 | Own | (HARTANTO; KARTIKASARI, 2016) |
| | 94 - 99 | 31 | Own | (SEYMOUR; TŠOEU, 2015) |
| | 99.8 | 26 (letters) | Own | (PAN et al., 2016) |
| | 94 | 36 (letters and numbers) | Public | |
| | 95.6 - 98.6 | - | - | (SHARMA; VERMA, 2015) |
| | 98.8 - 99.9 | 10 | Own | (WADHAWAN; KUMAR, 2020) |
| | 97.13 | 16 | - | (JIN; OMAR; JAWARD, 2016) |
| | 91.98 | 6633 | Own | (SHAH et al., 2021) |
| | 88 | 13400 | Own | (BASNIN; NAHAR; HOSSAIN, 2021) |
| Dynamic Gestures | 61.3 | 5675 (sentences) | Public | (CUI; LIU; ZHANG, 2017) |
| | 64 | 500 (words) | Own | (LIU; ZHOU; LI, 2016) |
| | 86 | 100 (words) | Own | |
| | 52.55 - 98.43 | 7 (gestures) | Public | (BARROS et al., 2017);(SANTOS; FERNANDES; BEZERRA, 2015) |
| | 85 | 30 | Public | (NEVEROVA et al., 2014) |
| | 86.5 | 100 | Own | (KO; SON; JUNG, 2018a) |
| | 70 - 97 | 34 | - | (ALMEIDA; GUIMARÃES; RAMÍREZ, 2014) |
| | 90.6 - 97.4 | 30 | Own | (AL-ROUSAN; ASSALEH; TALA'A, 2009) |
| | 91 | 95 | Public | (ROZADO; RODRIGUEZ; VARONA, 2012a) |
| | 94 | 25 (words) | Own | (HUANG et al., 2015) |
| | 94 | - | - | (KAU et al., 2015) |
| | 94 | 101 | Public | (KAPUSCINSKI; WYSOCKI, 2009) |
| | 96 | 6 (number of fingers raised) | Own | (NAGI et al., 2011) |
| | 96.7 | 8 | Own | (CELEBI et al., 2013) |
| | 97.5 | 12 | Public | (SANTOS; FERNANDES; BEZERRA, 2015) |
| | 100 | 30 (gesture trajectory) | Own | (CARIDAKIS et al., 2010) |
| | 96 | 3300 | Public | (SHARMA; KUMAR, 2021) |
| Static and Dynamic Gestures | 91.6 | 2600 | Own | (LEE et al., 2021) |
| | 78.33 - 95 | 3 (letters)<br>3 (numbers)<br>3 (words) | Own | (SHUKOR et al., 2015) |
| Gestures (Type Not Informed) | 39 - 47 | 19 | - | (PRASUHN et al., 2014) |
| | 51.7 - 96.9 | - | - | (PAULSON; CUMMINGS; HAMMOND, 2011) |
| | 87 | 23 | Own | (SHANABLEH; ASSALEH, 2011) |
| | 89 - 99 | 1656 (pointing gestures) | Own | (PARK; LEE, 2011) |
| | 90 | 19 | - | (RAO; KISHORE, 2017) |
| | 93 | 60 | Own | (KOSMIDOU; HADJILEONTIADIS, 2009) |
| | 93.75 | 14 | Own | (DEVI; DEB, 2017) |
| | 97.4 - 98.4 | 8 | Own | (JUNKER et al., 2008) |
| | 98.74 - 99.05 | 24 (letters) | - | (LI et al., 2015) |
| Facial Expressions | 55 | 40 | Public | (KOLLER; NEY; BOWDEN, 2015) |
| | 93 - 97 | 9 | Public | (BHUVAN et al., 2016) |
| | 99 | 9 | Public | (UDDIN, 2015) |
| Dynamic gestures with Facial Expressions | 98 | 6 | Own | (ELONS; AHMED; SHEDID, 2014) |
| Gaze Gestures | 96 | 50 | Own | (ROZADO; RODRIGUEZ; VARONA, 2011) |
| | 98 | 10 | Own | (ROZADO; RODRIGUEZ; VARONA, 2012b) |
| | 98 | 10 | Own | (SRUTHI et al., 2017) |

Source: Created by the author (2022)

Table 5 – List of special hardware used for gesture recognition and the authors that used it.

| Special hardware | Who used it? |
|---|---|
| Glove | (BAJPAI et al., 2015); (SEYMOUR; TŠOEU, 2015); (KOSMIDOU; HADJILEONTIADIS, 2009); (KAU et al., 2015); (SHUKOR et al., 2015); (PAULSON; CUMMINGS; HAMMOND, 2011); (DEVI; DEB, 2017); (ROZADO; RODRIGUEZ; VARONA, 2012a) |
| Colored Glove | (NAGI et al., 2011); (SHANABLEH; ASSALEH, 2011) |
| Kinect | (ALMEIDA; GUIMARÃES; RAMÍREZ, 2014); (LI et al., 2015) (LIU; ZHOU; LI, 2016); (HUANG et al., 2015) (PIGOU et al., 2014); (CELEBI et al., 2013) |
| Robot | (LUO; WU; LIN, 2015);(PARK; LEE, 2011) |
| PS3 + Accessories | (KHAMBADKAR; FOLMER, 2014) |
| Mobile Devices | (JIN; OMAR; JAWARD, 2016); (RAO; KISHORE, 2017) |
| Leapmotion | (LEE et al., 2021) |

Source: Created by the author (2022)

Table 6 – Overall types of background and the list of authors that used it.

| Background type | Who used it? |
|---|---|
| Simple | (PRASUHN et al., 2014);(NAGI et al., 2011);(ALMEIDA; GUIMARÃES; RAMÍREZ, 2014); (AL-ROUSAN; ASSALEH; TALA'A, 2009);(PATTANAWORAPAN; CHAMNONGTHAI; GUO, 2016) (THALANGE; DIXIT, 2016);(SHARMA; VERMA, 2015) (VINTIMILLA et al., 2016);(RAO; KISHORE, 2017) (KAPUSCINSKI; WYSOCKI, 2009);(WADHAWAN; KUMAR, 2020) (COSTER; HERREWEGHE; DAMBRE, 2020a);(BARROS et al., 2017) (KO; SON; JUNG, 2018a); (BASNIN; NAHAR; HOSSAIN, 2021); (SHAH et al., 2021) |
| Complex | (PAN et al., 2016); (JIN; OMAR; JAWARD, 2016) |

Source: Created by the author (2022)

## 3.2 DISCUSSION

This section first discusses gesture recognition in two groups: papers-based and not based on special hardware. Next, we present works that performed facial expressions (including gaze gestures) recognition, and lastly, we list the publicly available datasets used by the papers analyzed.

### 3.2.1 Papers based on Special Hardware

#### *3.2.1.1 Sign Language Context*

(LIU; ZHOU; LI, 2016) proposed a sign language recognition system using LSTM (HOCHREITER; SCHMIDHUBER, 1997) using skeleton joint (left hand, right hand, left elbow, and right elbow) trajectories provided by Kinect sensor camera (color image and depth information are discarded). The architecture consisted of seven layers, including the input layer. The first layer is the input layer fed with a 12-dimensional feature vector representing the skeleton joints of four 3D spatial coordinate vectors. The next layer is an LSTM layer with 512 dimensions. After the LSTM layer, there are two Fully Connected (FC) layers. The first FC contains 512 neurons, and the second contains 100 neurons that correspond to 100 classes which are then followed by a softmax layer and by a pooling layer. The last layer is the output layer predicting the class of the sequence. The authors did not inform how they selected their parameters.

The authors created two datasets: 100 isolated Chinese sign language words and another with 500 sign words. Five signers performed each gesture five times for the first dataset, resulting in 25000 images. For the second dataset, 50 signers performed each gesture five times, resulting in 125000 images. All images were captured using the Kinect sensor.

The authors achieved a classification rate of 86% using the dataset with 25K images and 64% with 125K images.

(HUANG et al., 2015) proposed a 3D CNN (JI et al., 2013) to recognize sign language by extracting spatial-temporal features from a video using Kinect sensor camera. Color channels, depth, and trajectory features are used as input to the proposed CNN. The 3D approach uses 3D filters instead of a regular 2D filter, capturing spatial and temporal information when convolving through an image.

The proposed architecture consists of eight layers, including the input layer. After the input

layer, the following five layers are convolution (C1), sub-sampling (S1), convolution (C2), sub-sampling (S2), and convolution (C3). Finally, the last two layers comprising the output layer are two FC layers.

To perform the 3D convolution, the authors stack image frames to form a "cube"for the five types of features: R, G, B, depth, and skeleton to use the 3D filter. The C1 layer uses 50 7x7x5 filters, where 7x7 is the spatial dimension, and 5 is the temporal dimension. In addition, a 2x2 sub-sampling is applied to make the network more resistant to small spatial distortion. When going through the architecture, new filters and sub-sampling are applied with the following sizes: 50x7x7x3, 2x2, 10x5x5x3. The first FC layer consists of 200 nodes of size 1x1, and the final layer (output layer) consists of 25 units corresponding to different sign words. The authors did not inform how they selected their parameters.

The authors made their dataset which consisted of 25 words that are widely used in daily life. Each word is articulated three times by nine signers.

This 3D CNN approach achieved a 94% classification result.

(CUI; LIU; ZHANG, 2017) developed a continuous Sign language recognition system using RNN. The authors employ a CNN and a RNN with the LSTM module to learn the mapping of feature sequences to sequences of the gestures.

The architecture consists of a CNN with temporal convolution and pooling for spatial and local temporal feature extraction, a bidirectional LSTM for global sequence learning, and a detection network to refine the sequence learning. Bidirectional LSTM computes the hidden state sequence by combining the output sequences of LSTM by iterating forwards and backward. The detection network combines the temporal convolution operations on the spatio-temporal features. According to the authors, this combination acts like a sliding window along with the feature sequences. The result goes to the softmax layer to get the detection scores.

The authors used a publicly available RWTH-PHOENIX-Weather multi-signer 2014 dataset. The results were presented in Word Error Rate (WER) which is used in the scope of continuous Sign language recognition. The authors achieved 61.3% accuracy.

(BAJPAI et al., 2015) proposed a translation system where static ASL gestures were translated into spoken English using a glove and a mobile device connected via Bluetooth. Instead of using gyroscope sensors, the authors used flex sensors and an Arduino Mega microcontroller board. Flex sensors provide voltage information when bent, while gyroscopes provide 3-axes coordinate data when moved around. The flex sensors are put along with the finger and, depending on the bent curvature, create a resistance caused by the electrons, from 4.1v to

5.0v. These sensors are combined with a Darlington pair transistor circuitry so that the 0.9v deflection is amplified to a more extensive range. The output of the circuit is filtered using a filter circuit and then converted into digital form. This conversion is done with the help of the Analog to Digital Converter - ADC pins from the AVR-2560 micro-controller.

The database is created by using this glove and storing each gesture's voltage. When a gesture is performed to be recognized, a matching voltage within the database is retrieved. First, the authors search for a group of voltages within their database, therefore not using any AI technique. After that, the glove sends the resulting label for the input voltages to the mobile phone, and through text-to-speech software, the result is spoken to the user. Using this approach, the authors achieved an 83% classification rate. One exciting functionality the authors implemented was creating a custom gesture, allowing the user to create a gesture for his or her name, for example.

(SEYMOUR; TŠOEU, 2015) proposed a mobile recognition system to recognize South African Sign Language (SASL). The idea of this work is very similar to (BAJPAI et al., 2015) with flex sensors being used within a glove and connected to a mobile device via Bluetooth. However, (SEYMOUR; TŠOEU, 2015) used 3 types of neural network models based on a Log-Sigmoid activation function; a Symmetric Elliott activation function (ELLIOTT, 1993); and SVM. According to the authors, although Log-Sigmoid (HAN; MORAGA, 1995) is the most common activation function, Symmetric Elliott reported faster training time due to its symmetric shape and gradual change in gradient. These neural networks were implemented using the Encog1[3] machine learning framework with JAVA programming language. In order to find the optimal number of neurons in the hidden layer for the neural networks that used the Log-Sigmoid and Symmetric Elliott activation functions, the authors chose an interval from 2 to 38 and added six neurons in every iteration. The number of neurons that produced the highest classification rate was set as their optimal number. For the Log-Sigmoid network, 20 neurons produced the highest classification rate, and for the Symmetric Elliott, 32 neurons produced the best result. The $C$ and $\lambda$ parameters of the SVM were found using a grid search approach. The values obtained were $C$=112 and $\lambda = 1.2$.

Five participants created the dataset by performing 31 static gestures in SASL using the glove. Each gesture resulted in 14 features captured by the flex sensors. The authors performed the classification using Log-Sigmoid and Symmetric Elliot neural networks in the mobile device while SVM was performed using a computer. According to the authors, SVM performed better

[3] http://www.heatonresearch.com/encog/

on a computer. The authors obtained a 94% classification rate using the Log-Sigmoid neural network, 96% using Symmetric Elliott neural network, and 99% using SVM. Nevertheless, the results for the SVM were not shown in the same environment as the other networks (mobile device).

(KAU et al., 2015) proposed a glove to recognize dynamic gestures in Taiwanese Sign Language. The authors also used flex sensors to detect finger flexion but combined with Gyroscope-sensor to detect palm orientation and a gyroscope to detect motion trajectory. This glove is connected to a mobile device via Bluetooth, displaying the final result.

The system architecture is entirely based on the information from the sensors, which is stored in different registries that are checked every clock cycle. The authors used one flag and four registers: Register(A) records the former posture; Register(a) records how long a posture persists; Register(B) records the orientation of the palm; and Register(C) records the motion trajectory. In addition, the flag indicates if a gesture has been recognized or not. All of the information of the sensor gives 20 bits of data, of which 10 bits are stored in Register(A), 6 in Register(B), and the remaining in Register(C).

The accuracy achieved by the authors was 94%, recognizing five dynamic gestures. However, this work did not use any machine learning technique.

(SHUKOR et al., 2015) also proposed a glove-based system (Figure 15), to recognize Malaysian Sign Language, comprised of 10 tilt sensors to capture finger flexion, an accelerometer for hand motion, and an Arduino microcontroller with a Bluetooth module to connect to a mobile device and send the recognized gesture.

Due to the limited capacity of the microcontroller's memory, the authors only recognized three letters and three numbers (represented by static gestures); and three words (represented by dynamic gestures) performed by four different persons. Nevertheless, the work achieved an average of 89% classification rate, 95% for letters, 93.33% for numbers, and 78.33% for words. The authors did not use any machine learning technique, thus making their recognition solely based on number comparison.

(DEVI; DEB, 2017) proposed a glove-based system connected to a mobile device to translate Hindi sign language to speech. The glove is equipped with Flex sensors to extract gesture information and an Arduino micro-controller to provide Bluetooth communication with the mobile device. In addition, euclidean distance is used to classify gestures despite the fact that it is a distance metric.

The dataset used by the authors consisted of 14 training sets, one for each gesture. They do

Figure 15 – Glove used from (SHUKOR et al., 2015).

Source: SHUKOR et al. (2015)

not inform how many examples per training set nor the type of gestures. Information from the sensors is processed with Principal Component Analysis (PCA) (JOLLIFFE, 2002) then stored in the microcontroller to populate the training set. Euclidean distance is used to calculate the distance between the input and the training sets. The classification result is in training set with the smallest distance to the input.

The authors did not inform how their testing phase was performed. However, the initial classification result obtained an 84.75%, and after adjusting the position of a flex sensor, the classification result increased to 93.75%.

(KOSMIDOU; HADJILEONTIADIS, 2009) proposed a gesture recognition system composed of a five-channel electromyogram and a 3D accelerometer (Figure 16). The electromyogram, located along with the muscles of the forearm, is responsible for capturing muscle contraction information, while the accelerometer, located near the wrist, is responsible for capturing spatial information of the hand. The data was analyzed using Intrinsic Mode Entropy (IMEn) to recognize 60 Greek Sign Language gestures. The electromyogram captures the motion of the wrist and fingers, while the 3D accelerometer captures the movement of the arm. Their dataset was composed of 60 gestures executed three times with a 2-second interval between each repetition by three signers.

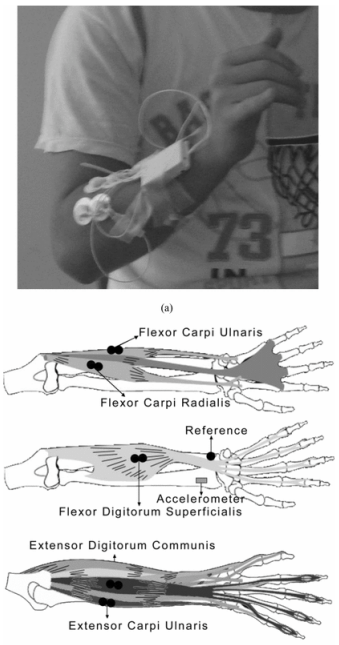
The classification is based on the following configurations:

1. Estimation of the best ($k$, $m$) pairs per single-signer.

   This part calculates the IMEn for each signer based on all the data from sensors based on the following formula:

$$IMEn(k, m, r) = SampEn(C^k_{IMF}(t), m, r), k = 1, ..., K + 1$$

Figure 16 – Hardware and schematics from (KOSMIDOU; HADJILEONTIADIS, 2009).




Source: KOSMIDOU; HADJILEONTIADIS (2009)

where $k$ is a gesture index, $m$ is the window length and $r$ is the tolerance. A discriminant analysis was calculated to find which pairs provide the maximum classification accuracy per signer.

2. Classification accuracy based on a single user.
   Based on the best pairs found in step 1, 70% and 30% of the optimal pairs were used for training and classification. This was randomly executed 100 times, averaging the classification result. According to the authors, a robust estimate of the classifier was obtained by doing this.

3. Estimation of the best ($k$, $m$) pairs per signer-pair.
   The optimal pairs found in step 1 were grouped into pairs of signers. Then, another discriminant analysis was calculated to reveal which pairs provided the maximum classification accuracy per signer pair.

4. Classification accuracy based on a signer-pair.
   Similar to step 2, resulting pairs from step 3 was used to evaluate the classification accuracy.

The authors achieved a 93% classification rate using concepts specific to the sensors used. The classification technique was briefly mentioned, not revealing how it was used, other than

the Mahalanobis distance.

(SHANABLEH; ASSALEH, 2011) proposed a colored glove and used KNN, linear discriminant functions (MARTÍNEZ; KAK, 2001) and Polynomial classifiers (GRAF; KRESSEL; FRANKE, 1997) to recognize Arabic Sign Language (ArSL). Using Discrete Cosine Transform (DCT) (AHMED; NATARAJAN; RAO, 1974) as feature extractor, feature vectors are formed by applying Zonal Coding to the DCT coefficients with varying cutoff values.

The colored glove was used to segment the images easily. Samples of the colored glove were used to estimate the mean and covariance matrix, and pixels similarities were calculated with Mahalanobis distance. Three signers performed 23 gestures with 50 repetitions over three sessions, resulting in 3450 video segments.

Motion information was extracted using successively the Accumulated Prediction Errors or Image Differencing, expressed by the following equation:

$$AD_{g,j} = \sum_{j=1}^{n-1} \partial_j(|I_{g,i}^{(j)} - I_{g,i}^{(j-1)}|)$$

where *n* is the total number of images in the *ith* repetition of a gesture at index *g*. $\partial_j$ is a binary threshold function of the *jth* frame. The next step is to transform the images from the AD process to frequency domain using DCT by using the following equation:

$$F(u,v) = \frac{2}{\sqrt{(MN)}} C(u)(v) f(i,j) cos(\frac{\pi u}{2M} \cdot (2i+1))$$

$$xcos(\frac{\pi v}{2N} \cdot (2j+1))$$

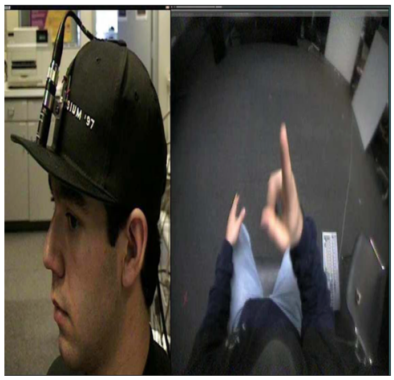
where *N x M* are the dimensions of the input image, $F(u,v)$ is the DCT coefficient at row $u$ and column $v$ of the DCT matrix and C($u$) is the normalization factor.

Classification with KNN was used with two distance measures: Euclidean distance and Correlation Factor. The classification decision using correlation factor is based on finding the maximum correlation between two vectors, which is defined as follows:

$$argmax_{x_{train_j}}(\frac{zscore(x_{test})^T \cdot zscore(x_{train_j})}{n-1})$$

where $x_{test}$ is a feature vector from the testing set and $x_{train_j}$ is the *jth* feature vector from the training set, and *n* is the dimensionality of the feature vector determined by the zonal cutoff, defined by the authors (although they did not explain how it was obtained).

Figure 17 – Environment setup from (MADHURI; ANITHA; ANBURAJAN, 2013).



Source: MADHURI; ANITHA; ANBURAJAN (2013)

KNN achieved the highest classification result, reaching 87%.

(MADHURI; ANITHA; ANBURAJAN, 2013) proposed a mobile gesture recognition to translate ASL to spoken English. The authors used a camera attached to a hat (Figure 17) under a controlled environment lighting.

Images were captured every 200 milliseconds analyzed in a 5-frame interval. All images went through a skin segmentation using a color thresholding method, the details of which were not informed by the authors. The feature extraction process calculates the centroid of the image. This centroid is represented by $x$ and $y$ coordinates and inserted into a 2 x *n* array (where *n* is the amount of images). This array is then transposed to a *n* x 2 arrays which are fed to the recognition system. The recognition of static signs is based on the finger's position in the bounding box. This recognition method was chosen because each letter of the alphabet and number has a unique combination of hand postures.

The authors did not provide any more information regarding their implementation, their dataset, or their results, although they stated that they used the LABVIEW software (DOMINGO; AKMELIAWATI; CHOW, 2007) to implement their work, mentioning that only static gestures were recognized in a controlled environment.

(ALMEIDA; GUIMARÃES; RAMÍREZ, 2014) proposed a feature selection system by joining Microsoft Kinect depth camera with nuiCaptureAnalyze[4] software to find a Region of Interest (ROI). After the background is removed using the Kinect camera, the nuiCaptureAnalyze software outputs the line-shaped image representing the person (Figure 18a). Based on the skeleton image, the coordinates of the hand can be obtained, and the skin detection algorithm is applied to segment the image. Figure 18b displays the segmentation results.

[4] http://nuicapture.com

Figure 18 – a) Background removal process and b) Segmentation results from (ALMEIDA; GUIMARÃES; RAMÍREZ, 2014).


Source: ALMEIDA; GUIMARÃES; RAMÍREZ (2014)

The authors stated that sign languages share a common phonological structure comprised of five elements: articulation points; configuration of the hands; type of movements of the hands; orientation; and facial and body expressions. Therefore, each gesture is a combination of any of these elements.

Based on these elements, seven features were extracted from the images. First, the two-dimensional Euclidean distance is calculated between each pixel of both hands and the shoulder's center (calculated using the skeleton image), and a 3-dimensional distance is obtained from the Kinect sensor to the signer to find another articulation point feature. Kinect calculates this 3-dimensional distance automatically. Velocity is calculated using the optical flow technique to find the movement and orientation elements. This feature provides vectors calculated using brightness difference and area of the hands. The brightness difference and area of the hands symbolize the configuration of the hand's phonological element. Corners' average position is calculated using Harris corner detection to find a combination of elements (articulation points, type of movement, orientation). Finally, detected lines are calculated using Hough transform (DUDA; HART, 1972) to find the feature configuration of the hands, and SURF (BAY et al., 2008) to detect several common pixels between frames, to populate the feature elements type of movement and orientation.

The authors classified 34 dynamic gestures, each having five images. Training and classification were done using SVM with kernel types: linear and Radial Basis Function (RBF). On the other hand, (ALMEIDA; GUIMARÃES; RAMÍREZ, 2014) focused on the feature extraction technique, not giving more details on training and classification. The overall classification result varied between 70% and 97%.

(LI et al., 2015) also used Microsoft Kinect camera to capture images, and proposed a feature learning system for RGBD images (Red Green Blue (RGB) + Depth image) based on Sparse-Autoencoder (SAE) (BENGIO et al., 2009) and PCA to increase accuracy. Their data set consisted of 24 letters in ASL, each having 500 images generated by five users resulting in

120K images. The authors used SVM and *softmax* to classify. Features are learned from the two types of images, concatenated and fed to a multiple-layer PCA to get the final feature vector.

The authors decided to use a sparse autoencoder with a bigger size than the input layer and added a sparsity constraint on the hidden units so that only a few neurons are activated. Suppose that the activation function of a given $jth$ neuron is given by:

$$\widehat{p} = \frac{1}{m}\sum_{m}^{i} = 1^{hj} \tag{3.1}$$

where *p* denotes the desired probability of being active (here p $<<$ 1). KL divergence is calculated using the following equation:

$$KL(p||\widehat{p_j}) = plog\frac{p}{\widehat{p}} + (1-p)log\frac{1-p}{1-\widehat{p_j}} \tag{3.2}$$

KL equation is used as a penalty measure between the desired (1) and actual distributions ($p_j$). The learning SAE model is formulated to solve the following optimization problem:

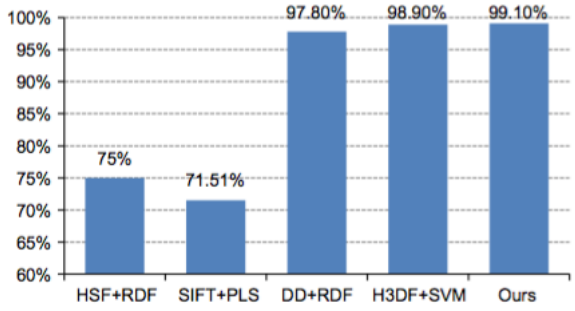
$$min_{w,b}(\sum_{i=1}^{m}||h_{w,b}(x^i) - y^i||^2 + \lambda(||W||^2) + \beta\sum_{k}^{j=1} KL(p||\widehat{p_j}) \tag{3.3}$$

The first term is the reconstruction cost, and the second term is weight decay which tends to suppress the magnitude of the weights W to reduce the risk of over-fitting. The parameters $\lambda$ and $\beta$ are used to balance reconstruction cost, weight decay, and a sparsity penalty. Backpropagation was used to train this model.

PCA was used to reduce the dimensionality, but its results are taken as input to SAE with a convolutional network. Then, 10 x 10 filters are used to learn the filters of the CNN instead of using the whole image as input. This approach speeded up the learning process and reduced the computational cost due to dimensionality reduction. Finally, Max-pooling was used to reduce dimensionality and learn shift-invariant features. Once this is done, PCA has applied again to both images (RGB and Depth) to reduce even more the dimensionality. Then vectors are concatenated and reduced again by PCA.

Images were normalized into the size of 96 x 96. A 10 x 10 patch was used to train SAE to learn the CNN filters. After PCA dimensionality reduction, final dimensions were 55 and 37 respectively for RGB and Depth images. One hundred neurons were used, thus producing 100 feature maps. A 10 x 10 max-pooling resulted in a 10K dimension vector which was reduced to a 6183 after applying PCA.

Figure 19 – Results comparison from (LI et al., 2015).




*HSF - hand shape feature, RDF - random decision forest, PLS - partial least square, DD - Depth difference and H3DF - histogram of 3D facet.*
Source: LI et al. (2015)

Figure 20 – Classification results from (VINTIMILLA et al., 2016).

| Selected Letter | Captured Letter | Answer |
|---|---|---|
| A | A | Letter A |
| A | A | Letter A |
| A | A | Letter A |
| A | A | Letter A |
| A | A | Letter A |
| A | O | Unknown frame |
| A | R | Unknown frame |
| A | Displays anything | Letter A |
| A | 5 | Unknown frame |
| A | Displays anything | Unknown frame |

Source: VINTIMILLA et al. (2016)

Using *softmax* classifier, the results achieved a 98,74% accuracy and using SVM achieved a 99,05% accuracy. Figure 19 displays the results' comparison with other works.

(VINTIMILLA et al., 2016) proposed a mobile application to translate static letters from the common manual alphabet from Spanish-speaking countries (A - Z) and numbers from 1 to 10 to text.

Images used were taken under controlled environment and lighting conditions for better feature extraction. Feature extraction consisted of 3 steps: grayscale transformation, image binarization, and re-sizing to a 40 x 40 image.

The authors used the backpropagation algorithm to perform training and classification. Information such as backpropagation parameters and the number of images were not provided. The classification was performed only for the gesture "A"and used as inputs gestures "A", "O", "R", "5", and a random gesture. The obtained results are displayed in Figure 20.

(RAO; KISHORE, 2017) proposed a mobile application to translate Indian sign language to text using the front-facing camera. The authors' main idea is to facilitate the communication between a signer and a non-deaf person. A signer records a video of himself in his sign language and sends it to the person he wants to communicate with. The video is decoded using computer

vision techniques and classified using AI algorithm, and the resulting text of the sign language video is displayed. The authors established a simple background where the user would use a t-shirt with the same color as the background.

In order to extract features from the video, the Gaussian filter is applied to smoothen each frame. Next, Sobel edge detection is applied to find the contours of the hand and head of the signer. Using these contours, hand, and head, a 2D DCT is calculated to find their energy value. PCA is then applied to reduce the feature vector.

Eighteen different signs performed by ten different signers were used for their dataset. The authors achieved a 90% classification accuracy using the Mahalanobis classifier.

(KAPUSCINSKI; WYSOCKI, 2009) proposed a sign language recognition system using HTM to classify Polish Sign language (PoSL) gestures. HTM is based on the neocortex of the human brain. For HTM, objects are represented by smaller parts organized in a spatio-temporal hierarchy, comprised of various nodes. Each node implements a common learning and memory function. HTM first finds the closest spatial grouping to which the input belongs, then finds common sequences in the temporal grouping.

These object representations are spread throughout the architecture and shared among other learning objects. Because objects are shared, HTM can recognize new objects that are composed of previous objects. Classification is performed through inference, where each label receives a percentage of likeliness to be the answer.

The authors used 101 words taken from a doctor's office and post office contexts. Skin detection was performed to extract relevant information. A 2D Gaussian distribution skin model is created using a person's hand, removing the background. Because HTM is a spatio-temporal algorithm, the center of gravity of the skin areas, ranges of motion, and size of skin areas, are used as input to the HTM. Simple background with controlled light and one-tone solid colored clothes was adopted to facilitate feature extraction. The authors used the software from Numenta[5], an HTM driven company, and did not provide details of their implementation. Nevertheless, the authors achieved a 94% accuracy.

Using the same HTM neural network, (KAPUSCINSKI, 2010) proposed a system to classify eight static gestures in any given rotation. Each gesture was performed four times and comprised 36 rotated images around its horizontal axis. The gestures were modeled in a graphical tool called Virtual Hand Studio[6].

[5] https://numenta.com/
[6] http://www.d-anatomystore.com/3d-virtual-hand-studio/

The architecture of the network was comprised of a sensor layer representing the spatial hierarchy and two temporal layers of size 16x16 and 8x8, respectively. Edge detection was performed, and its result was fed to the network's input. The sensorial layer only detected four edges: horizontal, vertical, and two slanted edges.

The authors made two experiments: 1) using the whole dataset (approximately 103,680 images); and 2) one image per rotation. A 92% accuracy was achieved in the first experiment, while 77% was achieved in the second.

(ROZADO; RODRIGUEZ; VARONA, 2012a) proposed an extended HTM to recognize dynamic gestures in the Australian Sign Language. According to the authors, the original HTM does not perform well under a time series instance (i.e., a gesture which is comprised of various images) because it performs classification at every single input image. This extension works as the last layer in the HTM to hold and compare sequences of spatio-temporal codified inputs to handle the temporal evolution of a dynamic gesture.

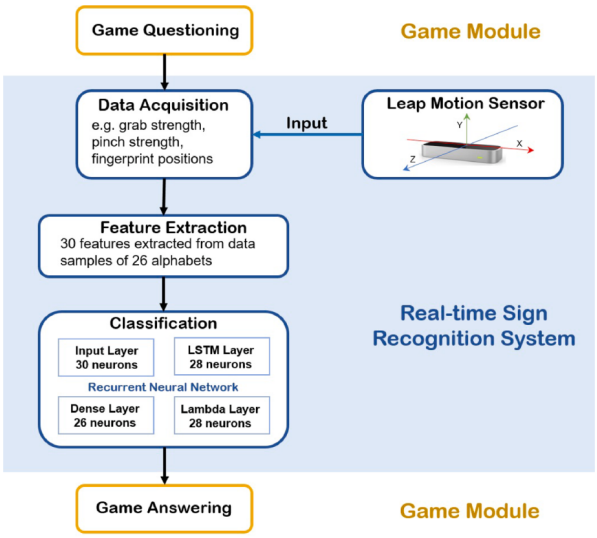
The idea of this extension is to create another layer above the classification layer capable of storing classification results. This sequence is compared with other stored sequences through a similarity threshold. The sequence with the highest similarity is the final result. The authors did not inform more details of this sequence creation. The similarity calculation is used in training as well. During training, the similarity threshold is used to see which sequence is the closest to any other sequence already stored and if the threshold is satisfied, the incoming sequence is stored. The authors used the Needleman-Wunsch algorithm (NEEDLEMAN; WUNSCH, 1970) as similarity measurement.

The authors used a publicly available UCI Australian Sign Language dataset[7]. This dataset was populated with information coming from a glove sensor-based, on which they based their entire experiment. This means that no new data was acquired to test this architecture, although they state that if a dataset contains a spatio-temporal structure, the proposed approach achieves a good classification accuracy. The authors achieved a 91% classification rate for 26 gestures.

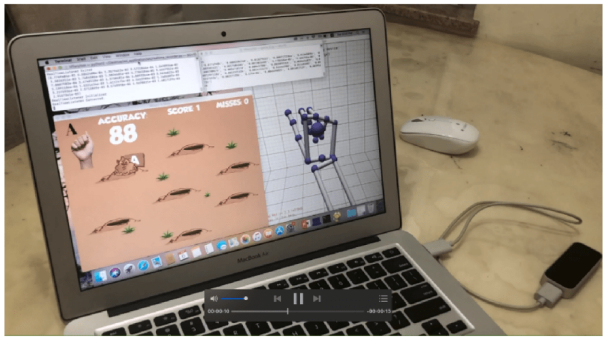
(LEE et al., 2021) proposed an interesting approach to Sign language recognition using LSTM combined with KNN. The authors proposed a game-based ASL alphabet recognition-based Leapmotion. Figure 21 displays the architecture screen capture of the game proposed by the authors.

[7] https://archive.ics.uci.edu/ml/datasets/Australian+Sign+Language+signs+(High+Quality)

Figure 21 – Game architecture and game screen proposed by (LEE et al., 2021).




(a) Game architecture using Leapmotion data as input and LSTM as classification



(b) Screen of the game proposed

Source: LEE et al. (2021)

As we can see in Figure 21 the authors use Leapmotion to capture hand data such as the position of palm center, sphere radius, grab strength, pinch strength, palm data, hand sphere radius, and finger data. The authors used approximately 30 features to create 100 examples for each alphabet word to create their dataset. The first four features were used as raw data directly from Leapmotion, while others were pre-processed to obtain the distance between fingers, the distance between finger and palm, and the angle between fingers.

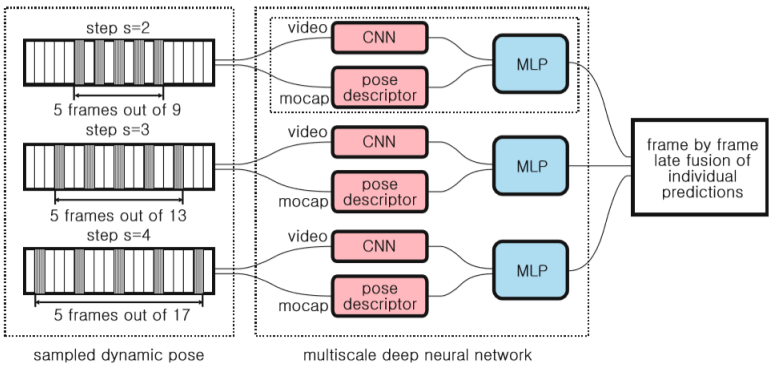
The neural network used by the authors is a LSTM architecture followed by KNN. According to the authors, KNN is an efficient clustering method for handling multi-class classification. Furthermore, during training, KNN compresses the 200 frames to obtain the average point of the features. Lastly, a fully connected layer with a softmax activation function. The authors achieved 91.6%.

#### *3.2.1.2 Non Sign Language Context*

Using Kinect sensor camera, (PIGOU et al., 2014) proposed a recognition system to classify 20 gestures in the Italian language.

The pre-processing is done using data information from the Kinect. First, the authors crop the region of the highest hand by using the joint information. The authors ended up with four videos: grayscale, depth of the body, and hand regions.

Figure 22 – Multi-model depp learning arquitecture from (NEVEROVA et al., 2014).




Source: NEVEROVA et al. (2014)

Two CNNs were used, one for hand feature extraction and the other for body features. Both were comprised of 3 convolutional layers with max-pooling and relu activation function (LECUN et al., 1998b) and a fully connected layer to perform classification.

The dataset used was from ChaLearn Looking (ESCALERA et al., 2014). The authors augmented the dataset by applying a 10% zoom, rotations up to 3%, spatial translations of 5 pixels in the $x$ and $y$ coordinates, and temporal translations up to 4 frames in the images.

The authors achieved a 91.7% classification result on the augmented dataset and a 90.7% on the original dataset.

(NEVEROVA et al., 2014) proposed a gesture detection and localization system by using a multi-modal convolutional neural network for the classification of dynamic gestures. The multi-model architecture consists of a combination of single-scale paths connected in parallel. Each path independently learns a representation and performs gesture classification at its temporal scale and articulated pose descriptors. Predictions from all paths are then aggregated through additive late fusion. Figure 22 displays the proposed architecture.

According to the authors, varying the parameter *s* allows the model to leverage multiple temporal scales for prediction, thereby accommodating differences in duration and styles of articulation of different users.

The authors used the publicly available dataset ChaLearn and used information such as depth, grayscale video, and articulated pose as a feature vector to predict. As a result, the authors achieved an 85% classification accuracy.

(CELEBI et al., 2013) used a dynamic programming technique called Dynamic Time Warping (DTW) to recognize six gestures using Kinect Sensor camera.

DTW is a template matching algorithm to find the best match for a test pattern using the

Figure 23 – a) 36 Mapped Orientations Mapping and b) examples of positions for the word Hello using Playstation Move by (KHAMBADKAR; FOLMER, 2014).


Source: KHAMBADKAR; FOLMER (2014)

reference patterns, where each pattern is represented in a time sequence. DTW classifies an input by calculating the distance between each time step. The stored pattern that has minimal distance is the final result. The distance is calculated using Bellman's principle.

The author uses a weighted distance in the cost computation based on how relevant a body joint is to a specific gesture class.

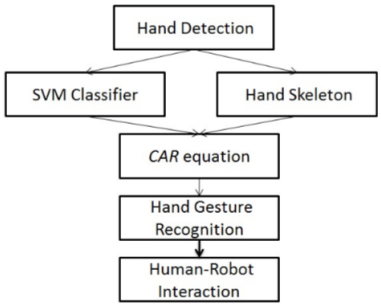
The authors created their dataset using the Kinect sensor camera and consisted of 8 gesture classes with 28 samples per gesture class. Eight samples of each gesture class are performed by trained users, while untrained users perform the remaining 20. As a result, the authors achieve a 96% accuracy.

(KHAMBADKAR; FOLMER, 2014) proposed a static gesture recognition system based on orientation from both hands using Playstation 3 - PS3 video game console[8] and its accessories (camera[9] and motion controller[10]). Being part of the PS3 ecosystem, these accessories are robust and easy to work with. The camera can easily detect the position of the motion controller in a 3D environment, and the console can easily process the camera's input. The authors mapped 36 possible positions to 36 static gestures (26 letters and ten numbers). Figure 23a and Figure 23b show an example of how a word is recognized.

The user has 15 seconds to get to the desired position of the controller and must remain in that position for at least 1 second. When the corresponding letter or number is recognized, a sound is emitted, and the classification result is displayed to the user on the screen. With this technique, the authors achieved a 75% classification rate. It was emphasized by (KHAMBADKAR; FOLMER, 2014) that the proposed system detects a gesture in 3.1 milliseconds. However, although the authors used the whole alphabet and numbers, the gestures used do not correspond to any real gesture in any sign language, and only 36 are mapped. The authors stated that making the positions closer to one another so that more gestures could be mapped would

[8] https://www.playstation.com/en-us/explore/ps3/
[9] http://www.sony. co.in/local/product/playstation+eye
[10] https://www.playstation.com/en-us/explore/accessories/playstation-move/

Figure 24 – Overall view of the proposed system from (LUO; WU; LIN, 2015).




Source: LUO; WU; LIN (2015)

make recognition harder. Furthermore, being based solely on the position of the PS3 motion controller LED, their recognition does not use any artificial intelligence technique.

(LUO; WU; LIN, 2015) proposed gesture recognition using a robot and combining Hand Skeleton Recognition (HSR) method with SVM (HSR + SVM). Their goal was to use the robot to aid a deaf or hard of hearing patient in a hospital to perform certain tasks such as finding his/her way throughout the hospital, calling a doctor, showing his/her current condition, etc. The authors proposed recognizing six gestures comprised of the number of fingers raised (0 - 5). Figure 24 displays the flow of the system.

The first part is hand detection. Using the robot's Charge-CoupledDevice (CCD) camera, each pixel is categorized in skin or non-skin in the Hue Saturation Value (HSV) color space. Next, segmentation using a fixed threshold is performed only in the skin regions. Then, using a Haar-like feature face detection algorithm, face regions are filtered out, leaving only hand regions in the image. From this point, the authors divide their approach in two: SVM and Hand Skeleton.

The Hand Skeleton part is performed using the binarized image containing the hand. First, a distance map (also known as distance field or distance transform) calculates the palm center and determines the hand skeleton. Next, the polygonal approximation is used to find a simplified contour of the hand based on the original contour. It becomes easy to count the number of raised fingers with this information.

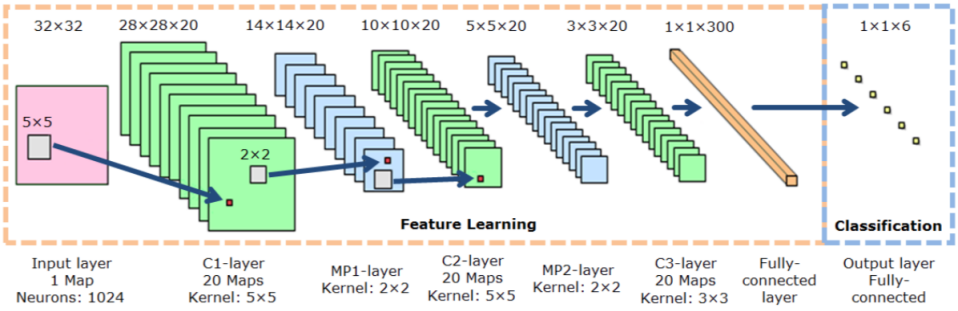
The authors give a few details of their SVM implementation. They state that they used the SVM provided by LIB-SVM version 3.0 from (CHANG; LIN, 2011) and that they used the Gabor feature, RAW, and LBP to extract the features to train and classify the SVM. The idea is to combine these two approaches (Hand Skeleton and SVM) in what they called the CAR equation:

Figure 25 – Results obtained using CAR equation.

| Sign | no | one | two | three | four | five |
|---|---|---|---|---|---|---|
| HSR | 88.2 | 84.4 | 83.2 | 81.4 | 82.0 | 79.5 |
| SVM | 91.8 | 82.4 | 81.2 | 82.0 | 80.2 | 90.5 |
| CAR | 92.2 | 85.7 | 84.3 | 83.7 | 81.9 | 91.0 |

Source: CHANG; LIN (2011)

Figure 26 – Deep learning architecture from (NAGI et al., 2011).




Source: NAGI et al. (2011)

$$CAR \ddot{=} \sum_{s=0}^{5} (N_s^{SVM} * S^{SVM} * N_s^{HSR} * S^{HSR})/(K_{frame} * 2)$$

$$CAR, K_{frame} \epsilon N$$

This equation is based on the K frames being tested.

Although this work achieved good results, as seen in Figure 25, it is not applicable in a daily scenario. Their dataset is limited (only six gestures) and is based on a number of fingers raised, not applied to sign language gestures.

(NAGI et al., 2011) also aimed to provide better human-robot interaction. They proposed a dynamic gesture recognition system using a big and deep neural network combining convolution and max-pooling for feature extraction and classification using a colored glove. The authors recognized six dynamic gestures. Figure 26 shows the architecture used.

The authors used eight multi-layers neural networks comprised of six hidden layers. The hidden layers alternated between convolutional and max-pooling layers. At the end of the network, they used a fully connected MLP. Each convolutional layer has a kernel size, number, and size of maps. In the first convolutional layer, the number of maps was set to 20. A kernel size of 5 x 5 runs throughout the 32 x 32 image, generating a 28 x 28 map. The second convolutional layer also has 20 maps and a kernel size 5 x 5, but instead of running through the original image, it goes through the previous layer, a max-pooling layer. The same idea can be used in the third convolutional layer, the only difference being the kernel size that

was 3 x 3. The number of maps can be better understood as the number of features. The max-pooling layer downsamples the image where the output is the maximum activation over non-overlapping rectangular regions of predetermined size. The size adopted by the authors in all max-pooling layers was 2 x 2. Because there was overlapping, the resulting image from the convolutional layer was half of its size. At last, for the final fully connected layer, MLP was used with a softmax activation function.

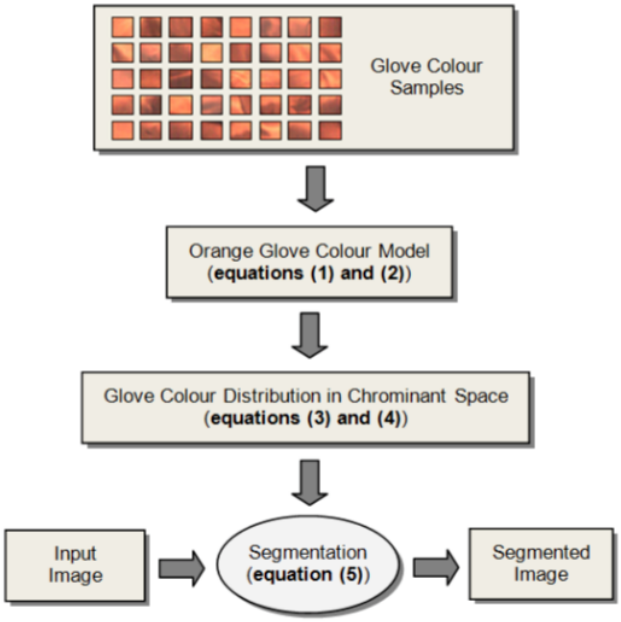
Input images were captured using the robot's camera while the person performing the gesture used a colored glove. The gestures to be recognized are only based on the number of fingers, 0 to 5, resulting in 6 gestures. One thousand images were captured for six different hand distances varying between the gestures. To segment the glove, the authors used the Single Gaussian Model (SGM) (VANĚK; MACHLICA; PSUTKA, 2013) using the mean and the covariance of the chrominance color with a bivariate Gaussian distribution. The glove color distribution is modeled using an elliptical Gaussian joint Probability Density Function (PDF). Mahalanobis distance is used to calculate the distance between the color matrix and the mean vector, which becomes the luminance and chrominance threshold to segment the orange-colored glove from the image. Figure 27 displays the steps taken to perform the segmentation. In the formulas 1, 2, 3, 4, and 5, $c$ is a color vector representing the random measured values of chrominance, $W_s$ is a class describing the glove color, $\mu$ is representing the mean value and $\sum_s$ represents the covariance matrix of the orange color of the glove.

Once the image is segmented, it is fed to the proposed deep learning architecture. The results achieved a 96% classification rate.

(PARK; LEE, 2011) followed the same purpose of creating a more natural way to interact with robots (other than using keyboard and mouse). They proposed a pointing gesture recognition system for mobile robots using a cascade Hidden Markov Model (HMM) (BAUM; PETRIE, 1966) and a particle filter. The robot is equipped with two cameras (depth and RGB) to detect the object to which a person is pointing.

The depth camera is helpful to detect a human body from the background, although the authors still applied some rules like size, the ratio between width and height, and position. Based on the human skin region, Viola and Jone's face detection algorithm was used to find the location of the face in order to build (JONES; REHG, 2002) histogram color model, thus the skin model. This was used to find the hand in combination with Mahalanobis distance quickly. Once the detection is finished, tracking is done using a 3D particle filter with information from the depth camera. The system recognizes three phases to reach a pointing gesture: non-

Figure 27 – Deep Neural Network steps and formulas used from (NAGI et al., 2011).




$$p[c/W_s] = (2\pi)^{-1}|\sum_s|^{-\frac{1}{2}} exp^{(c-\mu_s)^T \sum_s^{-1}(c-\mu_s)} \quad (1)$$

$$c = [x(i,j)y(i,j)]^T \quad (2)$$

$$\mu_s = \frac{1}{n}\sum_{j=1}^{n} c_j \quad (3)$$

$$\sum_s = \frac{1}{n-1} \cdot \sum_{j=1}^{n}(c_j - \mu_s)(c_j - \mu_s)^T \quad (4)$$

$$\lambda(c) = (c_j - \mu_s)^T \sum_s^{-1}(c_j - \mu_s) \quad (5)$$

Source: NAGI et al. (2011)

gesture, move to and point to. Each frame is analyzed according to an imaginary hemisphere previously created (Figure 28).

To train the HMM to identify the object a user is pointing to, users were asked to point to markers positioned on objects in a room such as television, picture frame, radio, etc. A total of 2004 pointing gestures was captured, 1656 used for training and 348 for testing. The results obtained varied from 89% to 99% accuracy.

Users have to stand precisely 3 meters away from the robot, and the background cannot have any similar color to the skin of the person performing a gesture. The markers have fixed places, and moving them around would require constant re-training. Complex background

Figure 28 – Virtual Hemisphere created from (PARK; LEE, 2011).

Source: PARK; LEE (2011)

Figure 29 – Glove from (PAULSON; CUMMINGS; HAMMOND, 2011).

Source: PAULSON; CUMMINGS; HAMMOND (2011)

analysis is only possible due to the depth camera used by the robot.

(PAULSON; CUMMINGS; HAMMOND, 2011) also proposed a system to detect objects with which a user is interacting, based on his/her current gesture, but based on a glove device (Figure 29). This glove has 22 sensors and was developed by Immersion Corporation[11]. This work is proposed to be used in an office environment only. Objects detected based on hand movement were: cup, telephone, earphones, keyboard, mouse, etc.

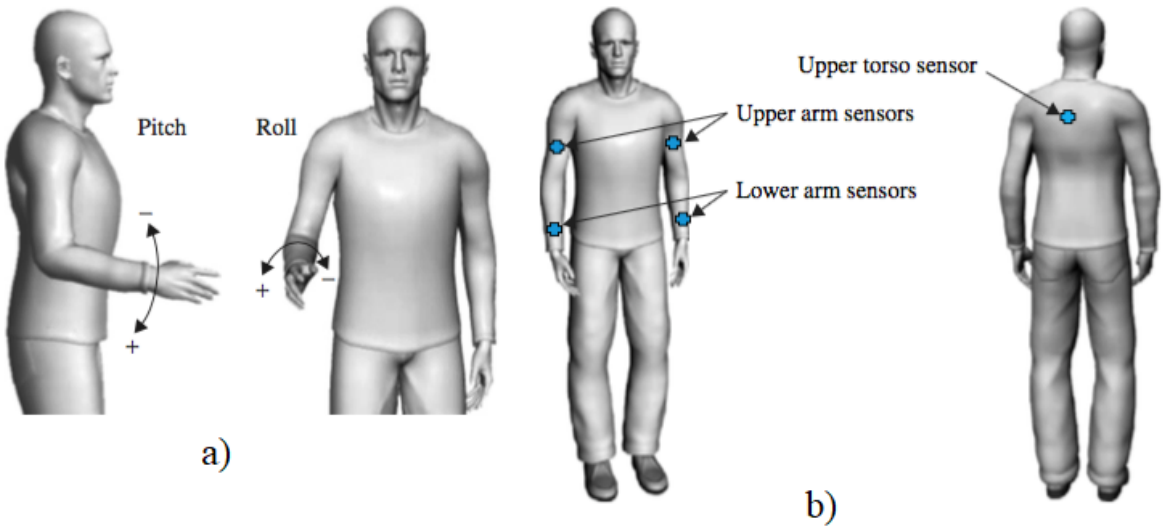
The authors used a linear classifier and KNN to perform the classification of gestures. The feature vector was composed of all 22 sensors. In addition, they used PCA and Linear Discriminant Analysis (LDA) to reduce the dimensionality, although the 22-feature vector was used even after applying PCA.

Combining classifiers and feature space allowed the results to vary from 51,7% to 96,9% classification rate. Using a linear classifier with raw input achieved the lowest accuracy of 51,7%. A 91,7% accuracy was achieved using classifier/feature space combination, although it was not informed which features were used. Using a linear classifier with LDA, a 66,7% minimum and 76,7% maximum was achieved. KNN with PCA achieved the best accuracy of 96,9%, with 88,2% minimum.

(RAHEJA; SUBRAMANIYAM; CHAUDHARY, 2016) proposed a gesture recognition system using PCA implemented in a Field Programmable Gate Array (FPGA)[12] simulator and Matlab.

[11] http://www.cyberglovesystems.com
[12] https://www.xilinx.com/training/fpga/fpga-field-programmable-gate-array.htm

Figure 30 – a) Pitch and roll features extracted and b) Sensors' location from (JUNKER et al., 2008).




Source: JUNKER et al. (2008)

The FPGA was configured to calculate the Squared Euclidian Distance (SED) to overcome the delay when computing squared root and Minimum Distance - MD. Six different SED were calculated and stored in different memory blocks. SED is calculated between the test input and each training image. Once all SED 's were calculated, Minimum Distance (MD) is calculated based on the SED vector. The memory block with the smallest distance is the result.

The authors did not present any results. For example, they did not implement PCA, only mentioning that it can be implemented and did not inform which gestures they recognized.

(JUNKER et al., 2008) proposed a two-step gesture recognition system through a continuous data stream coming from body-worn inertial sensors. The first stage is a pre-selection of signals, and the second is the classification of these signals using HMM.

The pre-selection of signals within a data stream is a difficult task. In order to retrieve the signals that are part of a gesture, the authors used some techniques to perform the extraction: the pitch and roll of the lower arm sensors (Figure 30a); the Sliding-window and Bottom-up Algorithm (SWAB); and a coarse search. However, it is unclear how the authors used these techniques as they did not give a detailed view of their application. HMM was chosen by the authors due to its ability to work with temporal and spatial variations of input patterns.

Five sensors were used on the body as shown in Figure 30b, and four people performed eight gestures: handshake; press light button; open door; answer the phone; insert coin; cut object; drink water; drink soup; and eat. The results obtained varied between 97,4% and 98,4% classification rates.

### 3.2.2 Papers not based on Special Hardware

#### *3.2.2.1 Sign Language Context*

(PAN et al., 2016) proposed a sign language recognition system for ASL in complex background by combining PCA, LDA and SVM to construct a hierarchical classification scheme.

Hand segmentation is based on skin color detection. Instead of using a fixed Cb and Cr channel threshold in the YCbCr color space, the authors proposed an adaptive skin color segmentation based on motion clues. This is done by asking the user to wave the hand in front of the camera before using the system. While the hand is moving, the system creates a moving block where, if 75% of the pixels within this block is in a current skin color range predefined, the color range is updated with the new values, and the connected component technique is used to find the largest skin blob. By applying Gaussian Mixture Model (GMM) to the colors of all the pixels within the skin blob, the new skin range is found and used to detect skin pixels quickly.

Feature extraction is based on 30 Scale-Invariant Feature Transform (SIFT) (LOWE, 2004) points, 7 Hu Moments features (HUANG; LENG, 2010) and Fourier Descriptors. To reduce the dimensionality of the feature vector, the authors applied PCA and hierarchically clustered all the 26 alphabet letters based on their reduced features. For each group of features, LDA was applied to train the discriminative feature set, which reduced the feature dimension to 25. This feature set was used to train the corresponding SVM classifiers based on linear kernel.

The authors used two datasets to perform their tests: a dataset of images collected by themselves based on the Chinese Sign Language (CSL), and a publicly available ASL dataset made by (BARCZAK et al., 2011). The authors achieved a 99.8% classification rate in the first dataset and 94% in the second dataset.

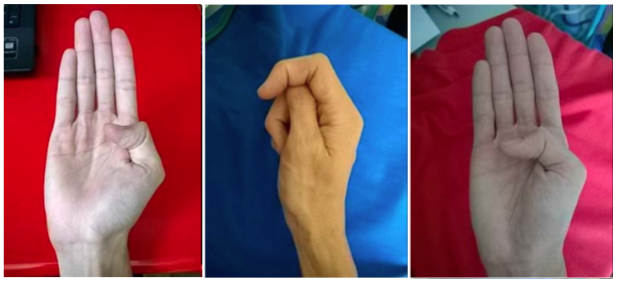
(JIN; OMAR; JAWARD, 2016) proposed a mobile gesture recognition system to translate 16 ASL static signs by using Canny filter (CANNY, 1986) and Speeded Up Robust Features (SURF) to extract features, and SVM to classify.

Initially, the authors captured the minimum and maximum RGB values by asking the user to calibrate the system. This calibration was done simply by taking a hand picture and analyzing the RGB values within a pre-determined region. The Canny filter was applied next to detect edges. The authors used a seeded region growing method to segregate the hand from the background using four seed points at the image's top left and right, bottom left, and center.

Figure 31 – a) Segmentation result and b) SURF feature extraction by (JIN; OMAR; JAWARD, 2016).


Source: JIN; OMAR; JAWARD (2016)

Figure 32 – Moderate complex backgrounds from (JIN; OMAR; JAWARD, 2016).



Source: JIN; OMAR; JAWARD (2016)

The edges detected by the Canny filter were used as a boundary to limit the growth of the region method. Figure 31a displays the segmentation result and Figure 31b displays the SURF features from the segmented image.

Because each SURF descriptor has its dimension, the Bag of Feature (BoF) method was used to represent the features in a histogram of visual vocabulary because SVM requires a uniform dimension of features as input. K-Means was used to minimize the sum of squared Euclidean distances between data points and their nearest clusters (16 defined, each representing one gesture). The sum of squares was computed for each data point, and the data points were assigned to the cluster which conveyed the nearest value. This was followed by the update step, which involved calculating the new mean to be the centroid of each cluster. Each cluster center was assigned to the new centroids. The convergence was defined by reaching a maximum of 20 iterations or a squared error of less than 0.001. The clustered features correspond to one gesture.

The authors used 100 images for each of the 16 gestures to train SVM and used RBF as the kernel. The final result achieved a 97.13% classification rate in moderate complex backgrounds as displayed in Figure 32.

(PATTANAWORAPAN; CHAMNONGTHAI; GUO, 2016) proposed ASL gesture recognition using Discrete Wavelet Transform (DWT) (AKANSU; HADDAD, 2001) and Area Level Run Lengths features and SVM to classify. The authors divided the gestures into fist signs (no finger extension out of the palm) and non-fist signs (at least one finger extended). In the first group, recognition was performed using the wavelet approach, while area-level run lengths were used for the latter group.

To determine whether a sign is part of the fist or non-fist group, the authors binarized the input image using Otsu filter (OTSU, 1979) to calculate the Shape Factor (SF) and the axis ratio. Fist signs tend to be more circle-like. Images were expected to have a simple background with uniform color. Shape vector was calculated using the following equation:

$$S = \frac{4piA}{p^2}$$

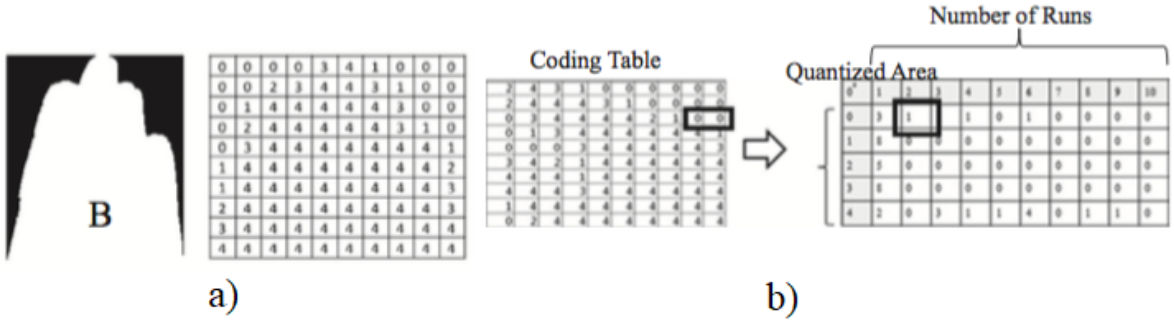
where $A$ is the hand area, calculated by the total of hand pixels, and $p$ is the number of counter pixels. If SF is more significant than a threshold value, SF is set to 1; otherwise, it is set to 0. The axis ratio is calculated by dividing the major axis by the minor axis. If the axis ratio is less than a fixed ratio, its final value becomes 1; otherwise, it becomes 0. The threshold value was found by performing 2300 sign samples that consisted of 700 fist and 1600 non-fist signs. The threshold values for SF and axis ratio were 0.6 and 1.6, respectively.

Feature extraction was done differently for each group. For the first group, feature extraction began by finding the centroid of the hand. Once the centroid was found, a Region of Interest - ROI was extracted using the finger region and used to classify. The finger's contour was transformed into a 1D image because DWT can isolate small details in this dimension. Statistical descriptors like mean, variance, skewness, and kurtosis were calculated to be used in the recognition phase. According to the authors, using this approach, approximate coefficients provide significant information to differentiate similar fist signs (e.g., A and S in ASL).

For the non-fist signs, a 10 x 10 block mask was used to distinguish fingers' states and names. This block can have five values:

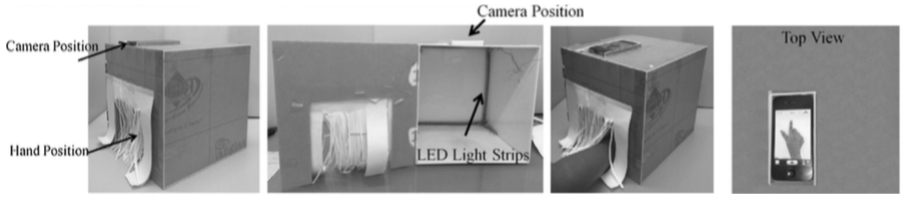
- 0 if no hand pixel is on the block.
- 1 if hand area is less than 25.01% of the block area.
- 2 if hand area is between 25.01% and 50%.

Figure 33 – a) Gesture B binarized representation and b) the run length matrix from (PATTANAWORAPAN; CHAMNONGTHAI; GUO, 2016).




Source: PATTANAWORAPAN; CHAMNONGTHAI; GUO (2016)

Figure 34 – Environmental setup from (PATTANAWORAPAN; CHAMNONGTHAI; GUO, 2016).




Source: PATTANAWORAPAN; CHAMNONGTHAI; GUO (2016)

- 3 if hand area is between 50.01% and 75%.
- 4 is hand area is greater than 75%.

Figure 33a displays the result from the process for the gesture B.

After that, a 5 x 10 block was used to create the run-length matrix. The first column of this matrix is the quantized values (0 - 4), and all the other elements represent the number of times a sequence of quantized values occurred. Figure 33b illustrates this process.

Statistical analysis was used to extract 28 features from this matrix. According to (XU et al., 2004), statistical analysis based on a run-length matrix is a texture metric that captures the coarseness of a texture in a specific direction (0°, 45°, 90°, 135°). The authors did not inform the direction nor statistical analysis they performed, but according to (GALLOWAY, 1975), such analysis can calculate features such as short or long runs emphasis, gray level uniformity, run-length non-uniformity, and run percentage. The authors used an MLP neural network trained with backpropagation and a sigmoid activation function to recognize the gestures. The proposed system achieved an 89.38% classification rate, considering that the background must be simple and homogeneous with constant lighting. Figure 34 displays the environment setup.

(AL-ROUSAN; ASSALEH; TALA'A, 2009) proposed an Arabic sign language recognition using

Figure 35 – Skin segmentation result from (HARTANTO; KARTIKASARI, 2016).

Source: HARTANTO; KARTIKASARI (2016)

HMM. A total of 18 signers performed 30 dynamic gestures, which resulted in 7860 images. To remove the background, image subtraction was performed for each sign of *n* frames. Feature extraction was done using DCT on the image with the background removed.

For training and classification, 30 HMM models were created to represent all gestures. The authors tested their system using four modes: signer-dependent online/offline and signer-independent online/offline. In the online mode, the signer performs in real-time, while a video is recorded to be analyzed in the offline mode. Signer dependent mode means that the same person will sign in the training and testing phases, while signer independent means that different persons perform training and testing. A 93,8% classification rate was achieved for the signer-dependent online results, while the offline approach achieved 97,4%. For signer-independent, the results were: 90,6% for online and 94,2% for offline. The background used by (AL-ROUSAN; ASSALEH; TALA'A, 2009) was simple and homogeneous.

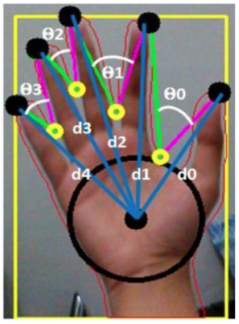
(HARTANTO; KARTIKASARI, 2016) proposed a static gesture recognition system of Indonesian Sign Language in mobile devices. Feature extraction was performed using skin segmentation in the YCbCr color space to extract the hand's convex hull and convexity defects. A backpropagation neural network was used to classify 24 static gestures.

As seen in previous papers, sample hand images were used for skin color reference. The color space is then converted to YCbCr color space to reduce illumination, and skin segmentation is performed in the Cb and Cr color channels (Figure 35).

Once the segmented image is obtained, convex hull and convexity defects are calculated to find the hand and finger position area. Features such as finger distance and angle between fingers were used to classify the static gestures. Figure 36 displays the representation of the distance between the center of the hand and the tip of the finger and the angle between fingers.

The distance is calculated using the following equation:

Figure 36 – Distance and angles features from (HARTANTO; KARTIKASARI, 2016).




Source: HARTANTO; KARTIKASARI (2016)

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \tag{3.4}$$

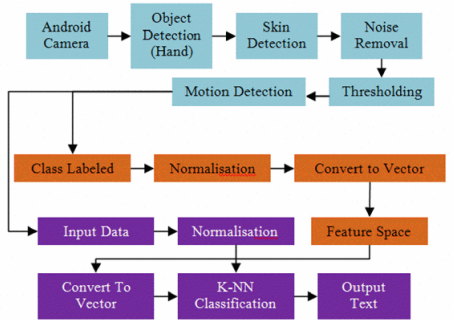
where x and y are coordinates of the tip of the finger and the center of the palm, the authors did not estimate the angle calculation.

The backpropagation neural network architecture had nine input neurons (five distances and four angles), 14 hidden neurons, and five output neurons. The authors represented the alphabet as binary numbers, thus reducing the number of output neurons from 24 to 5. To increase performance, all input data to the network were normalized, either using max value normalization or Z-score normalization. The dataset consisted of five images per gesture using Z-score normalization, and the authors achieved a 91.66% accuracy. The training was performed in a desktop environment, and once the output weights were found, the classification was performed on the mobile device.

(PRASUHN et al., 2014) combined a mobile and server approach to performing gesture recognition in ASL. First, the authors extracted the hand area of the input image. Their proposed system applies simple binarization in HSV color space. Then, morphological operations are applied to the binarized image to remove extraction noise. Once this area is extracted, the authors use Histogram of Oriented Gradients (HOG) to extract relevant information of the gesture.

The authors used HOG due to its advantage of being robust with illumination change and shadow. However, the downside is that HOG features are not robust regarding object orientation. Therefore, computer-generated images were used to populate the dataset to compensate for this disadvantage. The authors used two datasets: original images (front side view of the gesture); and computer-generated images. This generation was performed by rotating the images' azimuth and elevation angles either by +30 degrees or -30 degrees. The dataset

Figure 37 – Proposed system architecture from (HAKKUN; BAHARUDDIN et al., 2015).




Source: HAKKUN; BAHARUDDIN et al. (2015)

images were taken from 4 non-native sign language speakers, each performing 19 signs.

Feature extraction starts by first computing the gradient vector of every pixel in the image. Next, a 9 x 9 cell is defined without any overlap and computes each cell's weighted histogram of gradient orientations. The authors state that bins (class intervals) were evenly spread over 0 to 180 degrees; specifically, nine bins with 20 degrees intervals are used. Then, a set of blocks, each consisting of 3 x 3 cells, is defined with overlap. A feature vector of 9 x 3 x 3 = 81 dimensions is obtained for each block, thus composing the HOG descriptors.

Every image in the dataset has its HOG feature vector, which is compared to brute force with the feature vector from the input image, although the authors stated that any nearest-neighbor could have been used. They decided to use this approach due to the dataset's low number of images. The $L^2$ distance was used between the two HOG feature vectors, and the image with the minimum distance is selected as the final result.

With the front side view images dataset, a 47% accuracy was obtained, and when using the slanted images dataset, the accuracy decreased to 39%.

(HAKKUN; BAHARUDDIN et al., 2015) investigated the ideal distance and angle to detect a hand from a mobile device in the context of the Indonesian Sign Language (although the authors did not specify which gestures they used for classification). The authors used Viola and Jones Robust real-time object detection and KNN to detect the hand and perform classification, respectively. Figure 37 displays the system architecture.

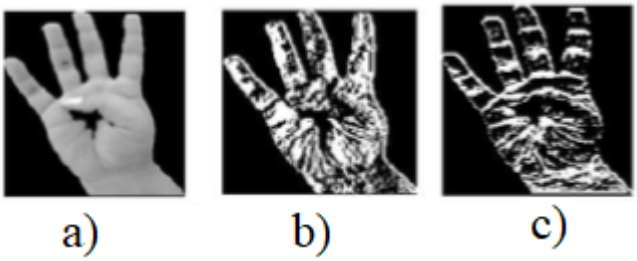
The authors' experiment took into account the following distances: 30 cm, 50 cm, 70 cm, 90 cm and 110 cm, and the following angles: 0°, 30°, 45°, 90°. The hand was best detected at less than 50 cm from the mobile phone and should be in an upright position (0°). Six to 20 frames were necessary for hand detection and classification.

The authors did not provide any details regarding their environment setup, implementation,

Figure 38 – Segmentation result from (THALANGE; DIXIT, 2016).

Source: THALANGE; DIXIT (2016)

Figure 39 – a) Cropped gray image, b) and c) are the dx and dy results from (THALANGE; DIXIT, 2016).




Source: THALANGE; DIXIT (2016)

database, and, more importantly, the classification rate for every distance and angle used.

(THALANGE; DIXIT, 2016) proposed the recognition of sign language gestures based on the following features extraction techniques: 1) Orientation Histogram, 2) Statistical Measures, 3) Combined Orientation Histogram and Statistical Measures (COHST) and 4) Wavelet. The gestures recognized by the authors were numbers from 0 to 9 in ASL.

The palm was extracted and cropped from the image by binarizing the image. A median filter and morphological operations were used to extract the noise from the result of the binarized image to be resized to 128 x 128. Figure 38 displays the initial results.

Features of Orientation Histogram and Statistical parameters were extracted from cropped and resized gray images, while Wavelet features were extracted from a binary image.

For the Orientation Histogram feature, the gradients in the *x* direction were calculated by applying the filter [ 0 -1 1 ], and in the *y* direction $[0\ 1\ -1]^T$, thus resulting in two gradient images: *dx* and *dy*. Gradient direction was obtained by calculating $tan^{-}1$ of *dy* / *dx*. By converting this matrix with radian values to degrees, the number of degrees in different histograms was used as a feature vector for Orientation Histograms. This process obtained 18 features. Figure 39 displays the cropped gray image and the gradient from *dx* and *dy*.

Statistical parameters were taken from the grayscale image of the hand after converting to a 1D vector. Six parameters were calculated: mean, standard deviation, variance, coefficient of variance, range, and root mean square of the successive difference. As mentioned before, COHST is a combination of the Orientation Histogram and Statistical parameters. The binary

Figure 40 – a) 1st decomposition result for LH, HL and HH bands and (b) Approximation coefficients matrix image from (THALANGE; DIXIT, 2016).


Source: THALANGE; DIXIT (2016)

image is used for feature extraction for the Wavelet feature vector. Using the Haar Wavelet algorithm, four regions were obtained from the transformed space: LL, LH, HL, and HH. The LL band was downsampled by 2 of the original image. LH contains the horizontal features, while HL contains the vertical features. HH regions isolate localized high-frequency point features. This first part is characterized as level one of decomposition. The authors applied the 3 Haar Wavelet algorithm to obtain the 4th level of decomposition, and the approximation coefficients matrix of the level decomposition (LL band) was used as a feature vector for training. Figure 40a displays the 1st decomposition for LH, HL and HH bands. Figure 40b displays the approximation coefficients matrix image LL at first, second, third, and fourth level decomposition, respectively.

The authors used a multi-layered Feedforward Backpropagation neural network to train and classify. Four different systems were created for the four different features. Thirty images corresponding to gestures 0 to 9, using simple homogeneous background, were captured from a single signer and varied in scale and hand rotation between +45 and -45 degrees in all four directions to make the system robust.

The authors achieved a 74,69% classification rate using only Statistical Measures, 82,92% using Orientation Histograms, 87,94% combining the previous two, and 98,17% using Wavelet features. Although recognizing only numbers in the simple homogeneous background, the authors stated that it could also be used in dynamic gestures.

(COSTER; HERREWEGHE; DAMBRE, 2020a) used a Transformer network (VASWANI et al., 2017b) to classify static Flemish Sign Language. Feature extraction was performed using OpenPose. A Transformer network is an encoder/decoder model that tackles a couple of problems that RNN/LSTM has: lack parallelization (RNN/LSTM and Gated Recurrent Unit (GRU) process word by word sequentially resulting in long training time) and the relationship between words far apart. This is done by using multi-headed attention. Inputs first go through an embedding layer to be represented as a vector. Each embedded representation is added to

a Positional Encoding (PE) given by the following equation:

$$PE_{(pos,2i)} = Sin(pos/1000^{2i/d_{model}})$$
$$PE_{(pos,2i+1)} = Cos(pos/1000^{2i/d_{model}})$$

From this new input, a query, key and value are calculated through trainable linear transformations:

$$Q = XW^Q,$$
$$K = XW^K,$$
$$V = XW^V$$

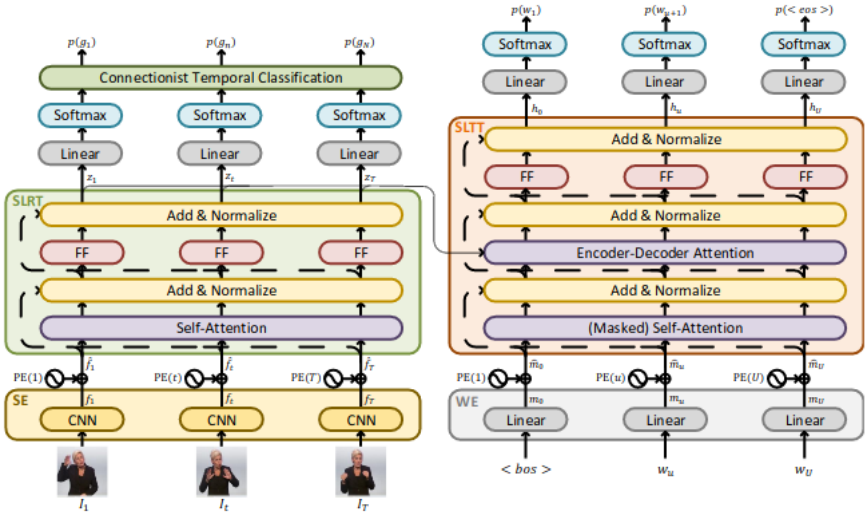
where $W^Q$, $W^K$, and $W^V$ are trainable weights. The scaled dot product is applied to Q and K to calculate the attention weights, which are then multiplied by V.

The dataset used by the authors consisted of 100 classes in 18730 samples performed by 67 different people. As a result, the authors achieved a 74.7% accuracy.

Using Transformers, (CAMGOZ et al., 2020a) recognized more than 8,000 sign language sentences from the PHOENIX-14T dataset within a German weather forecast context. Images were used as-is and were trained using an Efficientnet CNN to obtain 1024 features. Sentences were represented as glosses to facilitate recognition. The authors added a ConnectionistTemporal Classification (CTC) layer to calculate the encoder gloss loss and summed it with the decoder loss (Figure 41). The authors used three-layered, 512 encoder/decoder features with a 2048 dense layer. The authors achieved a 24.5% WER using a sign2gloss approach.

(YIN; READ, 2020) used a Spatial-Temporal Multi-Cue (STMC) network from (ZHOU et al., 2020) combined with a Transformer architecture. According to the authors, the STMC network comprises a Spatial Multi-cue (SMC), which decomposes the input video into spatial features of multiple visual cues (face, hand, full-frame, and pose). Then a Temporal Multicue (TMC) calculates temporal correlations within (inter-cue) and between cues (intra-cue) at different time steps. The inter-cue and intra-cue features are fed to a bi-directional Long Short-Term Memory (BiLSTM) and CTC units for sequence learning and inference. Two-layered transformers were used with the Phoenix-14T dataset. Using this approach, a 21% WER was achieved.

Figure 41 – Transformers architecture with the added CTC loss at the end of the encoder. Adapted from (CAMGOZ et al., 2020a).




Source: CAMGOZ et al. (2020a)

Using Korean Sign Language, (KO et al., 2019a) proposed a sign language translation using human key points. The dataset used was KETI, a Korean Sign Language for emergencies. The authors used 105 sentences and 41 words to train their network. Their architecture is a two bi-directional Gated Recurrent Units - GRU.

The authors use Openpose library ((Cao et al., 2019)) to extract 124 human keypoints and extract 2 features vectors:

$$V_x = (v_1^x, v_2^x, ..., v_n^x)$$
$$V_y = (v_1^y, v_2^y, ..., v_n^y)$$

and normalizes each vector as follows:

$$V_x^* = \frac{V_x - \overline{V_x}}{\sigma(V_x)}$$

where $V_x$ is the mean and $\sigma(V_x)$ is the standard deviation of $V_x$. The normalized vectors are then concatenated ($V^* = [V_x^*; V_y^*]$) and fed to the network. The authors extracted a fixed number of frames to be used from a sentence video. This frame extraction was based on the following skip frame equation:

$$Z = [\frac{l}{n-1}]$$

where $n$ is the fixed frames to extract, $l$ is the number of frames in a video. The authors mentioned in their work that the sentences were represented as glosses, meaning that instead of using the entire sentence as a label, they manually created unique glosses to represent each sentence. They stated that the translation is more appropriate. Figure 42 shows 10 examples of the resulting glosses.

Figure 42 – Glosses created by (KO et al., 2019a).

| ID | Korean Sentence | English sentence | Sign gloss |
|---|---|---|---|
| 1 | 화상을 입었어요. | I got burned. | FIRE SCAR |
| 2 | 폭탄이 터졌어요. | The bomb went off. | BOMB |
| 3 | 친구가 숨을 쉬지 않아요. | My friend is not breathing. | FRIEND BREATHE CANT |
| 4 | 집이 흔들려요. | The house is shaking. | HOUSE SHAKE |
| 5 | 집에 불이 났어요. | The house is on fire. | HOUSE FIRE |
| 6 | 가스가 새고 있어요. | Gas is leaking. | GAS BROKEN FLOW |
| 7 | 112에 신고해주세요. | Please call 112. | 112 REPORT PLEASE |
| 8 | 도와주세요. | Help me. | HELP PLEASE |
| 9 | 너무 아파요. | It hurts so much. | SICK |
| 10 | 무릎 인대를 다친 것 같아요. | I hurt my knee ligament. | KNEE LIGAMENT SCAR |

Source: KO et al. (2019a)

The authors achieve a 93.6% accuracy on a validation set and a 55.28% on a test set.

(KO; SON; JUNG, 2018a) proposed a very similar approach for the Korean Sign Language dataset as (KO et al., 2019a). The authors used the same dataset, architecture, and feature extraction. The difference was that they added a confidence value (already provided by Openpose) to the network input, and they used 1,000 videos corresponding to 100 sentences. The authors informed that they used two bi-directional GRU stacked with 400 cells in the first GRU (encoder) and 200 cells in the second (decoder). Adam optimizer was used with a batch size of 64 and a learning rate of 0.00001 over 80 epochs. They did not inform if they used a sentence-based or glossed-based approach. The authors achieved an 89.5% accuracy.

(BARROS et al., 2017) proposed a dynamic gesture recognition and prediction using a convexity approach as feature extraction. The authors used two datasets: RPPDI Dynamic Gestures Dataset and Cambridge Hand Data and used two algorithms: HMM and DTW. The convexity features can be understood as minimal points depicting a hand, performed in 3 steps.

The first step uses the Douglas-Peucker algorithm to find a set of minimal points. This algorithm iterates through all the points connecting two endpoints ($\overline{AB}$) and calculating their distance ($v_i$) and center. If the calculated values are below a pre-determined threshold, the

Figure 43 – Convextiy approach result on a LCS and SURF images used by (BARROS et al., 2017).

Source: BARROS et al. (2017)

points are kept; else, $\overline{AB}$ is subdivided. The second step is performed by selecting inner and outer points. Outer points are selected by using a convex hull around. Whatever vertex touches the hull will be selected as an external point. Internal points are selected by choosing the point located between 2 external points with the greatest distance. Figure 43 displays the results of the convexity approach applied to a Local Contour Sequence (LCS) and SURF images. The third and final step is normalization the distances between the external points.

As stated before, the authors implemented two types of recognition: full sequence recognition and incomplete sequence recognition (prediction). For both recognitions, HMM and DTW used similar architectures. A set of specialized HMMs or DTWs is responsible for predicting its probability based on approximation calculations to the input. While the HMMs used K-Means clustering to find the best initial approximation, Baum-Welch algorithm to perform faster training and choose the gesture based on the highest probability, DTW used the Euclidean distance and chose the gesture based on the smallest distance. Figure 44(a) and 44(b) displays the HMM and DTW architecture respectively.

For the RPPDI Dynamic Gestures Dataset, image preprocessing was performed using the Convexity approach for LCS (CLCS) and SURF (CSURF) images, both explained earlier. As for the Cambridge Hand Data dataset, preprocessing went through Difference of Gaussian (DoG) filter followed by Laplace-Beltrami operator before applying the CLCS and CSURF algorithm. Figure

For classification using the entire sequence on the Cambridge Hand Data, the author achieved a 93.98% accuracy using CSURF + DTW, and on the RPPDI Dynamic Gestures Dataset, a 77.44% accuracy using CSURF + HMM, 81.66% using CLCS + HMM and 52.55% using LCS + HMM.

As for the prediction, recognition of using a part of a sequence to recognize a gesture, the authors divided the results using 15%, 25%, 50%, 75%, 100% of the sequence. For Cambridge dataset using CLCS + HMM, Table 7 shows the results:

Figure 44 – HMM and DTW architectures used by (BARROS et al., 2017).

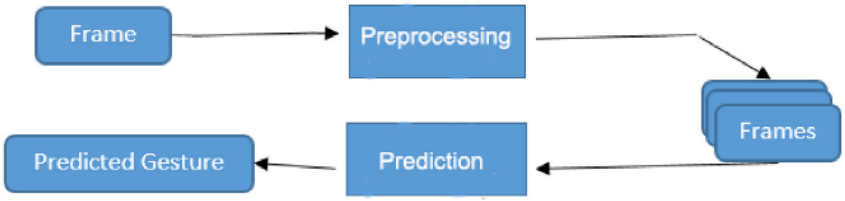
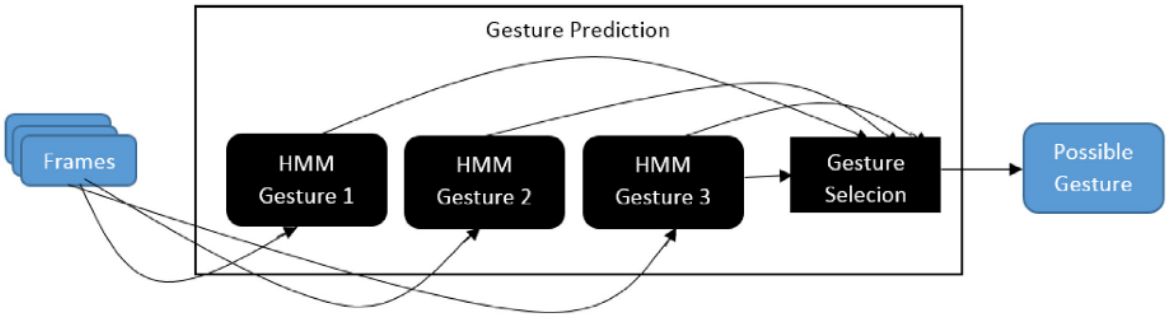
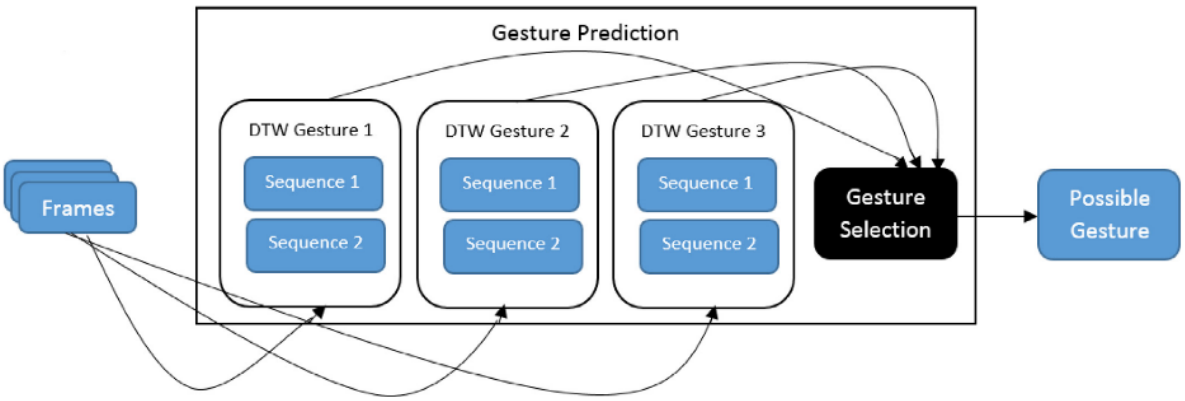



**Fig. 6.** Gesture prediction system general architecture.




(a) HMM architecture




(b) DTW achitecture

Source: BARROS et al. (2017)

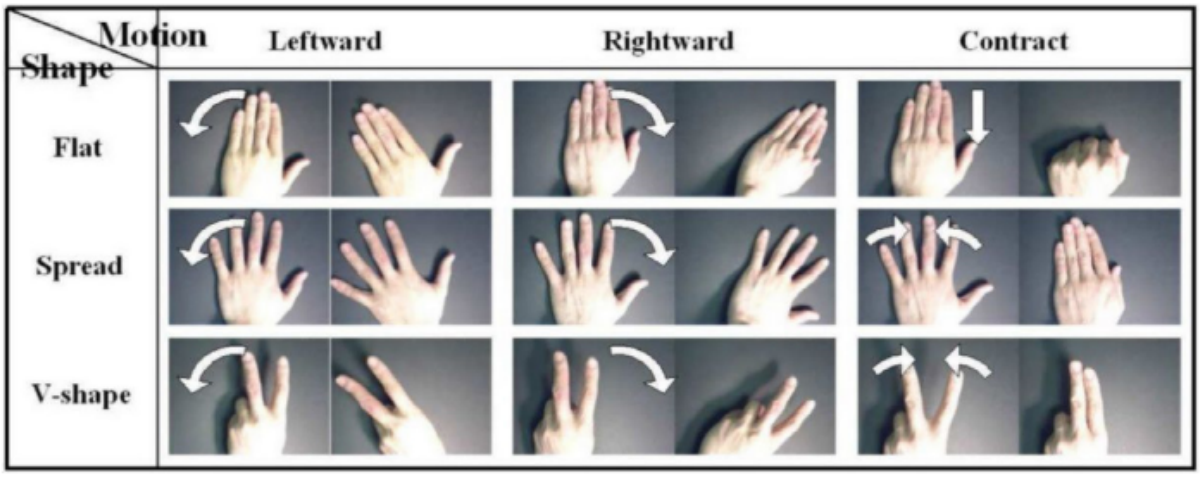
Table 7 – Results for prediction on Cambridge dataset using CLCS + HMM from (BARROS et al., 2017).

| % of Frames | 15 | 25 | 50 | 75 | 100 |
|---|---|---|---|---|---|
| Accuracy | 10.5 | 37.8 | 50.2 | 67.1 | 93.98 |

Source: BARROS et al. (2017)

Figure 45 – Examples images of the Datasets used by (BARROS et al., 2017).


(a) RPPDI Dynamic Gestures Dataset




(b) Cambridge Hand Data examples

Source: BARROS et al. (2017)

Table 8 – Results for prediction on the RPPDI dataset from (BARROS et al., 2017).

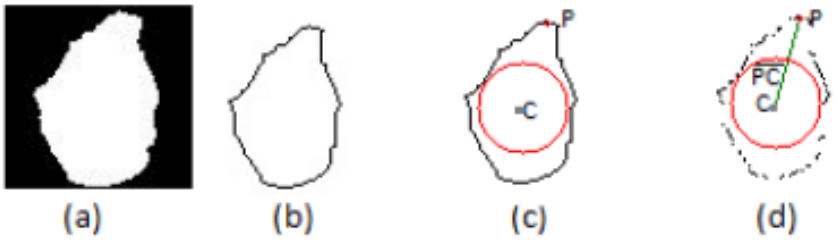
| Prediction results using CLCS with 1, 5, 10 and 14 frames. | | | | |
|---|---|---|---|---|
| Technique | 1 Frame | 5 Frames | 10 Frames | 14 Frames |
| HMM | 37.90 | 66 | 80 | 81.66 |
| DTW | 17.18 | 17.29 | 64.06 | 97 |
| Prediction results using CSURF with 1, 5, 10 and 14 frames. | | | | |
| HMM | 37.57 | 62.91 | 75.41 | 77.44 |
| DTW | 16.53 | 18.43 | 85.31 | 94.08 |
| Prediction results using LCS with 1, 5, 10 and 14 frames. | | | | |
| HMM | 36.14 | 44.27 | 52.65 | 52.55 |
| DTW | 12.50 | 21.87 | 78.43 | 89.06 |

Source: BARROS et al. (2017)

(SANTOS; FERNANDES; BEZERRA, 2015) proposed a dynamic gestures recognition using depth maps generated by Microsoft Kinect Camera. The authors proposed a hybrid classifier using HMM and DTW and achieved 97.49% in the MSRGesture3D dataset and 98.43% in the RPPDI dynamic gesture dataset.

Kinect depth image is used for feature selection (Figure 46a). The authors calculate the center of mass and the highest counterpoint (Figure 46c) and find the distance between these 2 points and find the biggest circumference within a contour (Figure 46d). Signatures are a set of distances and angles between the center of mass and convex hull points. The final feature selection step uses binary particle swarm optimization to find the optimal feature set size.

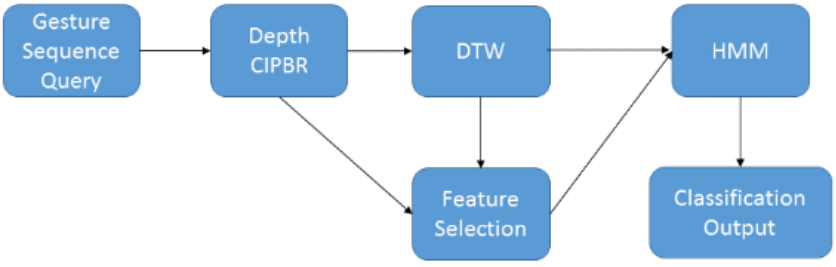
Figure 46 – Feature extraction step from (SANTOS; FERNANDES; BEZERRA, 2015).




Source: SANTOS; FERNANDES; BEZERRA (2015)

For the hybrid proposal, the authors state that DTW works well in classifying grouped patterns, but it is not very sensitive to very close patterns, and due to this reason, they decided to refine the DTW results by using an HMM. This refinement will occur when the outputs of the DTW are fed to the HMM. This reduces the number of classes to train in HMM, thus reducing missed classifications. So instead of DTW returning a sole class in its output, it returns the closest sequences to the input pattern belonging to the training set. Once HMM receives this sequence, it will have a higher probability of returning the correct class. Figure 47 display the architecture used by the authors.

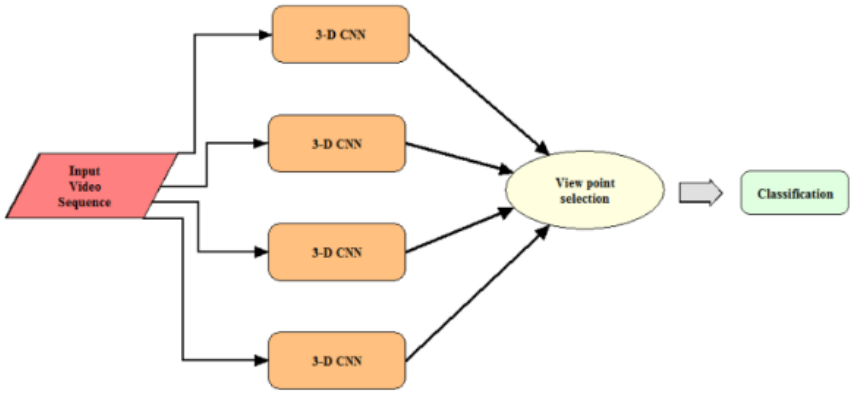
Figure 47 – Feature extraction step from (SANTOS; FERNANDES; BEZERRA, 2015)




Source: SANTOS; FERNANDES; BEZERRA (2015)

(SHARMA; KUMAR, 2021) proposes an ASL recognition with a cascaded 3D-CNN using ASLLVD dataset. The concept of a cascaded network is that multiple networks of the same type are used. In this paper, the author stated that they used four 3D-CNNs, one for every viewpoint of the dataset. Figure 48 displays the architecture used by the authors. The authors preprocess images by converting grayscale, applying a median filter to remove unwanted noise, and applying histogram equalization. Images are then resized to 512x384, and every sequence is reduced to 25 different frames. The authors do not inform how the viewpoint selection process occurs. They achieved a 96% accuracy.

Figure 48 – Architecture used from (SHARMA; KUMAR, 2021).

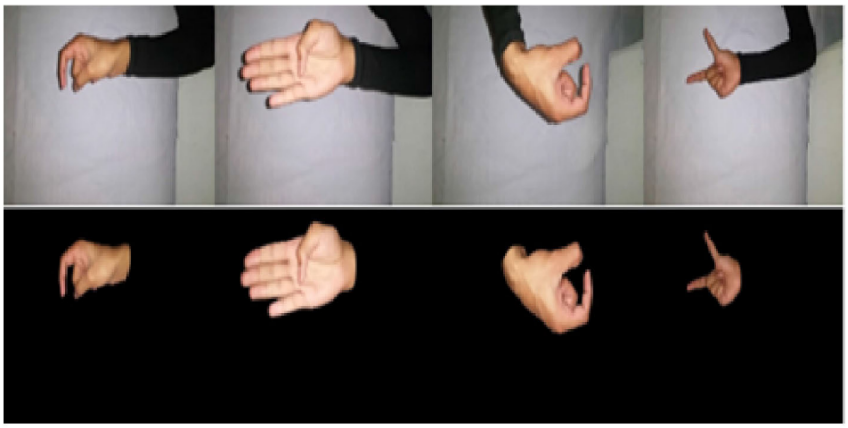



Source: SHARMA; KUMAR (2021)

(SHAH et al., 2021) propose a multi-kernel SVM to recognition Pakistan Sign Language (PaSL). The authors opted to use segmentation based on KNN clustering due to its accuracy for segmenting the fingers and palm area of the image from the rest of the image. After segmentation, images were converted to grayscale. The authors extracted four different features from the gray-scaled images: HOG; Edge orientation histogram (EOH); Local Binary Patterns (LBP); and SURF. Figure 49 displays some of the results after applying KNN clustering.

During training, the authors used three kernels for SVM: Gaussian; Linear; and Polynomial. These three kernels are applied to the four features to see which kernel performs best. The kernel selection uses K-fold cross-validation with K=10 and computes average accuracy through each of the three kernels and separately over each feature space. The kernel function over a single feature space (among the three kernels) that shows the highest average accuracy is selected. Once the best kernel is selected, it is used with the test data with the four previously mentioned features, thus resulting in four different outputs. These four outputs are then ensemble to produce the final result. Within the ensemble of SVM results, an algorithm is applied to calculate the class with the highest score. Figure 50 displays the architecture of

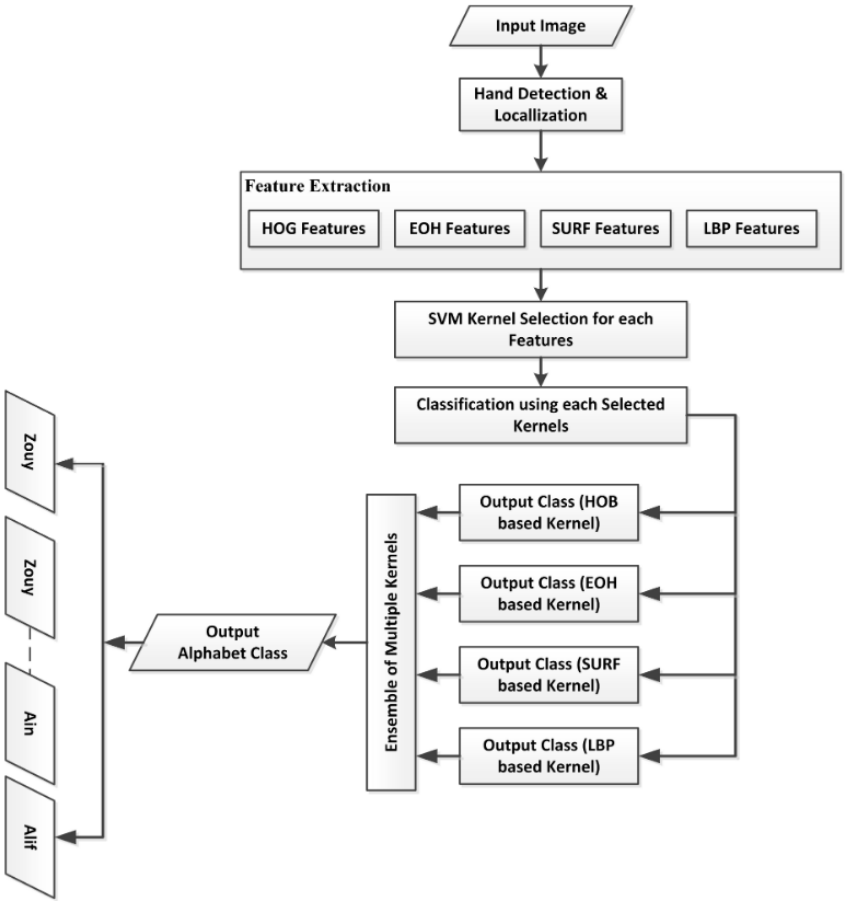
Figure 49 – Segmentation results after applying KNN clustering from (SHAH et al., 2021).



Source: SHAH et al. (2021)

the multi-kernel SVM and how it is applied within the four image feature. With this approach, the author achieved a 91.98% accuracy.

Figure 50 – Architecture used from (SHAH et al., 2021) consisted of a multi-kernel SVM.




Source: SHAH et al. (2021)

(BASNIN; NAHAR; HOSSAIN, 2021) proposed integrating a CNN with a LSTM to recognize Bangla Sign language. The dataset used by the authors consists of 13400 static images divided

into 36 classes. Images go through five pre-processing stages: Background subtraction; Grayscale conversion; Morphological erosion; Median filtering; and Resize (60x60). The authors augmented their dataset by rotating, scaling, shifting, and normalizing the images to increase the model's generalization. The authors' neural architecture comprises two convolution layers, followed by two max-pooling layers and two dropout layers. A time distributed flattening layer connects the CNN network with the LSTM network. According to the authors, they achieved a 90% and an 88.5% accuracy on the training and testing dataset in 80 epochs.

Figure 51 – Architecture used from (BASNIN; NAHAR; HOSSAIN, 2021) integrating CNN network with LSTM through time distributed flattening layer.

| Model Content | Details |
|---|---|
| 1st Convolution Layer 2D | Input Size 60x60, 64 filters of kernel size 3X3, ReLU |
| 1st Max Pooling Layer | Pool size 2x2 |
| Dropout Layer | Randomly deactivates 25% neurons |
| 2nd Convolution Layer 2D | 64 filters of kernel 3x3, ReLU |
| 2nd Max Pooling Layer | Pool size 2x2 |
| Dropout Layer | Randomly deactivates 25% neurons |
| Dropout Layer | Randomly deactivates 25% neurons |
| Flattening Layer | Time Distributed |
| LSTM Layer | 200 nodes |
| Dropout Layer | Randomly deactivates 25% neurons |
| Output Layer | 36 nodes, SoftMax |
| Optimization Function | Stochastic Gradient Descent (SGD) |
| Learning Rate | 0.001 |
| Momentum | 0.9 |
| Nesterov | True |
| Matrix | Loss, Accuracy |

Source: BASNIN; NAHAR; HOSSAIN (2021)

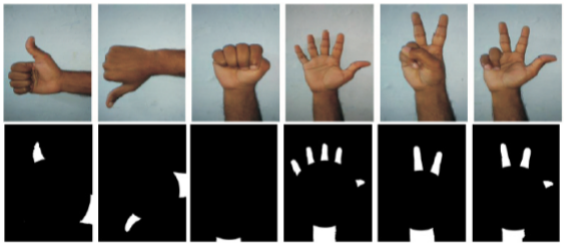
#### *3.2.2.2 Non Sign Language Context*

(SHARMA; VERMA, 2015) proposed a gesture recognition system to control multimedia applications. Gestures recognized are static with stable lighting and simple homogeneous background.

Images are converted to the YCbCr color space to perform skin segmentation. Morphological filters are then applied to increase quality and remove noise. Next, centroid and diameter are calculated, and with this information, a circle is drawn with the same background color (black). The result of this process is depicted in Figure 52.

The training was performed in two phases: hands positioned 16 cm and 21 cm from the camera. Then, the authors used these two distances to see which one achieved the highest

Figure 52 – Segmentation results from (SHARMA; VERMA, 2015).



Source: SHARMA; VERMA (2015)

accuracy.

No machine learning technique was used to classify the gestures, as the authors just counted the number of white objects in the image. The overall accuracy achieved with a 16 cm distance was 95,6% and 98,6% with 21 cm distance.

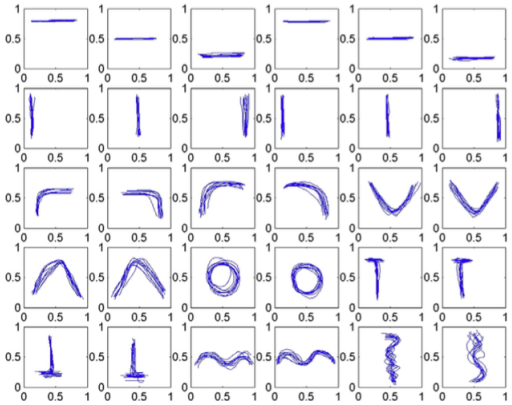
However, the background removal is not robust enough to handle different light conditions and different backgrounds, especially complex ones.

(YANG, 2013) proposed a Cellular Neural Network (CHUA, 1996) to find ROI and background regions. The main idea is to detect hands' ROI from a given video and send it over a network without losing the image quality of the ROI. The first part detects skin regions using a Cellular Neural Network template. Cellular Neural Network motion detection is performed on these results, followed by morphological structure elements to fill small gaps and smooth boundaries. These steps are important to detect the ROI of an image. To obtain the background, the ROIs are subtracted from the original image. The author (YANG, 2013) did not provide any details of the Cellular Neural Network used, and we did not have access to a paper in English by (ZHANG et al., 2013) mentioned in his references regarding this type of neural network. The authors did not show any classification results using their ROI. The final result focused on the ROI image quality presented by their work.

(CARIDAKIS et al., 2010) proposed a gesture trajectory classification using a combination of Self-Organizing-Maps (SOM) and Markov models. The authors did not dive into details about their feature extraction process. They only mentioned that they used skin segmentation to find the centroid of the hand in order to produce a series of coordinates. These coordinates were normalized so that users that occupy more or less space when performing a gesture would not introduce noise, bias, or scaling issues. In order to eliminate the issue of positioning the user in the right place, hand coordinates are relative to the position of the head. Once this is all done, coordinates are fed to SOM for training.

Each coordinate is represented by the Best Matching Unit (BMU) on the map using

Figure 53 – Dataset trajectory gesture from (CARIDAKIS et al., 2010).



Fonte: CARIDAKIS et al.

Euclidean distance. A Markov model is created for each gesture data based on the BMU. A directional model is then computed following the given equation:

$$OF(G) = \{v_1, v_2, ..., v_m\} : v_i = W_r(Q(arctan(\frac{y_i - y_{y-1}}{x_i - x_{i-1}})))$$

where $v_i$ is the quantized direction of a gesture, $W_r$ is a median filter and $Q$ is a quantization function.

A Generalized Median of the Markov model, given by:

$$M_j = generalizedmedian(D'_j) = argmin_g \sum_{G' \epsilon D'_j} L(g, G')$$

where $M_j$ is a generalized median of the dataset, and $L$ is the Levenshtein distance. The above formula is calculated in order to calculate the Levenshtein distance between $Mj$ and Markov models. Gestures are found calculating the probability of belonging to a category of a gesture:

$$P(G'_k|MM_j^{som}) = \frac{\prod_{i=1}^{q} S_i^{som}}{q} : q = |G'_k|$$

The authors used 30 gestures consisting of 10 repetitions. Figure 53 shows the dataset used by the authors. According to the authors, this approach achieved 100% accuracy.

The type of background used was not informed in the paper, but judging by the skin segmentation applied, it was simple and homogeneous.

### 3.2.3 Facial Expressions

Facial expressions have an important role in sign language. In the absence of intonation, facial expressions become a critical aspect of complementing the meaning of a sentence. According to (HUENERFAUTH; LU; ROSENBERG, 2011), facial expressions lead to significant improvement in the quality of the information exchanged in sign languages.

(BHUVAN et al., 2016) used nine different facial expressions from Libras in a user-dependent and independent recognition system. With a dataset consisting of nine expressions performed by two persons (identified as user A and user B), each performs five times the expression within a sentence. The dataset comprises 100 facial points obtained from a Kinect camera sensor. In addition, a user-independent experiment was performed by combining all the facial points from users A and B within one image. The authors did not inform how the 100 points were obtained nor how the user-independent experiment was performed.

The authors used different machine learning algorithms such as: MLP with ten neurons in the hidden layer; Bayes Network (BEN-GAL, 2007); RBF Classifier (BROOMHEAD; LOWE, 1988); Random Forest (no limit to maximum depth and 10 trees); Bootstrap Aggregating (BREIMAN, 1996); AdaBoost; Logit Boost (FRIEDMAN et al., 2000). The authors stated that Random Forest was the technique that achieved the best results: 93% for user-dependent and 97% for user-independent.

(ELONS; AHMED; SHEDID, 2014) proposed a facial expression recognition system of the ArSL. The authors recognized six facial expressions and used the RPCA for feature extraction and MLP for classification. According to the authors, adding this functionality improves classification because some gestures, such as marriage and divorce, are the same but are differentiated by the accompanying facial expression (happy or sad). To test this approach, the authors used an existing ArSL recognition system, which achieved an 88% classification rate on dynamic gestures.

Face position and region size are detected using Viola and Jones object detection (VIOLA; JONES, 2004). Once the face is recognized, RPCA is used for feature extraction. The authors opted for this feature extraction technique because it is less computational demanding in a real-time application. This is achieved because the covariance matrix is decomposed into eigenvectors and values, which reduces the time perturbation step interval in the diagonal eigenvalue matrix resulting in fast convergence and accurate sub-space tracking.

The facial dataset consisted of "Very happy", "Smiling", "Sad", "Surprised", "Normal"and

Figure 54 – Ada-Boost feature selection classification increase from (UDDIN, 2015).

| GFE | SVM-All | RF-All | RF-Ada-Boost |
|---|---|---|---|
| WH-questions | 0.9553 | 0.9850 | **0.9853** |
| Yes/No-questions | 0.9489 | 0.9798 | **1** |
| Doubt questions | 0.9629 | 0.9820 | **0.9833** |
| Topics | 0.969 | **0.9783** | 0.9770 |
| Negative | 0.9085 | 0.9848 | **1** |
| Assertion | 0.92 | 0.9701 | **1** |
| Conditional Clauses | 0.9579 | 0.9859 | **0.9866** |
| Focus | 0.9684 | 0.9855 | **0.9867** |
| Relative Clauses | 0.9736 | **0.9919** | 0.9918 |

Fonte: UDDIN

"Looking up"performed by five different persons in four different angles and three different distances. Tests showed that 360 RPCA features and 20 hidden neurons in the MLP achieve a 98% classification rate, resulting in a 10% increase in the original accuracy.

(UDDIN, 2015) proposed a facial recognition system based on Ada-Boost feature selector and Random-Forest to classify nine different facial expressions in Libras. The dataset was based on facial expressions from Libras where each expression had 100 features. The purpose of this work was to see how much Ada-Boost feature selection can increase classification.

Ada-Boost was used to automatically select a subset of each facial expression feature vector, thus resulting in a new vector comprised of 58%-61% of the original one.

According to the authors, Ada-boost increased the classification rate by approximately 8%. Figure 54 displays the classification increase for the nine facial expressions.

(KOLLER; NEY; BOWDEN, 2015) proposed mouth shape recognition incorporating CNN classifier outputs into a HMM, allowing a forced temporal alignment and iterative learning on videos.

The authors used a previously trained CNN model for the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) 2014, a 22 layer architecture. Convolutional layers and the last fully connected layer used rectified linear units as non-linearity, and the dropout layer, with a 70% ratio of dropouts, was used to prevent overfitting. The output layer was replaced with a 40 dimensional, fully connected layer. After a successful CNN training, the model's softmax outputs were used in an HMM framework to add temporal information and re-align the data to measure the final alignment error. To successfully decode a mouth shape sequence, the authors performed a maximum likelihood search on the visual model and converted the CNN's posterior output to likelihoods using the Bayes' rule.

According to the authors, there is no dataset on mouth annotations, making the problem

harder since the neural network must train on weakly labeled data. The authors used mouth-ground truths created by (KOLLER; NEY; BOWDEN, 2014). This ground truth is based on the publicly available dataset for sign language RWTH-PHOENIX-Weather corpus. The authors achieved a 55% classification accuracy.

#### *3.2.3.1 Gaze Gesture Recognition*

Another increasing research area is gaze gesture, which involves recognizing an ordered sequence of gaze positions over time. This type of recognition identifies the direction of the eye in a given space. This is helpful to people who do not have body movement to interact with computers through the gaze. Another application is to support communication between people mediated by this technology. Eye positions can have different outputs in a system producing vocabularies of meanings and actions. Death persons who also lack hand movement could benefit from this communication technique.

Since gaze recognition is highly spatio-temporal dependent, HTM is an appropriate technique due to its spatio-temporal neural architecture. Three works used this type of recognition with HTM in three different ways: traditional HTM with spatial layer, temporal layer, and a top classification layer; extended HTM with an added top layer to store sequences of incoming vectors; and Optimized Hierarchical Temporal Memory (OHTM), with multiple additional top layers to better classify temporal sequences. (ROZADO; RODRIGUEZ; VARONA, 2011) used the traditional HTM to recognize 50 gaze gestures. The dataset consisted of 1 person performing 30 times each gaze gesture within 60cm distance of the camera. The authors achieved a 96% classification rate. (ROZADO; RODRIGUEZ; VARONA, 2012b) used the extended HTM to classify ten gaze gestures. One user performed the dataset 50 times each gesture, thus achieving a 98% classification rate. Moreover, (SRUTHI et al., 2017) used the OHTM also to classify ten gaze gestures. The dataset used by the authors consisted of 10 gaze movements performed by one person 50 times. The authors achieved the same result as the previous work, a 98% classification rate, but using a six-layer network.

### 3.2.4 Datasets

This section presents the public datasets used by the papers in this review, divided into Facial Expressions, Sign Language, and General Gesture datasets.

#### 3.2.4.1 Sign Language Datasets

A dataset based on ASL created by (BARCZAK et al., 2011)[13] was used to recognize static gestures. This dataset comprised 36 static gestures and was populated under controlled light and a simple background to facilitate feature extraction. The authors added rotated and stretched images to increase the number of images in the dataset. The authors who used this dataset, (PAN et al., 2016), achieved a 94% classification rate.

Another publicly available dataset by (PUGEAULT; BOWDEN, 2011) was used to recognize static gestures. This dataset consists of 24 static gestures from the ASL alphabet. The images were captured in five different sessions using Microsoft Kinect sensors, with similar lighting and background. Each sign contains about 500 images generated by five users. Results achieved using this dataset by (LI et al., 2015) were 99%.

The Australian sign language dataset used by (ROZADO; RODRIGUEZ; VARONA, 2012a) was from the UCI Machine Learning repository[14]. This dataset consists of 95 gestures with 27 examples for each gesture. Data was captured using a glove sensor-based device containing $x$, $y$, $z$ coordinates, pitch, and yaw information. The classification achieved was 91%.

The RWTH-PHOENIX-Weather multi-signer 2014[15], is a dataset for continuous sign language recognition. This dataset contains 5,672 sentences in German sign language for training with 65,227 signs and 799,006 frames in total. Nine signers perform these videos. (CUI; LIU; ZHANG, 2017) achieved a 61.3% classification result using this dataset. This dataset was updated to a version called RWTH-PHOENIX-Weather-14T[16] and was used by (CAMGOZ et al., 2020a) and (YIN; READ, 2020). This dataset contains more than 8000 sentences performed by nine different signers totaling 2 million images. The authors achieved a 24.5% and 21% WER, respectively.

The RPPDI Dynamic Gestures Dataset (BARROS et al., 2013) displayed in Figure 55, is a publicly dynamic dataset that contains seven dynamic gestures with several examples per gesture where each gesture is composed of 14 frames. The authors who used this dataset, (BARROS et al., 2017), achieved 81.66% using CSURF features extraction and HMM. Another work that used this dataset, (SANTOS; FERNANDES; BEZERRA, 2015), achieved a 98.43% by combining DTW and HMM to create a hybrid system.

[13] https://mro.massey.ac.nz/handle/10179/4514
[14] https://archive.ics.uci.edu/ml/datasets/Australian+Sign+Language+signs+(High+Quality)
[15] https://www-i6.informatik.rwth-aachen.de/ koller/RWTH-PHOENIX/
[16] https://www-i6.informatik.rwth-aachen.de/ koller/RWTH-PHOENIX-2014-T/

The ASLLVD is a publicly available dataset that is comprised of more than 3300 ASL video sentences executed by six different signers. It also contains multiple viewpoints of the person performing a gesture. For example, (SHARMA; KUMAR, 2021) used a cascaded 3D-CNN and achieved 96% accuracy.

Figure 55 – RDDPI dataset.


Source: BARROS et al. (2017)

The KETI Sign Language dataset used by (KO; SON; JUNG, 2018a) is comprised of 10,480 videos in 1920 x 1080 resolution from 10 different signers. The dataset has 100 sentences for emergencies and was captured in the simple one-colored background. For example, figure 56 displays a frame from the sentence "I am burned". By using two bi-directional GRU's stacked, (KO; SON; JUNG, 2018a) achieved 89.5%.

MSRGesture3D is a dynamic hand gesture dataset captured by the Kinect RGB-D camera. There are 12 dynamic hand gestures in ASL, and each dynamic gesture was performed two or three times by each of 10 subjects. The gestures are: "bathroom", "blue", "finish", "green", "hungry", "milk", "past", "pig", "store", "where", "J"and "Z".

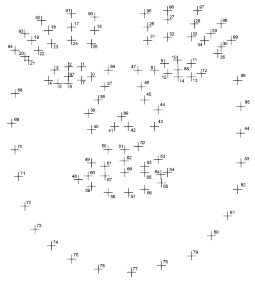
The authors (SANTOS; FERNANDES; BEZERRA, 2015) achieved 97.5% using a hybrid DTW and HMM system.

We were able to find a word based Libras dataset called V-Librasil that has more than 1350 gestures and 4000 examples performed by 3 different users. In Chapter 4 we detail this dataset and how it was used in our research.

Figure 56 – Example frame from KETI Sign Language dataset.

Source: KO; SON; JUNG (2018a)

Figure 57 – One hundred facial points from UCI Machine Learning Repository of the Grammatical Facial Expressions dataset.



Source: https://archive.ics.uci.edu/ml/datasets/Grammatical+Facial+Expressions (2022)

#### *3.2.4.2 Facial Expression Dataset*

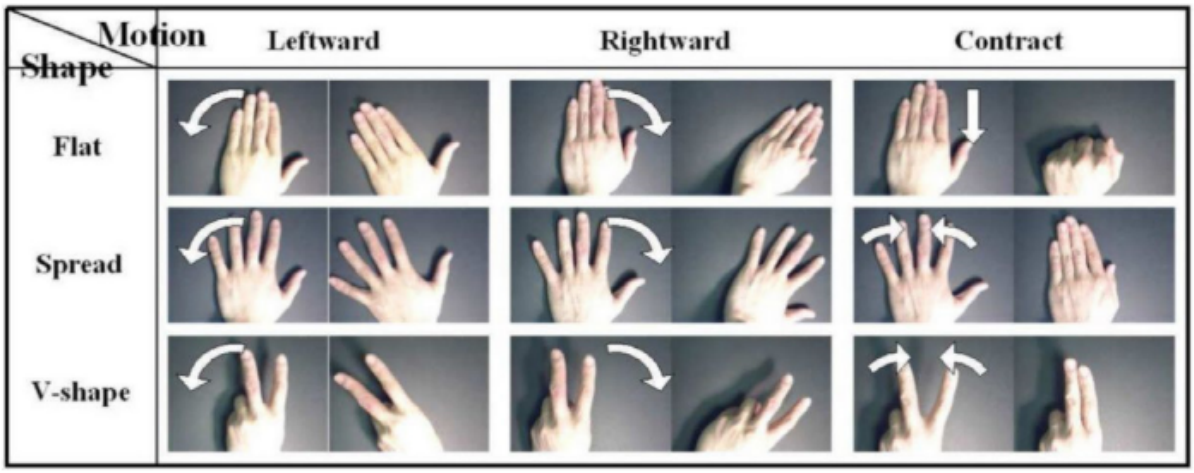
The facial expression dataset used in the reported works was from the UCI Machine Learning repository[17]. This dataset comprises nine facial expressions performed by two persons resulting in 18 videos. Each person performs five sentences in Libras in each video, five times each, requiring a grammatical facial expression. A total of 100 facial points were acquired using the Microsoft Kinect camera (Figure 57).

The dataset comprises a text file containing one hundred coordinates ($x$, $y$, $z$) of points from eyes, nose, eyebrows, face contour, and iris; each line in the file corresponds to points extracted from one frame. Using this dataset, (UDDIN, 2015) obtained a 95% classification rate and (BHUVAN et al., 2016) obtained the results showed in Figure 54.

The mouth dataset based on the RWTH-PHOENIX-Weather multi-signer 2014, discussed

[17] https://archive.ics.uci.edu/ml/datasets/Grammatical+Facial+Expressions

Figure 58 – Examples from Cambridge Hand Data.




Source: BARROS et al. (2017)

previously, has annotations of 5 sentences per signer on the frame level with 39 labels and one garbage label (used when encountered unclear or non-mouth shapes). The authors who used this dataset, (KOLLER; NEY; BOWDEN, 2015), achieved a 55% accuracy.

#### *3.2.4.3 Specific Gesture Datasets*

A publicly available gesture spotting dataset from (ESCALERA et al., 2014) was used to recognize Italian gestures (not sign language-related). This dataset consists of 20 different Italian gestures performed by 27 users with variations in surroundings, clothing, lighting, and gesture movement. The videos are recorded with a Microsoft Kinect. The authors, (PIGOU et al., 2014), achieved a 90.7% classification accuracy.

The Cambridge Hand Data dataset comprises 900 image sequences separated into nine hand gestures classes. In addition, the dataset is divided into five different types of illumination and contains ten sequences executed by two separated subjects resulting in a total of 100 sequences per class. Therefore, each sequence has a different number of images. Figure 58 displays some examples of this dataset. The authors who used this dataset, (BARROS et al., 2017), achieved a 93.98% accuracy.

Other datasets were used by some of the works analyzed, which were created specifically for the authors' experiments and are not publicly available. These datasets are not focused on sign language nor applied to a real-world deaf communication scenario but are still related to hand movement and gesture recognition. For example, recognition based on gesture trajectory, number of fingers raised, and video game controller positions. Unfortunately, datasets based

on glove experiments need the same hardware to be reproduced, making public availability difficult.

## 3.3 FINAL THOUGHTS

This literature review gives an overview of the techniques used for gesture and facial expression recognition in sign languages. Using various types of hardware and techniques (whether for feature extraction or classification), papers achieved a high classification rate, but some constraints were observed analyzing the setup, environment, dataset, feature extraction, and classification technique.

Regarding setup, various papers used special hardware such as gloves, cameras, or robots (Table 5). The usage of special hardware facilitates feature extraction and helps overcome background removal, hand location, orientation, and finger position within an image. This information is obtained through sensors (gyroscope, flex, accelerometers), specific color search (when using a colored glove), dedicated hardware such as Kinect camera, PlayStation camera, accessories, robots, etc. However, special hardware is mostly cumbersome (e.g., making a user wear a large glove connected to a microcontroller as seen in Figure 15 and 29, or making use of a video game console as seen in (KHAMBADKAR; FOLMER, 2014)) and affects portability. Having extra equipment to carry and setup is an undesired burden. Each of them has downsides that should be taken into consideration.

Taking into account the environment used, two types of background were observed: simple and complex (Table 6). Most of the papers dealt with images with a simple background. This type of background makes information easier to filter because there are only the hands in the image. On the other hand, the complex background has a lot more information that needs to be filtered and, if not correctly done, can have a negative impact on the classification rate. Therefore, results presented in the simple background would be completely different in complex background.

Another environment constraint observed was where the experiment took place. Some works applied uniquely to particular environments such as office (PAULSON; CUMMINGS; HAMMOND, 2011) or hospital (LUO; WU; LIN, 2015) and most commonly, under controlled lighting. Such environments do not represent a real-world scenario where background and light constantly change, or people are in different places. Having a fixed place to recognize gestures makes the system less usable. The constant changing represents a significant problem in feature

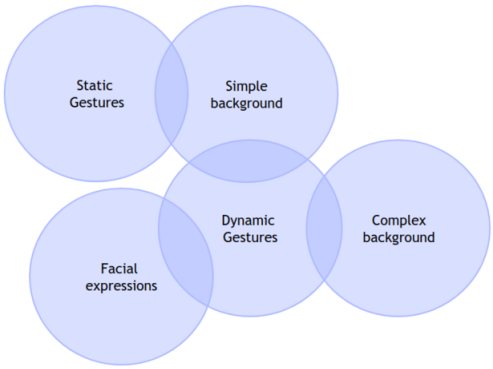
extraction because it is hard to predict what will happen next.

The datasets observed varied in size and type of gestures to be classified (e.g., amount of fingers raised, hand movement, static and dynamic gestures). Classifying hand movement (CARIDAKIS et al., 2010) and a number of fingers raised (LUO; WU; LIN, 2015) do not apply to sign languages in a real-world scenario. Another point to note is the number of gestures in a dataset. It is easier to achieve a good classification accuracy with a small number, but that does not apply to a real sign language composed of thousands of gestures. Out of the 50 papers analyzed here, only 10 used publicly available datasets. We notice that some sentence sign language datasets are made for one specific context. For instance, the PHOENIX-14T is for the weather forecast in German, KETI is for emergencies in Korea, and most do not apply for day-to-day situations. Finding a dataset or creating one, especially in Libras, that can fulfill these criteria is very challenging.

Feature extraction varied the most, with 26 different techniques (Table 1). The two most widely used were sensor information extraction and skin segmentation. These techniques are used in different contexts. For instance, sensors are used with special hardware (discussed previously), while skin segmentation is performed by analyzing the image's pixels via software. This technique presents a drawback that if the skin tone changes or if there is a color similar to the skin, which can lead to unwanted information in the segmentation result. As for classification techniques, most had similar classification rates making it hard to determine the best gesture recognition. Except for brute force comparison, all other techniques brought a wide range of acceptance when performing a gesture, meaning that a gesture can be performed slightly differently and still be recognized as such. For the brute force comparison, gestures would have to be done precisely in the same way to have the exact same results, thus not giving any flexibility to the signer. It would be interesting to see the results using a neural network in the data from sensors. Most state-of-the-art algorithms such as CNN and HTM are seen here depending on a good feature extraction technique or special hardware such as the Kinect camera to achieve high classification results.

In conclusion, this review shows that the high classification rates presented in the literature must be interpreted, considering the equipment used and the restrictions imposed to allow the classification. It was observed that most papers either recognized static or dynamic gestures and either used simple or complex backgrounds. We found only one paper that combined facial expressions with Sign language recognition. We identified five major research areas currently being studied regarding this area: static gestures, dynamic gestures, simple background, com-

Figure 59 – Major research areas in Sign language recognition and how they intersect with one another.




Source: Created by the author (2022)

plex background and facial expressions. We can see that there is no intersection involving all five areas as displayed in Figure 59.

Our final considerations from this literature review involve the following aspects:

1. Replicability - To achieve a high classification rate, most works rely on external hardware such as depth cameras and gloves and specific lighting conditions, making it hard to replicate the experiments.

2. Result Analysis - Most works do not compare their approach with other techniques nor provide a statistical analysis of the results. This would make it easier to verify if one technique outperformed another.

3. Number of Gestures and Computational Cost - Sign language has numerous gestures. Most of the works use a small dataset size and recognize less than ten gestures. The computational cost of recognizing thousands of gestures in a real-time application is quite challenging.

4. Social Inclusion - Although all of the works recognize gestures, most of them are not focused on a deaf person's perspective, i.e., gestures do not convey meaning in a sign language conversation.

## 3.4 FUTURE RESEARCH DIRECTIONS

This literature review showed that the research area of gesture recognition is scattered with different approaches, datasets, and techniques, with several research opportunities to achieve a more established solution for the context of sign languages.

We understand the diversity of datasets in sign language throughout the works, but it would be interesting to see future works comparing their technique to at least one consolidated dataset in academia instead of just using a specific dataset. This way establishing a dataset can consolidate, improve and allow better comparison between techniques. From all datasets analyzed in this work, there are: PHOENIX-14T[18]; ASLLVD[19]; ASL-Lex[20]; and Database for hand gesture recognition[21] that are easily available for download. As for Libras, (RODRIGUES, 2021) released a Libras word dataset called V-Librasil[22] that can be used for future research in Libras recognition.

State-of-the-art techniques such as CNN and Transformers show strong potential to improve gesture recognition, especially in complex backgrounds, and could also facilitate feature extraction by automatically learning features. It would be interesting to see that further research using these techniques could help create an intersection between the five major sign language recognition problems shown in Figure 59, achieving a complete sign language recognition system.

Last but not least, facial expression is an essential component in sign languages that are still left out from most works. Recognizing facial expressions is an important step to convey meaning (exclamation, question, emotion, etc.) to a sentence in sign language. However it is challenging to combine recognition of hand movements and facial expressions due to the complexity of facial muscular movement that can be sensitive as it involves eyebrow, cheeks, mouth, head position and forehead.

## 3.5 RESEARCH QUESTIONS

Based on our reading across the literature, we formulated the following research questions.

- Question 1: How to perform translation between sign languages?
  - We tackled this problem in our previous work (NEIVA; ZANCHETTIN, 2016), achieving 61% accuracy recognizing static signs using ELM. We want to improve our results and tackle dynamic gestures for this work. We decided to use Deep learning, more precisely Mask-RCNN, CNN, and Transformers, for the following reasons: the

[18] https://www-i6.informatik.rwth-aachen.de/ koller/RWTH-PHOENIX/
[19] http://www.bu.edu/av/asllrp/dai-asllvd.html
[20] http://asl-lex.org/
[21] http://sun.aei.polsl.pl/ mkawulok/gestures/
[22] https://libras.cin.ufpe.br/

ability to create precise masks to be subtracted from original images; the ability to extract features automatically; and interesting results in time-series problems, respectively. By the time of writing this thesis, we could not find any work that combined these techniques for translation between sign languages.

- Question 2: Can a word-based sign language dataset be used to recognize sign language sentences?
  - This question was brought up because we noticed that it is hard to find a publicly available sentence-based sign language dataset for day-to-day situations. Although some datasets are sentence-based, they are specific for one context (e.g., weather forecast). Others are word-based, being inapplicable for sentence recognition. We propose using a word-based dataset to combine words to form sign language sentences, thus decreasing the time spent performing sign language sentences to create datasets. This is helpful when having similar sentences (i.e., my house is red, my house is blue, my house is tall).

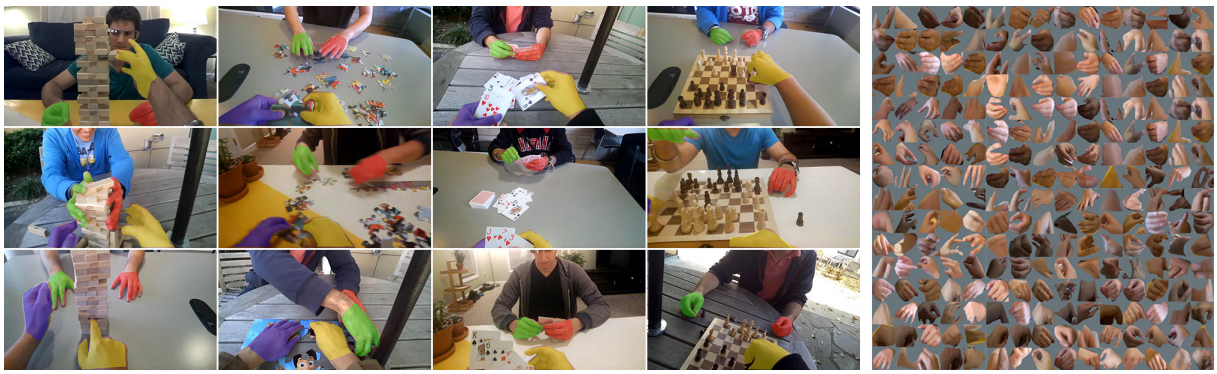
# 4 METHODOLOGY

In this work, our goal is to translate between sign languages, and our approach to accomplish this is to remove background information from images firstly; secondly, recognize the gesture being performed in a video; and finally, translate the recognized gesture. This chapter explains how we used our datasets and divided our data into training and testing.

## 4.1 DATASET FOR MASK R-CNN

For the first step of removing the background and leaving just the hand in the image, we needed a dataset to train a Mask R-CNN to remove the background. However, this approach was not seen in our literature review and none of the publicly available datasets found in the literature review fit this goal. Hence we opted to use EgoHands from (BAMBACH et al., 2015). This dataset contains 15k frames with annotated hands. Figure 60 displays some annotated hands images from this dataset. This helps us create hand masks from images. This dataset was not divided into training and testing since we were not benchmarking our results as we were just interested in the final mask output.

Figure 60 – Sample images from EgoHands dataset (BAMBACH et al., 2015) with different color hand masks followed by images containing only the hands.



Source: BAMBACH et al. (2015)

Figure 61 – Sample images from six different users from PHOENIX-14T dataset (CAMGOZ et al., 2020b) in DGS in a weather forecast context.

Source: CAMGOZ et al. (2020b)

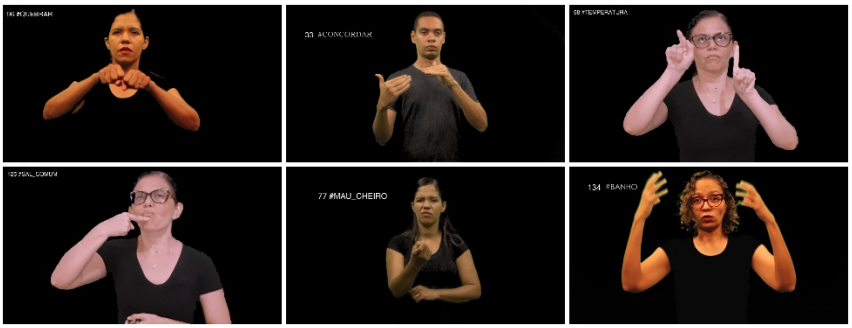
## 4.2 DATASETS FOR SIGN LANGUAGE RECOGNITION

### 4.2.1 PHOENIX-4T

Regarding the recognition of the gestures, we needed a baseline to evaluate the efficiency of our code. From the publicly available sign language datasets found in our literature review, we opted to use PHOENIX-14T because it has been used by related works with Transformers architecture (CAMGOZ et al., 2020b; YIN; READ, 2020) to recognize sign language gestures from videos. This dataset contains more than 8,000 video sentences with more than 2,000 words from the German Sign language, within a forecast prediction context, and is performed by nine persons and has more than 2 million images (Figure 61) which requires more memory and training time to traverse all of the data.

We used this dataset in three different approaches: 1) using CNN features provided from (CAMGOZ et al., 2020b); 2) training a CNN and extracting features from images as-is; 3) and pre-processing images using our Mask-RCNN and training and extracting CNN features. To train the Efficientnet on this dataset, we decided to use every sentence as a class thus resulting in approximately 8000 classes.

Figure 62 – Sample images from V-Librasil dataset from (RODRIGUES, 2021) performed by six different persons under controlled and simples background.




Source: RODRIGUES (2021)

The first approach was used to validate our Transformers implementation to see if we could achieve the same results from the authors. We used the entire features set provided by the authors, which consisted of 7096 training, 519 validation, and 642 testing videos. This helped us to verify that our code was achieving state-of-art results.

However, training an Effiencientnet CNN to extract features from this dataset is time-consuming. Therefore, we used half of this dataset in our experiments due to time and memory limitations. We divided the dataset using approximately 3800 training, 260 validation, and 320 testing videos. The validation set is used to check wether our model is over or under fitting so that we can change our model parameters. CNN training was performed per sentence, where each sentence is a class. From this division, we fine-tuned an Efficientnet from Imagenet weights using entire images (second approach) and pre-processed images using our trained Mask-RCNN to leave the hands in the image (third approach).

### 4.2.2 V-Librasil

Additionally, since our work is based in the Brazilian language, we tested our translation approach with a Libras dataset. From the publicly available Libras datasets, we opted to use V-Librasil (RODRIGUES, 2021) as it contains the highest amount of gestures, performed by three different persons and approximately 1,350 words and 4,200 examples. Each gesture is performed by 3 different persons (Figure 62). This dataset is significantly smaller and because of this, presents generalization challenges since DL algorithms require a huge amount of data to achieve good results.

To test V-Librasil word recognition, this dataset was divided into 3,800 training, 250 validation, and 336 testing examples and was CNN trained using the same process as PHOENIX-14T:

using the entire image and pre-processed image. We adopted the same strategy of using every word as a class when training the Efficientnet. This approach resulted in approxmately 1350 classes.

As we wanted to verify our network ability to predict gloss sentences (HUANG et al., 2019) in Libras, we created a sentence-based dataset by combining V-Librasil words since it is a word-based dataset (i.e., each gesture symbolizes one word). We do not need videos of sentences representing similar things with this approach. For example, let us consider the following words in Portuguese: "vamos"(let's), "comprar"(buy), "vinte"(twenty), "gravata"(tie), "vassoura"(broom), "urso"(bear), "pulseira"(bracelet) and "azul"(blue). With these words, we can create the following gloss sentences:

1. vamos comprar vinte gravata azul (let's buy twenty blue ties)
2. vamos comprar vinte vassoura azul (let's buy twenty blue brooms)
3. vamos comprar vinte urso azul (let's buy twenty blue bears)
4. vamos comprar vinte pulseira azul (let's buy twenty blue bracelet)

As we can see, these sentences are very similar. However, we need to have each performed separately in a sentence-based dataset. In our approach, we combine the eight words to form each sentence, thus only needing to have the gestures for the words.

We created 68 sentences (3-5 words long) by manually combining words from the V-Librasil dataset and generated 400 training examples, 64 validation examples, and 46 testing sentences.

### 4.2.3 Dataset for translation

To train our Transformers for the NLP translation problem, we used two Portuguese to English translation datasets: one from (TIEDEMANN, 2020), and another from Apertium[1]. The first dataset is a collection of thousands of sentences and translations in various languages. It contains 180K Portuguese sentences with their respective English translation. Table 9 shows three examples from this dataset (lines 1 to 3). The second dataset is word-based with approximately 8K word examples. Table 9 (lines 4 to 6) displays examples from Apertium.

We encountered two problems when translating from Portuguese to English: 1) when using the dataset from (TIEDEMANN, 2020) which is sentence based, we encountered some

[1] $https://github.com/apertium/apertium-en-pt$

Table 9 – Examples 1 to 3 from (TIEDEMANN, 2020) Portuguese to English translation dataset and 4 to 6 from Apertium dataset.

| | Portuguese | English |
|---|---|---|
| 1 | Tom quer praticar francês com falantes nativos. | Tom wants to practice French with native speakers. |
| 2 | O que você vai fazer antes de almoçar? | What are you going to do before lunch? |
| 3 | Não devemos gastar nossos recursos energéticos. | We mustn't waste our energy resources. |
| 4 | auditório | auditorium |
| 5 | barreira | barrier |
| 6 | censura | censorship |

Source: Created by the author (2022)

translation problems since some words were being translated to sentences instead of words (e.g., "doente"(sick) resulted in "I feel sick", or "I am sick"); and when training using the Apertium, our translation accuracy was very low due to its small size. Because of this, we decided to first train on the dataset from (TIEDEMANN, 2020) and fine-tune our trained model with the Aperthium word-based dataset.

## 4.3 IMPLEMENTATION DETAILS

In this section we describe what metrics we used, our environment, libraries and implementation details.

### 4.3.1 Metrics and Development environment

When dealing with Natural Language Processing (NLP) problems such as translations, it is common to use BLEU scoring (PAPINENI et al., 2002). But since we are performing Sign language recognition, according to (KOLLER; FORSTER; NEY, 2015), WER is the recommended form of assessing recognition models. The formula to calculate WER is given below:

$$WER = \frac{Insertions + Deletions + Substitutions}{NumberOfWords}$$

We developed, trained and tested our DL algorithms (Mask R-CNN, Efficientnet and Transformer) using Python, Tensorflow[2] and Keras[3] libraries. Our development computer consisted of a second generation Ryzen 7, 16GB 3000MHz, and an RTX Titan with 24GB.

[2] https://www.tensorflow.org/
[3] https://keras.io/

### 4.3.2 Our Transformers

Our Transformers consisted of stacks of 3 encoders, 8 decoders, 512 hidden units, dropout rate of 0.1 and batch of 32. We used Adam optimizer with learning rate of $10^{-4}$. We used values of 0.9 and 0.998 for Adam's $\beta 1$ and $\beta 2$ parameters. We trained our network from scratch using Xavier initialization (GLOROT; BENGIO, 2010). We trained the Transformers unil we were able to achieve (CAMGOZ et al., 2020b) result of 24% WER and adopted this value for the rest of the training for this dataset. We verified that we achieved the authors result in 100 epochs. As for V-Librasil we stopped training when our model could not decrease anymore the WER value for 10 epochs. We verified that our stopping criteria went until 150 epochs.

# 5 SAIGNS: A SIGN LANGUAGE TRANSLATION SYSTEM

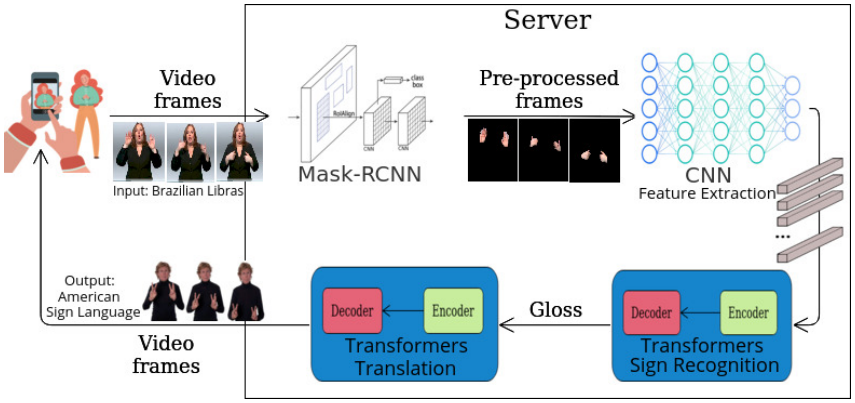
This chapter presents sAIgns: our proposed Sign language translation system, comprised of a mobile application and a web page. Our system's primary goal is to allow deaf persons to communicate with people who use Sign languages from other countries. This means, for example, that a deaf person that only knows Libras could use our mobile application on his/her smartphone to capture ASL gestures from another person, and the system would display the corresponding gestures in Libras. Furthermore, the users can visualize different Sign languages and view translations on the web interface. In addition, the webpage presents other functionalities so that the community of users can help create and improve Sign language contents by customizing hand masks and setting start and endpoints of gestures in videos.

## 5.1 SAIGNS ARCHITECTURE

sAIgns' architecture comprises four modules: background removal, feature extraction, gesture recognition, and Sign language translation. Figure 63 displays our architecture flow. The mobile application sends video frames to the web server, while the server is responsible for background removal, feature extraction, gesture recognition, and translation.

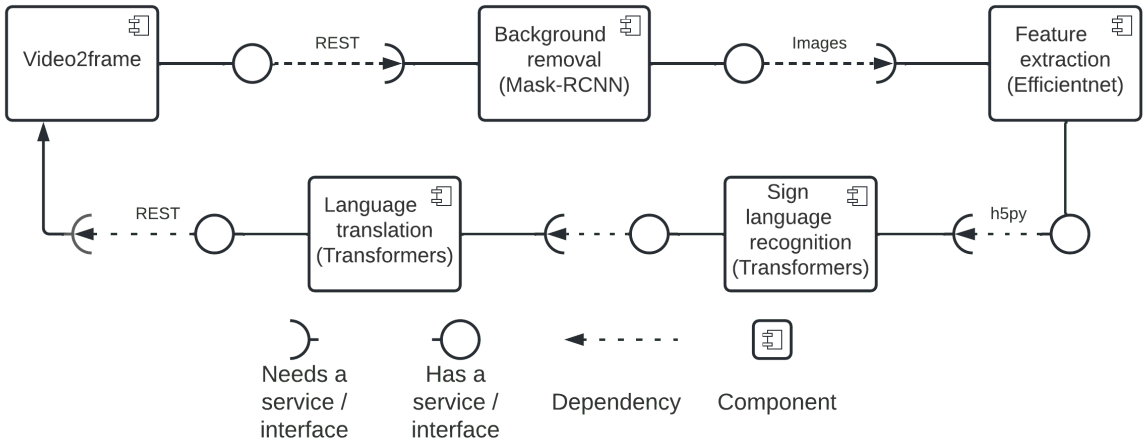
Figure 64 presents a component diagram of our architecture. This is a UML structural diagram that helps visualize the organization of system components and the dependency relationships between them. They provide a high-level view of the components within a system.

Figure 63 – Overall view of the proposed system. The smartphone captures video and sends the frames to the server. The server performs background removal and features extraction on the frames. The first Transformer uses those features to recognize glosses. The second Transformer uses the glosses to translate to a destination language and sends video frames of the corresponding translated gloss back to the smartphone to be displayed to the user.




Source: Created by the author (2022)

Figure 64 – Component diagram of sAIgns architecture and its dependencies. Each square represents a component in the proposed solution, and every line represents how each component relates and communicates with one another.




Source: Created by the author (2022)

Our diagram starts with the "Video2Frame"component present in the mobile device. This component outputs data and sends it through the REST protocol to the next component. On the other hand, the background removal component needs input to execute its process to output data to the next component. Its output data are pre-processed images. The feature extraction component follows the same idea: it needs input to output data. Its form of communication with the next component is through h5py data, where extracted features are saved. Finally, the Sign language recognition component needs this h5py data to process and send its output to the Language translation component. This last component performs translation and sends back to the user the data through REST protocol.

We tackled three problems in Sign language recognition: 1) unwanted information regarding a gesture; 2) dynamic Sign language gestures (time-series problem); and 3) translation between Sign languages (Sign languages are not universal).

## 5.2 UNWANTED INFORMATION REGARDING SIGN LANGUAGE GESTURE

As mentioned before, Sign language involves gesture movements to convey an idea or meaning. For a computer, information not pertaining to a gesture, such as a background information, can negatively influence the recognition of a gesture as the computer will consider everything given as input.

Approaches involving special hardware such as sensors and special cameras generally provide

precise information. For instance, LeapMotion uses a sensor to get a hand skeleton, while Kinect uses an RGBD camera to capture a person's body in an environment. However, downsides to this approach include the lack of mobility due to needed specific equipment and higher cost.

Software-based approaches such as skin detection, CNNs, and pose detection can be used with any device anywhere, but they do not provide the same levels of precision as hardware-based approaches. For instance, skin detection is very prone to light conditions (KOLKUR et al., 2017), CNNs can include unwanted features, and pose estimation produces different skeletons in every frame, causing instability (KONRAD; MASSON, 2020).

## 5.3 DYNAMIC GESTURES

We mentioned that dynamic gestures contain multiple images to convey meaning (whereas static gestures only need one image or video frame), resulting in a time-series problem. The Libras gesture for a bottle of wine (Figure 65) is an example of this type of gesture: it starts with the gesture "bottle,"and it ends with the gesture "V"on the cheek. We mentioned in Chapter 2 that LSTM and GRU achieved state-of-the-art results in time series problems but as the problems become bigger, they tend to have slow training time, due to the lack of parallelization and context loss with long inputs. Transformers were introduced to solve these problems and have been surpassing state-of-the-art LSTM and GRU results (VASWANI et al., 2017a; ZEYER et al., 2019) For these reasons, we opted to investigate Transformers in the context of Sign language, especially with Libras.

Figure 65 – "Bottle of wine"sample from V-Librasil dataset from (RODRIGUES, 2021). The sign starts with the hand in one configuration and ends with a different hand configuration thus representing a dynamic gesture.


Source: RODRIGUES (2021)

## 5.4 MULTIPLE SIGN LANGUAGES WORLDWIDE

We mentioned previously that every country has its sign language, just like its oral language. While verbal languages such as English, French, and Spanish are well spread around the world, meaning that a person can easily find a language school to learn from, this does not apply to Sign languages. Therefore, it can be challenging for a deaf Brazilian person, for example, who only knows Libras, to travel to another country with an entirely different Sign language. Considering this context, and using the techniques mentioned above, we propose a system to translate between Sign languages which can be used on a mobile phone.

In a world with almost 8 billion people and with more than 6.3 billion smartphones[1], mobile technology is part of our daily lives. With features such as a camera, internet, and an infinite amount of applications to be downloaded, the smartphone quickly became our day-to-day go-to gadget. However, even high-end smartphones do not have enough computing power to run deep learning algorithms, especially regarding Sign language recognition, which requires massive amounts of memory, storage, and GPU capability to train and test efficiently. Moreover, trying to do so would affect the smartphone's performance and battery life. Thus, we opted to leave the mobile application solely responsible for capturing video frames and presenting the translated signal due to computational time, power, and memory involved in using a Deep Learning model in a mobile device. Thus it leaves the Deep Learning part of our system running on the server-side.

In this way, our approach is composed of the following parts:

1. Mobile application for Sign language translation - Development of a mobile application integrated with a webserver to send images from recorded Sign language videos and display translated Sign language video to the user.

2. Background removal - Using Mask-RCNN trained on publicly available hand dataset as we want to remove unnecessary information for Sign language recognition by leaving just the hand in the images sent by the mobile application.

3. CNN Features - Use of Efficientnet CNN trained on our pre-processed dataset to extract features from learnt weights.

[1] $https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/$

4. Sign language Transformers - Adaptation of Transformers architecture in a Sign language context. Transformers input uses features extracted from a CNN neural network.

5. Webpage - Development of a web page to provide an intuitive visualization of different Sign languages, engage users to contribute in creating their native Sign language dataset, and provide new forms of user interaction when viewing a Sign language gesture.

## 5.5 PROPOSED TRANSLATION APPROACH

We use two Transformers to perform Sign language recognition and translation. The first Transformers recognize gloss sentences from video frames, while the second translates these gloss sentences to a destination language. Once translated, our system maps these translated glosses to gestures in the destination Sign language. Finally, the mobile application displays the target gestures to the user. To perform the translation between Sign languages, our architecture is comprised of 4 modules: 1) background removal using Mask-RCNN to leave just the hands in images captured from real environments; 2) feature extraction using Efficientnet; 3) recognition of Sign language gloss sentences using Transformers; and 4) a second Transformers to translate gloss sentences to a destination language. These modules are hosted on our server due to their computing power.

### 5.5.1 Background Removal Module

As previously explained, we trained a Mask R-CNN architecture on 15K annotated hand images from the EgoHands dataset. This module works when a user uploads an image, and the background removal module will automatically create a mask and save its mask coordinates in a JSON file to be used later to customize this mask. This mask is also used to create final images for future fine-tuning our network to create more precise masks. As we can see in Figure 66, our model was able to leave just the hand and remove everything else from the background. The customized mask functionality is better explained in Section 5.6.2.4.

Figure 66 – Top row images contains complex background with dynamic lighting and bottom row are images pre-processed using our background removal approach.

Source: Created by the author (2022)

### 5.5.2 Feature Extraction Module

For feature extraction we used Efficientnet CNN network. We chose this network due to its novel architecture and high results being achieved. Figure 67 shows our Efficientnet model to train and extract features from the PHOENIX-14T and V-Librasil dataset.

Figure 67 – Our efficient model to extract feature from V-Librasil. We replaced the last layers from the Efficientnet with three layers: GAP; dropout; and dense. Our features are the result from the GAP layer.

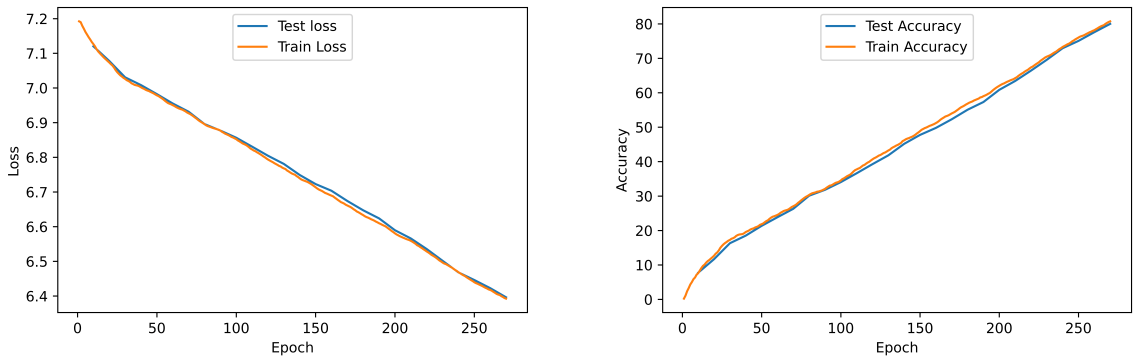
```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
efficientnet-b0 (Functional) (None, 10, 6, 1280)       4049564
_________________________________________________________________
global_average_pooling2d (Gl (None, 1280)              0
_________________________________________________________________
dropout (Dropout)            (None, 1280)              0
_________________________________________________________________
dense (Dense)                (None, 1329)              1702449
=================================================================
Total params: 5,752,013
Trainable params: 5,709,997
Non-trainable params: 42,016
```

Source: Created by the author (2022)

Our features were the result of the output of the GAP layer. This same model was used for

PHOENIX-14T training and feature extraction. We just changed the output parameter of the dense layer during training with the PHOENIX-14T. In both datasets, we used 80% for training and 20% for testing. In Figure 68 we can visualize our CNN model progression throughout approximately 250 epochs when training Efficientnet on V-Librasil pre-processed using our background removal approach. Our model ended up with 80% accuracy and decreased loss from 7.2 to 6.4.

Figure 68 – Loss and accuracy progression when training an EfficientNet with preprocessed V-Librasil dataset. a) represents training and test loss and b) represents training and test accuracy of our model.




(a) (b)

Source: Created by the author (2022)

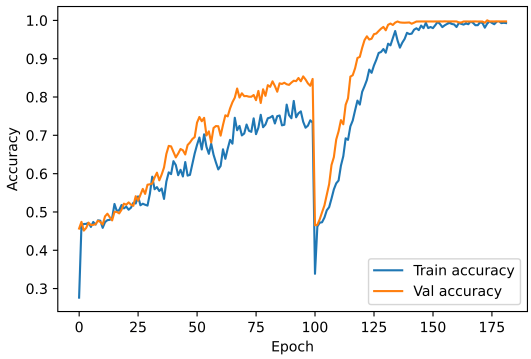
### 5.5.3 Recognition and Translation Module

We used two Transformers to recognize and translate gloss sentences in Sign languages. The first Transformers received inputs from a CNN feature extractor and outputs their corresponding gloss sentences. The second Transformers is responsible for translating from previously recognized glosses to destination glosses.

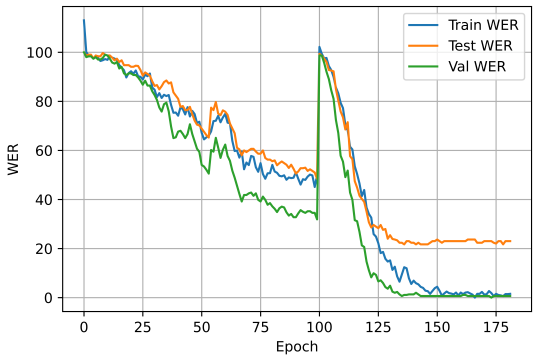
In Figure 69(a) and 69(c) we can accompany the accuracy and WER progression during approximately 180 epochs to check for over or under-fitting and to see if our model is generalizing well. Please note that the spike in epoch 100 in Figures 69(a) and (c) is due to training restart. In Figure 69(b) and 69(d) we can accompany the progression for sentences using our previously described V-Librasil sentence approach. Our Transformers for Sign language recognition consisted of 2 layers and 4 heads, the same used from (CAMGOZ et al., 2020b). As for the Transformers for language translation we used 4 layers with 8 heads as seen in (VASWANI et al., 2017a). We used the default values from Adam optimizer (KINGMA; BA, 2014) as our

optimization algorithm with a learning rate of 0.0001, beta 1 = 0.9, beta 2 = 0.98 and epsilon = 1e-9 (KINGMA; BA, 2014) where beta 1 is responsible for the exponential decay rate for the first moment estimates, beta 2 is responsible for the exponential decay rate for the second-moment estimates and the epsilon prevents any division by zero in the implementation.

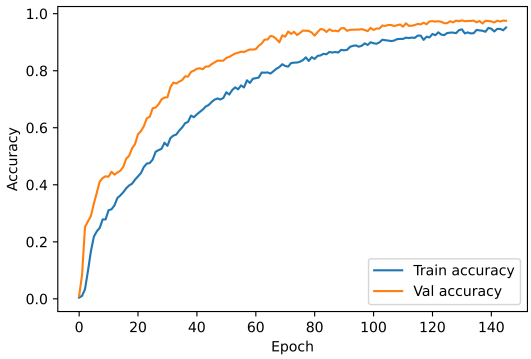
Figure 69 – Accuracy and WER progression when training on the V-Librasil dataset with all images pre-processed using our background removal. a) Represents training and validation accuracy of our model when training on V-Librasil words; b) Represents the training and validation accuracy of our model when training using our word-based V-librasil sentence dataset; c) Represents WER results for words; and d) Represents WER results for sentences.



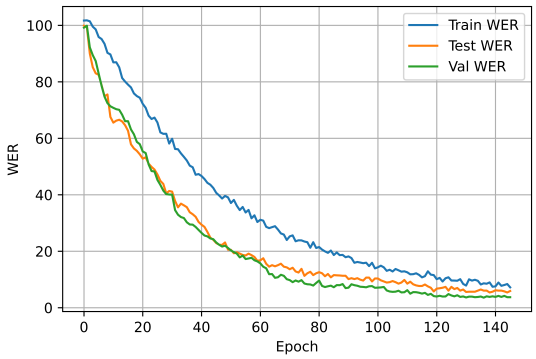

(a) Accuracy progression with words.




(c) WER progression with words.




(b) Accuracy progression with sentences.




(d) WER progression with sentences.

Source: Created by the author (2022)

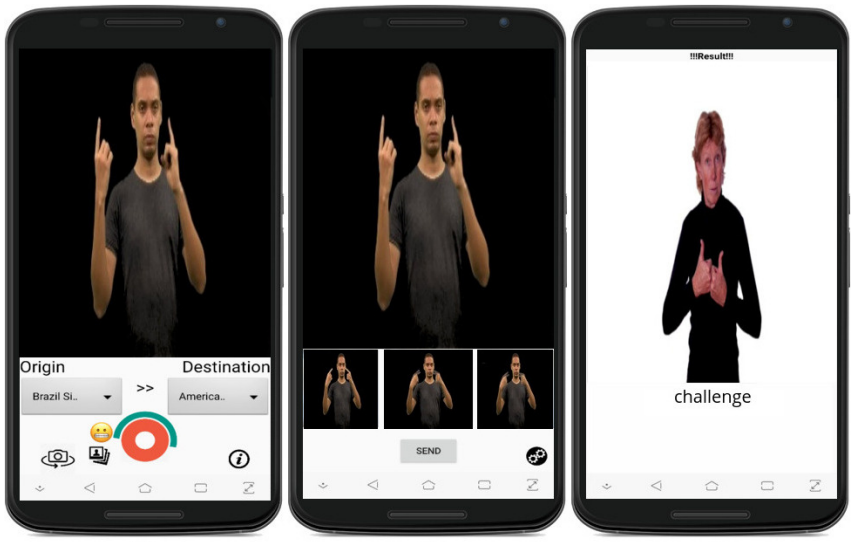
## 5.6 FUNCTIONALITIES AND USER INTERFACES

In this section we describe the main functionalities of our web and mobile application and detail them for better understanding why we developed them.

### 5.6.1 Mobile Application

Our mobile application is comprised of the following steps to perform translation. First, the user selects the origin and destination Sign languages (Figure 70(a)). Then the user presses the record button to capture the video of the communication partner performing a gesture. Once video capturing is done, the application displays to the user the recorded video as well as its frames (Figure 70(b)). The user sends the frames to the server for background removal, recognition, and translation. The translation is displayed in the destination Sign language along with the gloss sentence in the corresponding spoken language of the country (Figure 70(c)).

Figure 70 – Proposed mobile application: a) capturing video of the gesture "desafio"(challenge) in Libras, b) slicing into frames and c) displaying the translation in ASL.

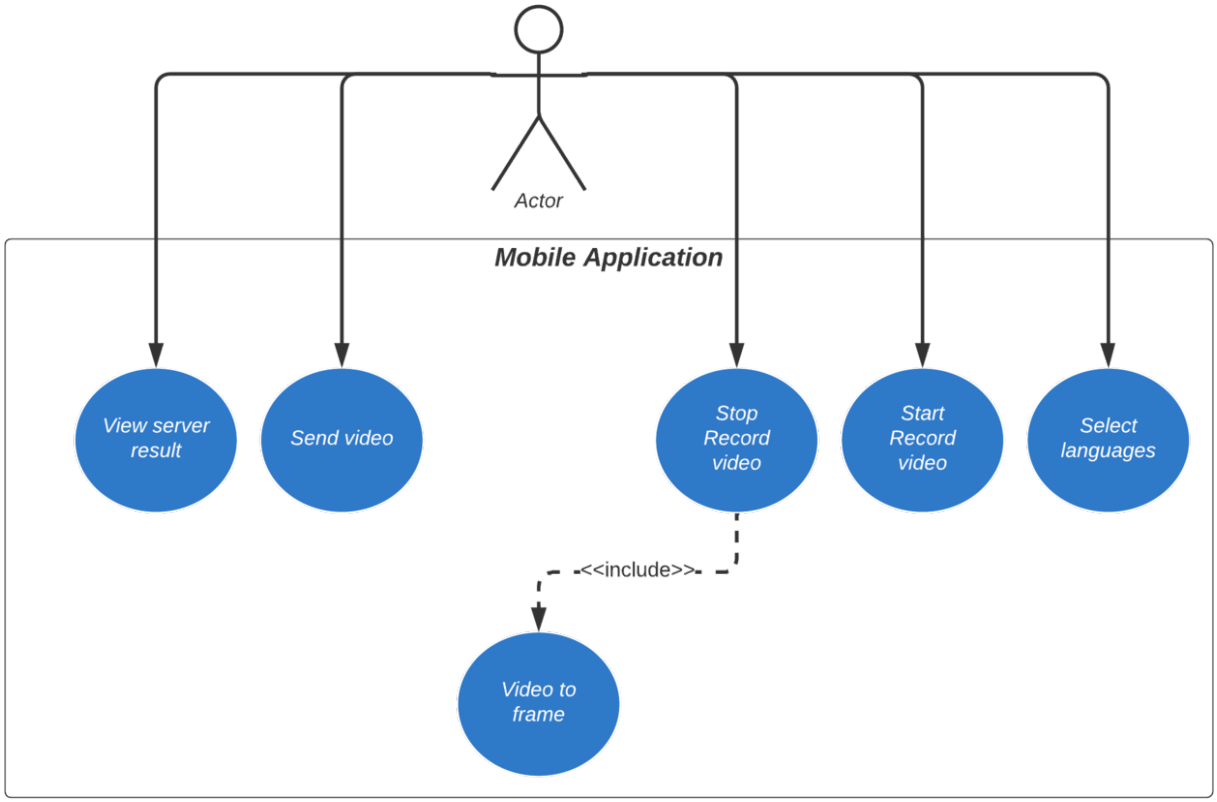



Source: Created by the author (2022)

In Figure 71 we show our mobile application UML use case diagram. As we can see, there are five different use cases with which a user can interact: Select language; Start record video; Stop record video; Send video; and View server result. All lines represent relationships. Solid arrows represent user interaction relationships, and dotted arrows represent either include or extend relationships. An include relationship is a dependency of another use case that will happen right after a base use case is called. Our mobile application presents a simple use case diagram and only includes a relationship. When the user interacts with "Stop record video"it automatically calls "Video to frame"to save the frames from the recorded video.

In Figure 72 displays the UML activity diagram of our mobile application. This diagram details our mobile application, server, and user behaviors. The black circle represents the

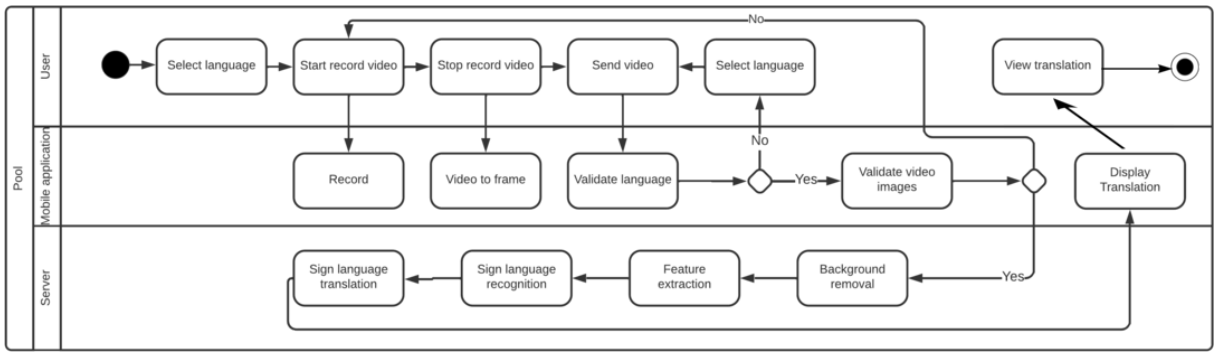
Figure 71 – UML use case diagram for our mobile application showing how a user can interact with our mobile application and the flow of each use case which is represented by blue circles.




Source: Created by the author (2022)

starting point, and it progresses through our system actions until it reaches the endpoint represented by the surrounded black circle.

Figure 72 – UML activity diagram for the mobile application. This diagram shows the behavior of our system when performing Sign language translation. It displays the starting point (black circle), actions (rectangles), and endpoint of our system (surrounded black circle).

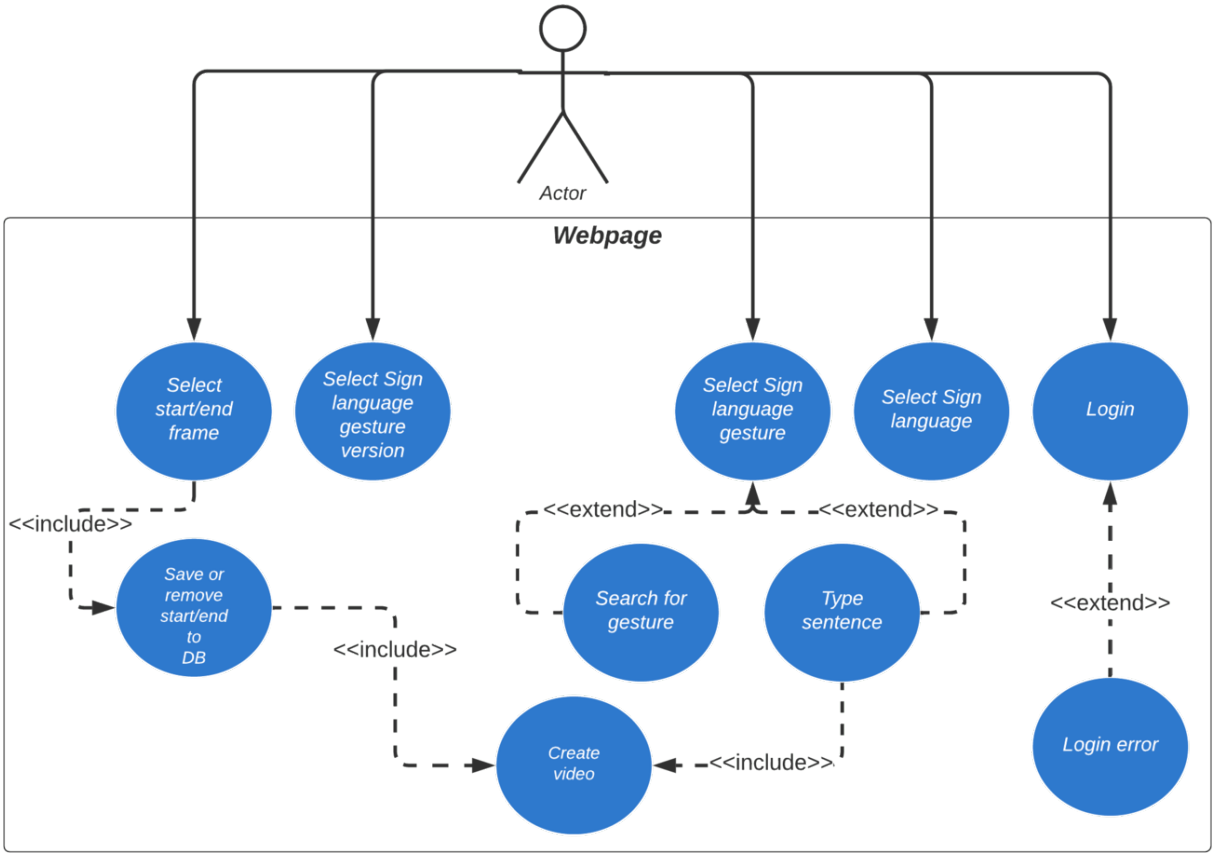



Source: Created by the author (2022)

### 5.6.2 Webpage

Our webpage is hosted through our server. The goal of the webpage is to provide a collaborative portal for building, expanding, and improving the quality of the database. The idea is to create an international community of people interested in globalizing communication for deaf people and learning Sign languages. On this page, the user can: 1) visualize all Sign languages (and the respective gestures) available in the system; 2) create new Sign languages and upload gestures; view the Sign language gestures for a typed gloss sentence; 3) access the videos of the gestures to mark the beginning and end of the gesture; and 4) annotate the hands in a particular image. Next, each of these functionalities is explained.

Figure 73 – UML use case diagram for our webpage showing how a user can interact with our webpage and the flow of each use case which is represented by blue circles.

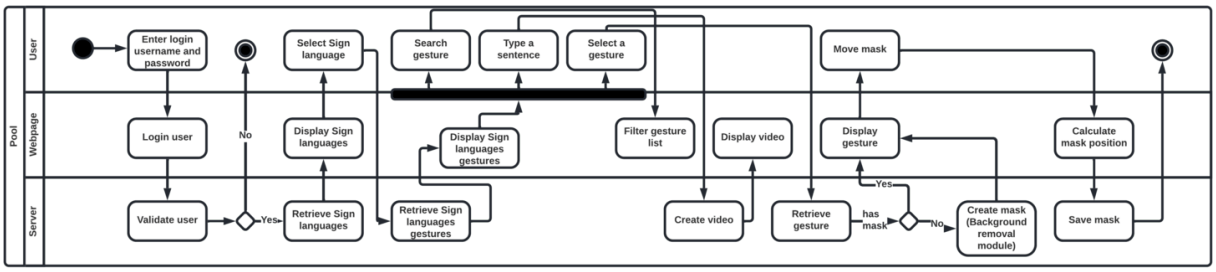



Source: Create by the author (2022)

In Figure 73 we show our webpage UML use case diagram. As we can see, there are five different use cases with which a user can interact: Login; Select Sign language; Select Sign language gesture; Select Sign language version; and Select start/end frame. It is slightly more complex than our mobile app UML use case diagram. It follows the same principle of lines

and their relationships, but we have an extended relationship in this diagram. This type of relationship means that a use case may or may not happen, and it is there to complement the calling use case. For example, when a user interacts with "Select Sign language gesture", the "Search for gesture"use case will only happen if the user searches for something. Therefore, it only complements the "Select Sign language gesture". This diagram also allows us to combine relationships, as seen in the use case "Select Sign language gesture"when a user types a sentence. The "Type sentence"use case may or may not happen, but if it happens, it will automatically call "Create video"via the include relationship. Another combination of relationships occurs when we have two include relationships. For example, when a user interacts with "Select start/end frame", the "Save or remove start/end to Database"use case will be automatically called, which will call another use case "Create video".

Figure 74 – UML activity diagram for our webpage. This diagram shows the behavior of our webpage with user interaction. It displays the starting point (black circle), actions (rectangles), and endpoint of our system (surrounded black circle).



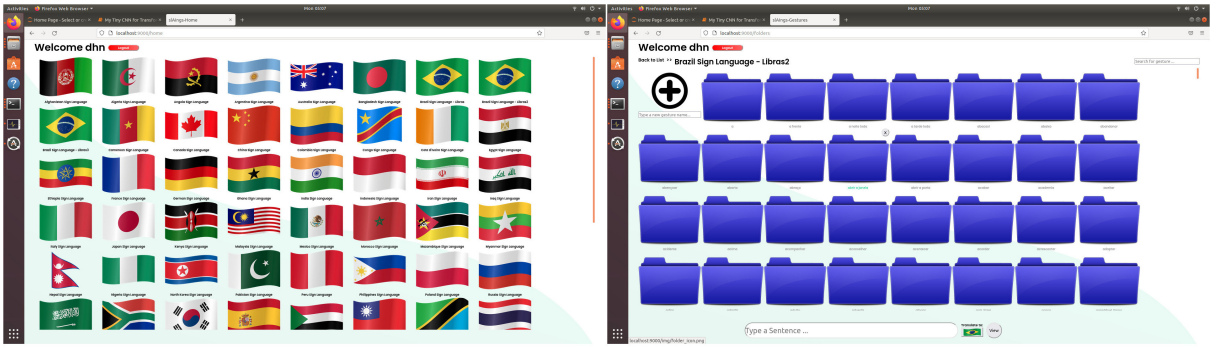

Source: Create by the author (2022)

In Figure 74 displays the UML activity diagram of our webpage application. This diagram details the behavior of our webpage and server with the user.

#### *5.6.2.1 Visualization and upload*

The user can create a new Sign language in the system and/or upload gestures. If the gesture is already registered, the system creates a new version to incorporate into the dataset, increasing our training set. Figure 75(a) displays the all Sign languages registered in our server and Figure 75(b), all gestures of a Sign language.

Figure 75 – Our webpage displaying Sign language and their gestures.



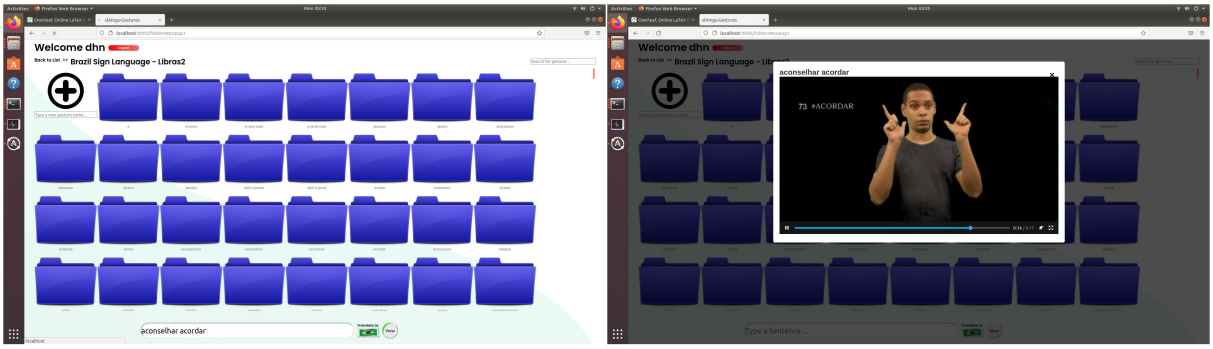

(a) (b)

Source: Created by the author (2022)

### 5.6.2.2 *Sentence view*

This functionality allows users to view a combination of gestures in Sign language from a typed gloss sentence. Figure 76(a) shows a text field for the user to type the words of a gloss sentence. Then, the system automatically creates a video from the typed words, and the result is displayed as shown in Figure 76(b). With this functionality, people can search for sentences in various Sign languages, helping them learn and communicate.

Figure 76 – Our webpage dynamically combines words and shows to the user their sentence in a video.




(a) (b)

Source: Created by the author (2022)

### 5.6.2.3 *Start and end point of a gesture*

Sign languages have a specific gesture to denote intervals between words or sentences, which can be called "rest gesture"(Figure 77). In datasets, every word or sentence begins and ends with the rest gesture. However, the rest gesture is not part of the primary gesture.

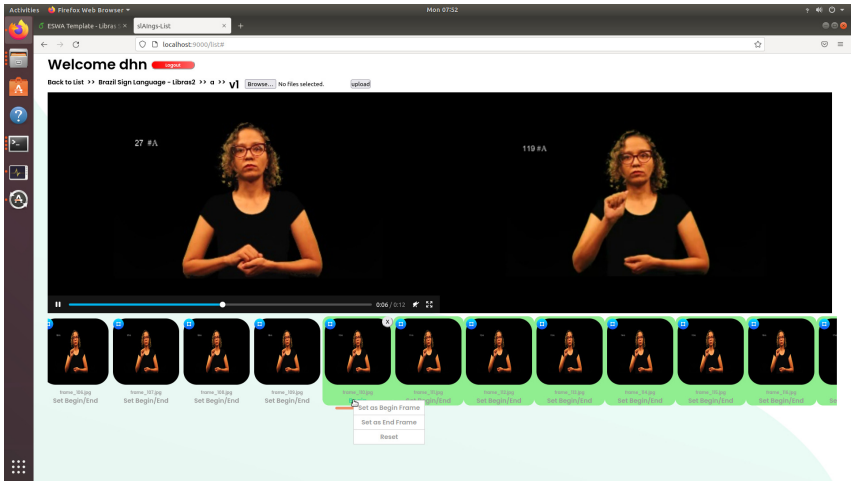
To facilitate the system's identification of the main gesture, the user can mark where the main gesture starts and ends in the uploaded video. This helps gesture recognition as the rest

Figure 77 – Rest gesture examples from V-Libras by (RODRIGUES, 2021).

Source: RODRIGUES (2021)

gesture is considered unwanted information. Figure 78 displays this functionality. The system also creates a video with the frames in between the start and end frames and displays it to the user for checking.

Figure 78 – User can set begin and end frames of a gesture.




Source: Created by the auhor (2022)

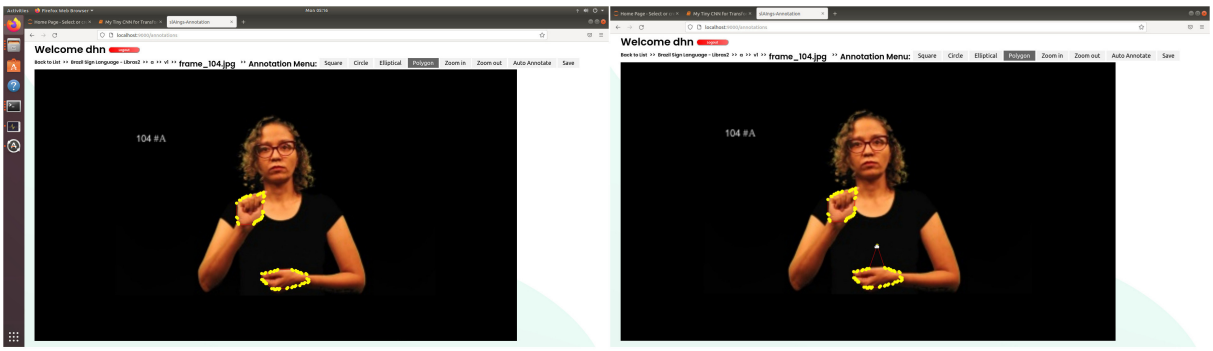
### *5.6.2.4 Gesture Annotation*

In this work, our features are based solely on the hands in an image. As previously explained, we used a Mask-RCNN to remove everything but the hand. However, creating a hand mask for every image can be very time-consuming. Thus, we added functionality to our website for hand annotation by the users.

Initially, images uploaded to our server are automatically pre-processed, meaning that

everything has everything but the hands removed. For example, in Figure 78, the blue icon indicates that this image has been pre-processed.

Figure 79 – Our webpage automatically creates a mask around the hands and lets the user customize this mask.




(a) (b)

Source: Created by the author (2022)

The user can click on any of these images to see the recognized hands with boundaries marked with yellow dots (Figure 79(a)). Yellow dots are obtained by detecting external contours from the Mask-RCNN mask image. We go through each outline and extract their x and y positions. These yellow dots are customizable through the user interface: if the dots do not mark the hands' contour precisely enough, the user can drag them anywhere in the image to adjust, as we can see in Figure 79(b). This helps us have more precise hand annotations leading to better Mask-RCNN models.

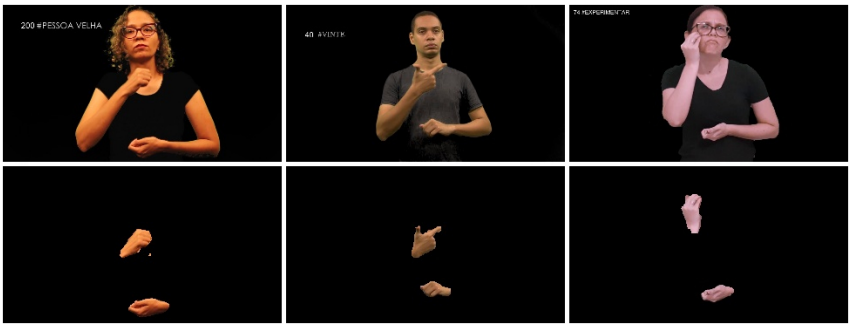
# 6 EXPERIMENTS AND RESULTS

This section presents our results obtained using RWTH-PHOENIX-WEATHER 2014T and the V-Librasil datasets. Our goal is to use Deep Learning to correctly recognize sign language gestures from a Sign Language video and translate them to another Sign language.

## 6.1 BACKGROUND REMOVAL RESULTS

As we can see in Figure 80 and Figure 81, our Mask-RCNN model was able to remove all irrelevant information from the images, leaving only the hands.

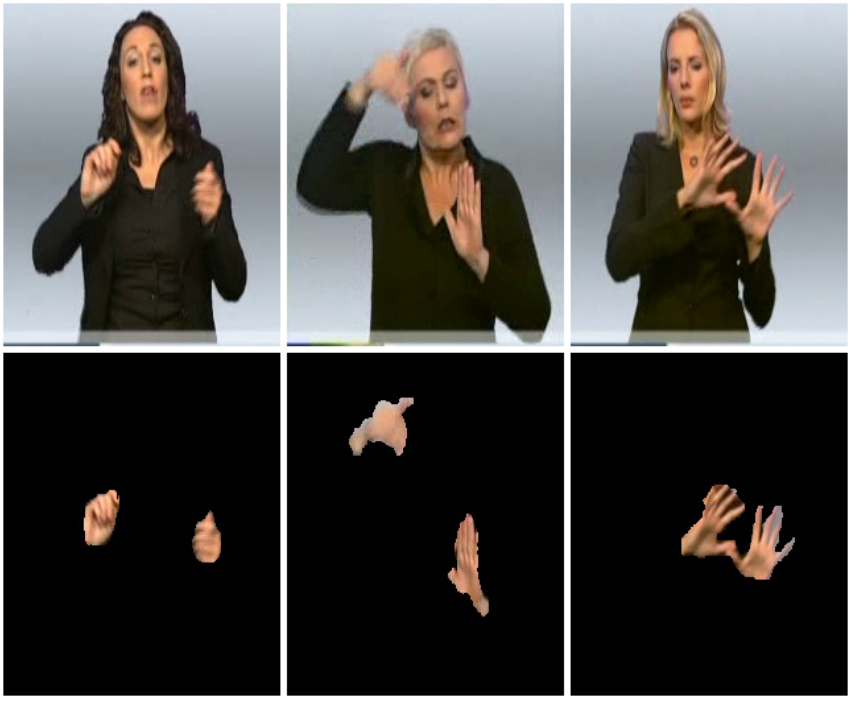
Figure 80 – Images from V-Librasil dataset (RODRIGUES, 2021) before and after background removal.




Source: Created by the author (2022)

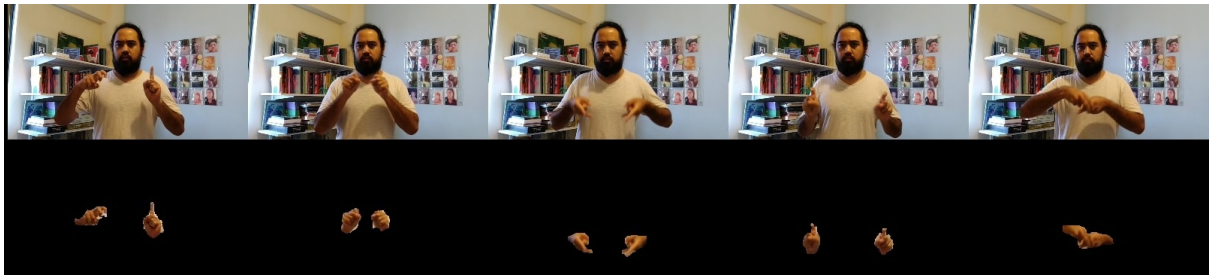
However, as stated previously, our system aims to process images from real scenarios, which typically have complex backgrounds. Therefore, although the publicly available datasets did not contain any complex background, we created images with complex backgrounds for 15 Libras sentences to investigate the performance. These samples allowed us to test our Mask-RCNN solution for the background removal model within a real-world scenario. Besides, we also tested our model performance in a user-independent recognition, i.e., gestures performed by a person that is not present in the training dataset. For these images with a complex background, our model achieved promising results, as we can see in Figure 82.

Figure 81 – Images from PHOENIX-14T dataset (CAMGOZ et al., 2020b) before and after background removal.



Source: Created by the author (2022)

Figure 82 – Sample images with complex background images before and after background removal.



Source: Created by the author (2022)

## 6.2 RECOGNITION RESULTS

We used two publicly available datasets to obtain our recognition results. The first dataset we used, called PHOENIX-14T from (CAMGOZ et al., 2020b) is a consolidated Sign language dataset in academia to recognize German Sign language. We used this dataset to compare state-of-the-art Sign language recognition with our approach. The other dataset is the V-Librasil dataset, a Brazilian Sign language dataset. We opted to use this dataset for two main reasons: the location of our work, which is set in Brazil, and a very recent dataset with more words. In addition, no other works have published comparable results with this dataset.

Table 10 – WER Progression using the entire PHOENIX-14T dataset from (CAMGOZ et al., 2020b). We achieved our lowest WER in epoch 83. Our model accuracy varied on the last epochs, indicating that it would no longer decrease its accuracy. Epoch 100 finished with 24.7% WER.

| Epoch | Train WER | Val WER | Test WER |
|---|---|---|---|
| 1 | 100.19 | 94.22 | 94.68 |
| 10 | 73.72 | 70.26 | 70.93 |
| 20 | 64.79 | 60.06 | 60.77 |
| 30 | 58.66 | 54.72 | 54.09 |
| 40 | 53.89 | 47.05 | 48.28 |
| 50 | 49.43 | 41.66 | 40.40 |
| 60 | 44.80 | 37.61 | 35.84 |
| 70 | 40.97 | 30.98 | 31.77 |
| 80 | 37.36 | 25.95 | 26.83 |
| **83** | **36.41** | **23.75** | **24.14** |
| 100 | 36.41 | 23.75 | 24.70 |

Source: Created by the author (2022)

### 6.2.1 Results with PHOENIX-14T

(CAMGOZ et al., 2020b) used Transformers on the PHOENIX-14T dataset and made their CNN features publicly available, allowing us to run our first test, which consisted of confirming that our code could achieve the same results. As we can see in Table 10, we were able to accomplish the proposed result of 24% WER in 83 epochs. We can see that our model is able to achieve a good generalization and improve its results quite significantly until epoch 83. From epoch 83 onward, our model could not decrease anymore its WER results and reached its plateau, meaning that our model learnt what it could from the dataset. Letting it run longer would not have any influence on the final result.

Our second test preprocessed the PHOENIX-14T dataset using our background removal method. Again, we used the same EfficientNet CNN architecture to extract relevant features. Since this dataset has more than 1.5M images, we decided to use only half due to computational memory capacity. With the dataset properly divided, we had approximately 4,200 videos for training, 400 videos for validation, and 500 for testing. Our results, using part of the dataset with the provided features from (CAMGOZ et al., 2020b), are displayed in Table 11(a). As we can see, we achieved a 27.07% WER in 100 epochs. Our model had 67 out of 341 sentences with 0% WER, meaning that it correctly recognized all words in 67 sentences.

We then trained our model with the same examples from the divided dataset but with

Table 11 – a) WER progression using half of the PHOENIX-14T dataset from (CAMGOZ et al., 2020b); b) WER progression with the same number of epochs while using the same half of PHOENIX-14T with pre-processed images using our background removal.

| (CAMGOZ et al., 2020b) Features | | | | Our approach | | | |
|---|---|---|---|---|---|---|---|
| Epoch | Train WER | Val WER | Test WER | Epoch | Train WER | Val WER | Test WER |
| 1 | 103.49 | 100 | 100 | 1 | 105.52 | 96.26 | 97.27 |
| 20 | 71.04 | 69.51 | 69.61 | 20 | 80.48 | 77.14 | 76.24 |
| 40 | 60.85 | 54.73 | 57.49 | 40 | 72.1 | 64.44 | 66.21 |
| 60 | 52.99 | 45.91 | 46.83 | 60 | 62.37 | 53.51 | 56.79 |
| 80 | 46.33 | 36.77 | 36.22 | 80 | 52.87 | 37.37 | 38.4 |
| 100 | 39.79 | 27.98 | 27.07 | 100 | 42.29 | 24.19 | **23.53** |
| (a) | | | | (b) | | | |

Source: Created by the author (2022)

our background removal approach. As we can see from Table 11(b), our approach achieved 23.5% WER within the same 100 epochs, resulting in approximately 4% WER decrease. Our approach correctly recognized 79 sentences with 0% WER out of 371, approximately a 19% increase.

This result shows that removing unwanted information from a gesture image such as the background could make the network learn what is important and achieve better results faster. For example, in Table 11 we can see that we achieved a 23.5% with the same number of epochs from before.

### 6.2.2 Results with V-Librasil

With our code validated and our background removal approach improving results from a consolidated dataset, we performed tests on the V-Librasil dataset. Our dataset contained approximately 3,750 training examples and 430 test examples.

As we can see in Table 12(a) without background removal, our model only achieved a 61.7% WER on the test set and was not able to generalize and learn enough to decrease its accuracy. This approach achieved a 36.7% accuracy. We then performed the same test but using our background removal approach. As we can see in Table 12(b), we achieved a lowest WER 21.7% in epoch 147 on the test set. By looking at the prediction results and comparing them with our labels (prediction vs. label), our model achieved approximately 81.4% accuracy.

Finally, we tested our model using sentences in Libras instead of individual words (as explained in Section 4). As we can see in Figure 13, we achieved a lowest WER accuracy of

Table 12 – a) WER progression using V-Librasil result from (RODRIGUES, 2021) without background removal; b) WER progression using V-Librasil with pre-processed images using our background removal.

| Transformers with V-Librasil without background removal | | | | Our approach | | | |
|---|---|---|---|---|---|---|---|
| Epoch | Train WER | Val WER | Test WER | Epoch | Train WER | Val WER | Test WER |
| 1 | 100.52 | 99.10 | 99.57 | 1 | 113 | 100 | 100 |
| 20 | 47.24 | 14.02 | 68.19 | 20 | 91.53 | 91.39 | 94.73 |
| 40 | 5.64 | 0.56 | 62.23 | 40 | 75.43 | 65.34 | 82.34 |
| 60 | 0.99 | 0.71 | 62.73 | 60 | 73.42 | 60.26 | 76.31 |
| **73** | **0.10** | **0.56** | **61.73** | 80 | 50.72 | 38.52 | 56.25 |
| 80 | 0.28 | 0.566 | 62.30 | 100 | 49.84 | 31.89 | 48.35 |
| 100 | 0.80 | 0.566 | 62.15 | 120 | 43.85 | 20.64 | 39.03 |
| 120 | 0.50 | 0.566 | 62 | 140 | 5.57 | 1.32 | 22.36 |
| 140 | 0.48 | 0.56 | 62.19 | **147** | **2.63** | **0.66** | **21.71** |
| 150 | 0.45 | 0.56 | 61.92 | 150 | 1.44 | 0.66 | 23.03 |
| (a) | | | | (b) | | | |

Fonte: Created by the author

Table 13 – Results obtained from generated Libras sentences.

| Epoch | Train WER | Val WER | Test WER |
|---|---|---|---|
| 1 | 101.71 | 99.18 | 99.89 |
| 20 | 74.36 | 57.91 | 54.32 |
| 40 | 47.32 | 27.53 | 30.36 |
| 60 | 30.34 | 16.39 | 16.61 |
| 80 | 21.14 | 8.75 | 12.18 |
| 100 | 13.92 | 7.04 | 10.33 |
| 120 | 11.53 | 4.1 | 5.7 |
| **140** | **8.94** | **3.81** | **5.39** |
| 150 | 7.19 | 3.71 | 5.91 |

Source: Created by the author (2022)

5.39% on the test dataset on epoch 140. This result means that our model correctly classified 53 out of 68 gloss sentences, resulting in approximately 78% accuracy.

Our user-independent test (i.e. using novel data from users not included in V-Librasil dataset) achieved 73% accuracy correctly recognizing 11 out of 15 sentences (Table 14). In WER accuracy, it achieved 5.8%. From the 15 sentences we tested (shown in Table 14), our model wrongly recognized the words "capacete"(helmet) as "calouro"(freshman) or "carpinteiro"(carpenter) in sentence #1; in sentence #8 "cancelar"(cancel) as "demitir"(fire); in sentence #10, "computador"(computer) was recognized as "devemos"(we must); and in sentence

Table 14 – Experiment with 15 sentences for our user-independent with complex background test with their English translation from Portuguese. Words in bold are incorrect recognition results.

| | Input | Recognition | Translation |
|---|---|---|---|
| 1 | CAPACETE CONFORTAVEL<br>(helmet comfortable) | CALOURO CONFORTAVEL<br>(**freshman**comfortable)<br><br>CARPINTEIRO CONFORTAVEL<br>(**carpenter** comfortable ) | **FRESHMEN** COMFORTABLE;<br><br>**CARPENTER** COMFORTABLE |
| 2 | DESAFIO AGORA ABRIR A PORTA<br>(Challenge now open the door) | DESAFIO AGORA ABRIR A PORTA<br>(Challenge now open the door) | CHALLENGE NOW OPEN THE DOOR |
| 3 | AUTORIDADE AGIR BEM<br>(Authority act well) | AUTORIDADE AGIR BEM<br>(Authority act well) | AUTHORITY ACT WELL |
| 4 | ADULTO ACEITAR DESAFIO DIFICIL<br>(Adult accept difficult challenge) | ADULTO ACEITAR DESAFIO DIFICIL<br>(Adult accept difficult challenge) | ADULT ACCEPT DIFFICULT CHALLENGE |
| 5 | CHEFE AVALIAR CULTURA DEPARTAMENTO DE NOVO<br>(Boss evaluate culture department again) | CHEFE AVALIAR CULTURA DEPARTAMENTO DE NOVO<br>(Boss evaluate culture department again) | BOSS EVALUATE DEPARTMENT CULTURE AGAIN |
| 6 | BOMBA DESTRUIR CIDADE ACIDENTE<br>(Bomb destroy city accident) | BOMBA DESTRUIR CIDADE ACIDENTE<br>(Bomb destroy city accident) | BOMB DESTROY CITY ACCIDENT |
| 7 | CALOURO CONFRONTAR DESAFIO<br>(Freshman confront challenge) | CALOURO CONFRONTAR DESAFIO<br>(Freshman confront challenge) | FRESHMEN CONFRONT CHALLENGE |
| 8 | AUTORIDADE CANCELAR BENEFICIO<br>(Authority cancel benefit) | AUTORIDADE **DEMITIR** BENEFICIO<br>(Authority fired benefit) | AUTHORITY **RESIGN** BENEFIT |
| 9 | AR ACIMA AGRADAVEL<br>(Air above pleasent) | AR ACIMA AGRADAVEL<br>(Air above pleasent) | AIR ABOVE PLEASANT |
| 10 | COMPUTADOR DESCONECTAR<br>(Computer disconnect) | **DEVEMOS** DESCONECTAR<br>(we must disconnect) | **WE MUST** DISCONNECT |
| 11 | CHEFE CULPAR AUTORIDADE<br>(Boss blame authority) | CHEFE CULPAR AUTORIDADE<br>(Boss blame authority) | BOSS BLAME AUTHORITY |
| 12 | ASNO AGRESSIVO BRIGAR COM FREQUENCIA<br>(Donkey aggressive fight frequently) | ASNO AGRESSIVO BRIGAR COM FREQUENCIA<br>(Donkey aggressive fight frequently) | DONKEY AGGRESSIVE FIGHT FREQUENTLY |
| 13 | BISCOITO CAIR DE NOVO<br>(Cookie fall again) | **AVO** CAIR DE NOVO<br>(grandfather fall again) | **GRANDFATHER** FALL AGAIN |
| 14 | BOMBEIRO ACEITAR ACOMPANHAR CHEFE<br>(Fireman accept accompany boss) | BOMBEIRO ACEITAR ACOMPANHAR CHEFE<br>(Fireman accept accompany boss) | FIREMAN ACCEPT TO ACCOMPANY BOSS |
| 15 | DOENTE DESASSISTIDO DESISTIR<br>(Unattended sick give up) | DOENTE DESASSISTIDO DESISTIR<br>(Unattended sick give up) | SICK UNATTENDED GIVE UP |

Source: Created by the author (2022)

#13, "biscoito"(cookie) as "avo"(grandfather). All other sentences were correctly recognized.

## 6.3 TRANSLATION RESULTS

We mentioned previously that our translation module uses the gloss sentence output and translates to a destination language. Our proposed approach maps the translation to a destination Sign language gesture and sends it to our mobile application. To accomplish this, we trained a Transformers model with a Portuguese-English translation dataset from (TIEDEMANN, 2020) and fine-tuned it with a word-based translation. As we can see in Table 14, the wrong translations were consequences of wrong recognitions (Sentences #1, #8, #10 and #13). There were no translation errors per se. Thus, our translation result was similar to the recognition result, achieving 15% WER, which resulted in 73% accuracy, correctly translating 11 out of 15 sentences.

## 6.4 DISCUSSION

Our results show that our approach of removing the background and leaving just the hands in the image improved the accuracy of our model from 27.07% to 23.5% WER when using the

Table 15 – Overview of our results with and without our background removal.

| | WER | | Accuracy (%) | |
|---|---|---|---|---|
| | Phonenix-14T | V-Librasil | Phonenix-14T | V-Librasil |
| Without background removal | 27.07 | 61.73 | 67 | 36 |
| Our approach | **23.53** | **21.71** | **79** | **81.4** |

Source: Created by the author (2022)

PHOENIX-14T. Even though we used half videos of this dataset, we believe that our results while using the entire dataset would have the same behavior because only the hands were left in the images.

With the V-Librasil dataset without background removal, the best WER achieved by our model was 61.7%. We believe this happened because V-Librasil is a small dataset (3 examples per gesture) and contains unwanted information (resulting in unwanted features) in the images. Our model was unable to generalize enough to decrease its accuracy. Using the approach to remove everything but the hands, our model achieved 21.7% WER. This approach made our model focus on only what is important, the hands. Table 15 displays our results with and without background removal.

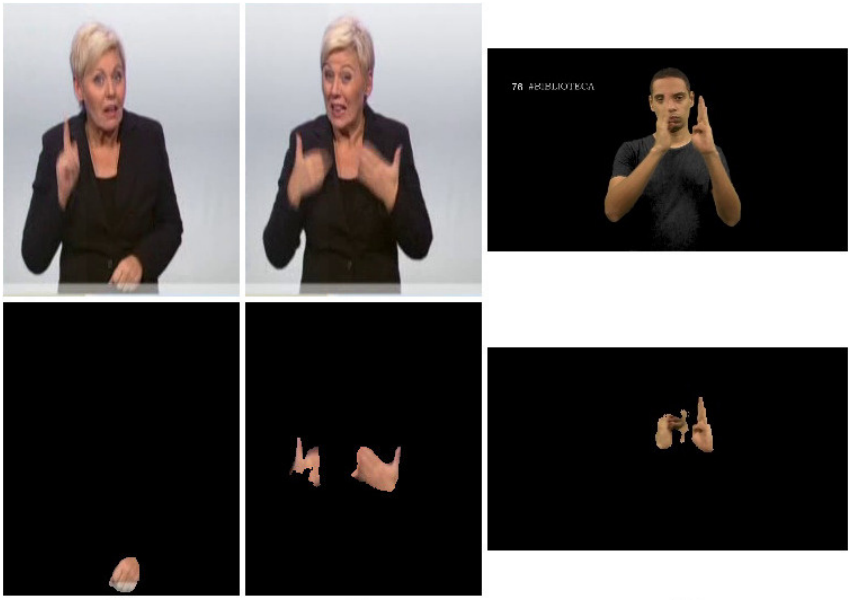
Our sentence-based dataset had 68 sentences. Achieving a 5% WER (78% accuracy) with sentences with hundreds of frames is very interesting. The ability of our model to recognize Libras sentences while having a word-based dataset sentence is promising. We believe that even by increasing the number of frames and words involved, our approach of leaving just the hands in the images made our model obtain a good generalization ability of the dataset. As for the translation mechanism, our translation module correctly translated our gloss sentences.

Nevertheless, our approach presented problems yet to be overcome. Although our background removal presented interesting results, some images were not perfectly processed. As we can see in Figure 83, our Mask-RCNN model was not able to properly leave the hand in the image, sometimes it removed more than what was needed (Figure 83(a) and Figure 83(b)) and sometimes it removed less, leaving unwanted parts of the body (Figure 83(c)).

One possible solution is to fine-tune Mask-RCNN with new hand annotations. This approach would make background removal more precise and increase our recognition results.

Another problem we faced was overfitting. While using the V-Librasil dataset, we noticed that the more we increased our model architecture, the higher the tendency to overfit. Since Transformers is a robust architecture, having a small dataset with many parameters makes our

Figure 83 – Unwanted results from our Mask-RCNN background removal. a) Only displaying one hand; b) Displaying half of one hand; c) Displaying half of one hand and part of the face.




Source: Create by the author (2022)

model prone to overfit. One possible solution is to increase our dataset with more examples or use Self-Supervised Learning (HENDRYCKS et al., 2019).

Finding a translation dataset for our needs (word-based) was not easy. Although the NLP area has a lot of consolidated translation datasets, most of them are sentence-based. Since we are dealing with translations of different languages, creating such a dataset can be an ordeal. To overcome this issue, a sentence-based translation dataset can be transformed to word-based, although human intervention would be needed to have a more accurate dataset.

# 7 CONCLUSION

In this work, we proposed a system comprised of a mobile application integrated with a server capable of translating between Sign languages. In addition, we created a web page to easily display different Sign languages to visualize individual gestures or sentences, set start and endpoints of gestures, and create custom hand masks. The main academic contribution of our webpage involves the Interactive AI domain. The ability of creating an initial hand mask and letting the user customize this mask creates a flow of interaction that provides more precise masks for fine-tunning a Mask R-CNN.

Concerning our first research question (Is it possible to perform translation between sign languages using a deep learning NLP architecture?), our approach involved four deep learning architectures: Mask-RCNN to leave only the hand in the image; EfficientNet to extract features; Transformers to perform gesture recognition; and another Transformers to perform translation. This approach improved WER accuracy when compared to other works. For example, our approach of leaving just the hand in the image achieved a better WER result (27% against 23%) if using the entire image and performed well under user-independent tests. This means that more people could have their gestures recognized and translated while having less diversity in our dataset. Furthermore, our approach produces more stable background removal results than pose estimation approaches and this approach of leaving just the hand in the image was not seen throughout the literature. Moreover, our solution can be used in budget-friendly to high-end smartphones since all computational calculations are done on a server. The adaptation of a seq2seq Transformers architecture for image sequence, specifically Sign language recognition, not only performed well on Sign language words but on Sign language sentences as well.

Regarding our second research question (Can a word-based sign language dataset be used to recognize sign language sentences?), we recognized Sign language video sentences by training a Transformer model on a word-based dataset. This approach showed that we could have more flexibility when creating sentences by combining words. We can also spend more time having individual Sign language words instead of complete sentences.

Our work tackles a problem not seen in our literature review that is Sign language translation. By performing this type of translation, we enable the deaf community to seize information from around the world. We involved more major research areas in Sign language recognition when comparing with our literature review which we believe is a good starting point to cre-

ate a more complete Sign language recognition solution. We also believe that our work when comparing with other works that use special hardware is a lot more inclusive since the only hardware needed is a smartphone. And lastly we did not see works worried about centralizing dataset so that other people can easily view. We believe that this is important to spread Sign language knowledge worldwide.

We believe that our work has a great potential to increase the social inclusion of people with hearing disabilities into a speaking society and sparkle curiosity about Sign languages worldwide.

## 7.1 CONTRIBUTIONS

Our specific contributions in this work are:

1. Development of a mobile application integrated with a webserver to send images from recorded Sign language videos and display translated Sign language videos to the user.
2. Development of a web page to provide an intuitive visualization of different Sign languages, engage users to contribute in creating their native Sign language dataset, and provide new forms of user interaction when viewing a Sign language gesture.
3. Background removal approach focused on Sign language recognition, using Mask-RCNN trained on publicly available hand dataset.
4. Adaptation of a seq2seq Transformers architecture to receive image inputs in order to translate between Sign languages.
5. Creation of a sentence dataset based on words to be used in continuous Sign language recognition.

We produced three papers from this work:

1. Neiva, D. H., & Zanchettin, C. (2022). Translating between Sign Languages using Transformers. Expert Systems with Applications. Under review.
2. Neiva, D. H., & Zanchettin, C. (2018). Gesture recognition: A review focusing on sign language in a mobile context. Expert Systems with Applications, 103, 159-183 (NEIVA; ZANCHETTIN, 2018).

3. Neiva, D. H., & Zanchettin, C. (2016). A Dynamic Gesture Recognition System to Translate between Sign Languages in Complex Backgrounds. In 2016 5th Brazilian Conference on Intelligent Systems (BRACIS) (pp. 421-426). IEEE. (NEIVA; ZANCHETTIN, 2016).

## 7.2 LIMITATIONS AND FUTURE WORK

Our system has some limitations. We briefly mentioned that the grammar structure in Sign languages differs from spoken ones. At present, our system only presents gloss sentences along with gestures. As future work, a model could be created to translate Sign language grammar to spoken language grammar and vice-versa, improving the translation to spoken language. On the same note, training our translation model would be very interesting. We can use a parallelized translation datasets from (TIEDEMANN, 2020) to train different translation models. Although we tackled Libras and ASL our architecture was thought to be used with different Sign languages. The inclusion of a new Sign language would follow the same steps of training a CNN and a Transformers with the repective dataset.

Another important future work is the increase in the size of the dataset. Deep Learning models require a considerable amount of data to achieve state-of-the-art results. However, we believe that dataset creation will become easier with our integrated system since it will be accessible to anyone, requiring a smartphone with an internet connection. It would be interesting to use Generative Adversarial Network (GAN) to increase our dataset.

And last but not least, facial expressions play an important role in Sign language communication, complementing the movement of the hands. Facial expressions can convey anger, happiness, doubt, danger, sadness, and so much more. In this work, we did not include facial expressions (in line with (CAMGOZ et al., 2020b; YIN; READ, 2020; KO; SON; JUNG, 2018b; KO et al., 2019b; COSTER; HERREWEGHE; DAMBRE, 2020b)), focusing on recognizing signs made with the hands due to its already high complexity involving a time-series problem and the removal of unwanted information regarding a gesture such as the complex background. Our decision not to include facial expressions also relies on the fact that the publicly available datasets we used only contain one facial expression (neutral). (KUZNETSOVA et al., 2021) show the complex nature of facial expressions. To include them in our recognition module, it would be interesting to have another network capable of analyzing facial expressions and adding to the outcome of our network.

# REFERENCES

AHMED, N.; NATARAJAN, T.; RAO, K. R. Discrete cosine transform. *IEEE transactions on Computers*, IEEE, v. 100, n. 1, p. 90–93, 1974.

AKANSU, A. N.; HADDAD, R. A. *Multiresolution signal decomposition: transforms, subbands, and wavelets*. [S.l.]: Academic Press, 2001.

AL-ROUSAN, M.; ASSALEH, K.; TALA'A, A. Video-based signer-independent arabic sign language recognition using hidden markov models. *Applied Soft Computing*, Elsevier, v. 9, n. 3, p. 990–999, 2009.

ALBAWI, S.; MOHAMMED, T. A.; AL-ZAWI, S. Understanding of a convolutional neural network. In: IEEE. *2017 international conference on engineering and technology (ICET)*. [S.l.], 2017. p. 1–6.

ALMEIDA, S. G. M.; GUIMARÃES, F. G.; RAMÍREZ, J. A. Feature extraction in brazilian sign language recognition based on phonological structure and using rgb-d sensors. *Expert Systems with Applications*, Elsevier, v. 41, n. 16, p. 7259–7271, 2014.

ALTMAN, N. S. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, Taylor & Francis, v. 46, n. 3, p. 175–185, 1992.

BAJPAI, D.; POROV, U.; SRIVASTAV, G.; SACHAN, N. Two way wireless data communication and american sign language translator glove for images text and speech display on mobile phone. In: IEEE. *Communication Systems and Network Technologies (CSNT), 2015 Fifth International Conference on*. [S.l.], 2015. p. 578–585.

BAMBACH, S.; LEE, S.; CRANDALL, D. J.; YU, C. Lending a hand: Detecting hands and recognizing activities in complex egocentric interactions. In: *The IEEE International Conference on Computer Vision (ICCV)*. [S.l.: s.n.], 2015.

BARCZAK, A.; REYES, N.; ABASTILLAS, M.; PICCIO, A.; SUSNJAK, T. A new 2d static hand gesture colour image dataset for asl gestures. *Research Letters in the Information and Mathematical Sciences*, Massey University, v. 15, p. 12–20, 2011.

BARROS, P.; MACIEL-JUNIOR, N. T.; FERNANDES, B. J.; BEZERRA, B. L.; FERNANDES, S. M. A dynamic gesture recognition and prediction system using the convexity approach. *Computer Vision and Image Understanding*, Elsevier, v. 155, p. 139–149, 2017.

BARROS, P. V.; JUNIOR, N. T.; BISNETO, J. M.; FERNANDES, B. J.; BEZERRA, B. L.; FERNANDES, S. M. Convexity local contour sequences for gesture recognition. In: *Proceedings of the 28th Annual ACM Symposium on Applied Computing*. [S.l.: s.n.], 2013. p. 34–39.

BASNIN, N.; NAHAR, L.; HOSSAIN, M. S. An integrated cnn-lstm model for bangla lexical sign language recognition. In: SPRINGER. *Proceedings of International Conference on Trends in Computational and Cognitive Engineering*. [S.l.], 2021. p. 695–707.

BAUM, L. E.; PETRIE, T. Statistical inference for probabilistic functions of finite state markov chains. *The annals of mathematical statistics*, JSTOR, v. 37, n. 6, p. 1554–1563, 1966.

BAY, H.; ESS, A.; TUYTELAARS, T.; GOOL, L. V. Speeded-up robust features (surf). *Computer vision and image understanding*, Elsevier, v. 110, n. 3, p. 346–359, 2008.

BECKER, S.; CRANDALL, M.; COWARD, C.; SEARS, R.; CARLEE, R.; HASBARGEN, K.; BALL, M. A. *Building digital communities: a framework for action*. [S.l.], 2012.

BEN-GAL, I. Bayesian networks. *Encyclopedia of statistics in quality and reliability*, Wiley Online Library, 2007.

BENGIO, Y. et al. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, Now Publishers, Inc., v. 2, n. 1, p. 1–127, 2009.

BHUVAN, M. S.; RAO, D. V.; JAIN, S.; ASHWIN, T.; GUDDETTI, R. M. R.; KULGOD, S. P. Detection and analysis model for grammatical facial expressions in sign language. In: IEEE. *Region 10 Symposium (TENSYMP), 2016 IEEE*. [S.l.], 2016. p. 155–160.

BREIMAN, L. Bagging predictors. *Machine learning*, Springer, v. 24, n. 2, p. 123–140, 1996.

BREU, H.; GIL, J.; KIRKPATRICK, D.; WERMAN, M. Linear time euclidean distance transform algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE, v. 17, n. 5, p. 529–533, 1995.

BROOMHEAD, D. S.; LOWE, D. *Radial basis functions, multi-variable functional interpolation and adaptive networks*. [S.l.], 1988.

CAMGOZ, N. C.; KOLLER, O.; HADFIELD, S.; BOWDEN, R. Sign language transformers: Joint end-to-end sign language recognition and translation. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. [S.l.: s.n.], 2020. p. 10023–10033.

CAMGOZ, N. C.; KOLLER, O.; HADFIELD, S.; BOWDEN, R. Sign language transformers: Joint end-to-end sign language recognition and translation. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. [S.l.: s.n.], 2020. p. 10023–10033.

CANNY, J. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, Ieee, n. 6, p. 679–698, 1986.

Cao, Z.; Hidalgo Martinez, G.; Simon, T.; Wei, S.; Sheikh, Y. A. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

CARIDAKIS, G.; KARPOUZIS, K.; DROSOPOULOS, A.; KOLLIAS, S. Somm: Self organizing markov map for gesture recognition. *Pattern Recognition Letters*, Elsevier, v. 31, n. 1, p. 52–59, 2010.

CELEBI, S.; AYDIN, A. S.; TEMIZ, T. T.; ARICI, T. Gesture recognition using skeleton data with weighted dynamic time warping. In: *VISAPP (1)*. [S.l.: s.n.], 2013. p. 620–625.

CHANG, C.-C.; LIN, C.-J. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, ACM, v. 2, n. 3, p. 27, 2011.

CHERNYKH, V.; PRIKHODKO, P. Emotion recognition from speech with recurrent neural networks. *arXiv preprint arXiv:1701.08071*, 2017.

CHO, K.; MERRIËNBOER, B. V.; GULCEHRE, C.; BAHDANAU, D.; BOUGARES, F.; SCHWENK, H.; BENGIO, Y. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

CHUA, L. Cellular neural networks: Theory. *IEEE Trans. CAS*, v. 32, n. 10, p. 559–577, 1996.

CIORBA, A.; BIANCHINI, C.; PELUCCHI, S.; PASTORE, A. The impact of hearing loss on the quality of life of elderly adults. *Clinical interventions in aging*, Dove Press, v. 7, p. 159, 2012.

CORTES, C.; VAPNIK, V. Support-vector networks. *Machine learning*, Springer, v. 20, n. 3, p. 273–297, 1995.

COSTER, M. D.; HERREWEGHE, M. V.; DAMBRE, J. Sign language recognition with transformer networks. In: *12th International Conference on Language Resources and Evaluation*. [S.l.: s.n.], 2020.

COSTER, M. D.; HERREWEGHE, M. V.; DAMBRE, J. Sign language recognition with transformer networks. In: EUROPEAN LANGUAGE RESOURCES ASSOCIATION (ELRA). *12th International Conference on Language Resources and Evaluation*. [S.l.], 2020. p. 6018–6024.

CUI, R.; LIU, H.; ZHANG, C. Recurrent convolutional neural networks for continuous sign language recognition by staged optimization. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 2017. p. 7361–7369.

DEVI, S.; DEB, S. Low cost tangible glove for translating sign gestures to speech and text in hindi language. In: IEEE. *Computational Intelligence & Communication Technology (CICT), 2017 3rd International Conference on*. [S.l.], 2017. p. 1–5.

DEY, R.; SALEM, F. M. Gate-variants of gated recurrent unit (gru) neural networks. In: IEEE. *2017 IEEE 60th international midwest symposium on circuits and systems (MWSCAS)*. [S.l.], 2017. p. 1597–1600.

DOMINGO, A.; AKMELIAWATI, R.; CHOW, K. Y. Pattern matching for automatic sign language translation system using labview. In: IEEE. *Intelligent and Advanced Systems, 2007. ICIAS 2007. International Conference on*. [S.l.], 2007. p. 660–665.

DONG, M. Convolutional neural network achieves human-level accuracy in music genre classification. *arXiv preprint arXiv:1802.09697*, 2018.

DOSOVITSKIY, A.; BEYER, L.; KOLESNIKOV, A.; WEISSENBORN, D.; ZHAI, X.; UNTERTHINER, T.; DEHGHANI, M.; MINDERER, M.; HEIGOLD, G.; GELLY, S. et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

DUDA, R. O.; HART, P. E. Use of the hough transformation to detect lines and curves in pictures. *Communications of the ACM*, ACM, v. 15, n. 1, p. 11–15, 1972.

ELLIOTT, D. L. *A better activation function for artificial neural networks*. [S.l.], 1993.

ELONS, A.; AHMED, M.; SHEDID, H. Facial expressions recognition for arabic sign language translation. In: IEEE. *Computer Engineering & Systems (ICCES), 2014 9th International Conference on*. [S.l.], 2014. p. 330–335.

ESCALERA, S.; BARÓ, X.; GONZALEZ, J.; BAUTISTA, M. Á.; MADADI, M.; REYES, M.; PONCE-LÓPEZ, V.; ESCALANTE, H. J.; SHOTTON, J.; GUYON, I. Chalearn looking at people challenge 2014: Dataset and results. In: *ECCV Workshops (1)*. [S.l.: s.n.], 2014. p. 459–473.

FREUND, Y.; SCHAPIRE, R. E. A desicion-theoretic generalization of on-line learning and an application to boosting. In: SPRINGER. *European conference on computational learning theory*. [S.l.], 1995. p. 23–37.

FRIEDMAN, J.; HASTIE, T.; TIBSHIRANI, R. et al. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The annals of statistics*, Institute of Mathematical Statistics, v. 28, n. 2, p. 337–407, 2000.

GALLOWAY, M. M. Texture analysis using gray level run lengths. *Computer graphics and image processing*, Elsevier, v. 4, n. 2, p. 172–179, 1975.

GIDUMAL, J. B. Impact of artificial intelligence in travel, tourism and hospitality. Springer, 2020.

GLOROT, X.; BENGIO, Y. Understanding the difficulty of training deep feedforward neural networks. In: JMLR WORKSHOP AND CONFERENCE PROCEEDINGS. *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. [S.l.], 2010. p. 249–256.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep learning*. [S.l.]: MIT press, 2016.

GRAF, I.; KRESSEL, U.; FRANKE, J. Polynomial classifiers and support vector machines. *Artificial Neural Networks—ICANN'97*, Springer, p. 397–402, 1997.

HAKKUN, R. Y.; BAHARUDDIN, A. et al. Sign language learning based on android for deaf and speech impaired people. In: IEEE. *Electronics Symposium (IES), 2015 International*. [S.l.], 2015. p. 114–117.

HAN, J.; MORAGA, C. The influence of the sigmoid function parameters on the speed of backpropagation learning. *From Natural to Artificial Neural Computation*, Springer, p. 195–201, 1995.

HARTANTO, R.; KARTIKASARI, A. Android based real-time static indonesian sign language recognition system prototype. In: IEEE. *Information Technology and Electrical Engineering (ICITEE), 2016 8th International Conference on*. [S.l.], 2016. p. 1–6.

HAWKINS, J.; BLAKESLEE, S. *On intelligence: How a new understanding of the brain will lead to the creation of truly intelligent machines*. [S.l.]: Macmillan, 2007.

HE, K.; GKIOXARI, G.; DOLLÁR, P.; GIRSHICK, R. Mask r-cnn. In: *Proceedings of the IEEE international conference on computer vision*. [S.l.: s.n.], 2017. p. 2961–2969.

HE, K.; GKIOXARI, G.; DOLLÁR, P.; GIRSHICK, R. Mask r-cnn. In: *Proceedings of the IEEE international conference on computer vision*. [S.l.: s.n.], 2017. p. 2961–2969.

HENDRYCKS, D.; MAZEIKA, M.; KADAVATH, S.; SONG, D. Using self-supervised learning can improve model robustness and uncertainty. *Advances in Neural Information Processing Systems*, v. 32, 2019.

HO, T. K. Random decision forests. In: IEEE. *Document Analysis and Recognition, 1995., Proceedings of the Third International Conference on*. [S.l.], 1995. v. 1, p. 278–282.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. *Neural computation*, MIT Press, v. 9, n. 8, p. 1735–1780, 1997.

HUANG, J.; FENG, Y. Optimization of recurrent neural networks on natural language processing. In: *Proceedings of the 2019 8th International Conference on Computing and Pattern Recognition*. [S.l.: s.n.], 2019. p. 39–45.

HUANG, J.; ZHOU, W.; LI, H.; LI, W. Sign language recognition using 3d convolutional neural networks. In: IEEE. *Multimedia and Expo (ICME), 2015 IEEE International Conference on*. [S.l.], 2015. p. 1–6.

HUANG, L.; SUN, C.; QIU, X.; HUANG, X. Glossbert: Bert for word sense disambiguation with gloss knowledge. *arXiv preprint arXiv:1908.07245*, 2019.

HUANG, Z.; LENG, J. Analysis of hu's moment invariants on image scaling and rotation. In: IEEE. *Computer Engineering and Technology (ICCET), 2010 2nd International Conference on*. [S.l.], 2010. v. 7, p. V7–476.

HUENERFAUTH, M.; LU, P.; ROSENBERG, A. Evaluating importance of facial expression in american sign language and pidgin signed english animations. In: ACM. *The proceedings of the 13th international ACM SIGACCESS conference on Computers and accessibility*. [S.l.], 2011. p. 99–106.

JI, S.; XU, W.; YANG, M.; YU, K. 3d convolutional neural networks for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, IEEE, v. 35, n. 1, p. 221–231, 2013.

JIN, C. M.; OMAR, Z.; JAWARD, M. H. A mobile application of american sign language translation via image processing algorithms. In: IEEE. *Region 10 Symposium (TENSYMP), 2016 IEEE*. [S.l.], 2016. p. 104–109.

JOLLIFFE, I. Mathematical and statistical properties of population principal components. *Principal Component Analysis*, Springer, p. 10–28, 2002.

JONES, M. J.; REHG, J. M. Statistical color models with application to skin detection. *International Journal of Computer Vision*, Springer, v. 46, n. 1, p. 81–96, 2002.

JUNKER, H.; AMFT, O.; LUKOWICZ, P.; TRÖSTER, G. Gesture spotting with body-worn inertial sensors to detect user activities. *Pattern Recognition*, Elsevier, v. 41, n. 6, p. 2010–2024, 2008.

KAMATH, U.; LIU, J.; WHITAKER, J. *Deep learning for NLP and speech recognition*. [S.l.]: Springer, 2019. v. 84.

KAPUSCINSKI, T. Using hierarchical temporal memory for vision-based hand shape recognition under large variations in hand's rotation. *Artifical Intelligence and Soft Computing*, Springer, p. 272–279, 2010.

KAPUSCINSKI, T.; WYSOCKI, M. Using hierarchical temporal memory for recognition of signed polish words. *Computer Recognition Systems 3*, Springer, p. 355–362, 2009.

KAU, L.-J.; SU, W.-L.; YU, P.-J.; WEI, S.-J. A real-time portable sign language translation system. In: IEEE. *2015 IEEE 58th International Midwest Symposium on Circuits and Systems (MWSCAS)*. [S.l.], 2015. p. 1–4.

KHAMBADKAR, V.; FOLMER, E. A tactile-proprioceptive communication aid for users who are deafblind. In: IEEE. *2014 IEEE Haptics Symposium (HAPTICS)*. [S.l.], 2014. p. 239–245.

KIM, H.-G.; JANG, G.-J.; CHOI, H.-J.; KIM, M.; KIM, Y.-W.; CHOI, J. Recurrent neural networks with missing information imputation for medical examination data prediction. In: IEEE. *2017 IEEE International Conference on Big Data and Smart Computing (BigComp)*. [S.l.], 2017. p. 317–323.

KIM, Y. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

KIMMELMAN, V.; IMASHEV, A.; MUKUSHEV, M.; SANDYGULOVA, A. Eyebrow position in grammatical and emotional expressions in kazakh-russian sign language: A quantitative study. *PloS one*, Public Library of Science San Francisco, CA USA, v. 15, n. 6, p. e0233731, 2020.

KINGMA, D. P.; BA, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

KO, S.-K.; KIM, C. J.; JUNG, H.; CHO, C. Neural sign language translation based on human keypoint estimation. *Applied Sciences*, Multidisciplinary Digital Publishing Institute, v. 9, n. 13, p. 2683, 2019.

KO, S.-K.; KIM, C. J.; JUNG, H.; CHO, C. Neural sign language translation based on human keypoint estimation. *Applied Sciences*, Multidisciplinary Digital Publishing Institute, v. 9, n. 13, p. 2683, 2019.

KO, S.-K.; SON, J. G.; JUNG, H. Sign language recognition with recurrent neural network using human keypoint detection. In: *Proceedings of the 2018 Conference on Research in Adaptive and Convergent Systems*. [S.l.: s.n.], 2018. p. 326–328.

KO, S.-K.; SON, J. G.; JUNG, H. Sign language recognition with recurrent neural network using human keypoint detection. In: *Proceedings of the 2018 conference on research in adaptive and convergent systems*. [S.l.: s.n.], 2018. p. 326–328.

KOLKUR, S.; KALBANDE, D.; SHIMPI, P.; BAPAT, C.; JATAKIA, J. Human skin detection using rgb, hsv and ycbcr color models. *arXiv preprint arXiv:1708.02694*, 2017.

KOLLER, O.; FORSTER, J.; NEY, H. Continuous sign language recognition: Towards large vocabulary statistical recognition systems handling multiple signers. *Computer Vision and Image Understanding*, Elsevier, v. 141, p. 108–125, 2015.

KOLLER, O.; NEY, H.; BOWDEN, R. Read my lips: Continuous signer independent weakly supervised viseme recognition. In: SPRINGER. *European Conference on Computer Vision*. [S.l.], 2014. p. 281–296.

KOLLER, O.; NEY, H.; BOWDEN, R. Deep learning of mouth shapes for sign language. In: *Proceedings of the IEEE International Conference on Computer Vision Workshops*. [S.l.: s.n.], 2015. p. 85–91.

KONRAD, S. G.; MASSON, F. R. Pedestrian skeleton tracking using openpose and probabilistic filtering. In: IEEE. *2020 IEEE Congreso Bienal de Argentina (ARGENCON)*. [S.l.], 2020. p. 1–7.

KOSMIDOU, V. E.; HADJILEONTIADIS, L. J. Sign language recognition using intrinsic-mode sample entropy on semg and accelerometer data. *IEEE transactions on biomedical engineering*, IEEE, v. 56, n. 12, p. 2879–2890, 2009.

KUZNETSOVA, A.; IMASHEV, A.; MUKUSHEV, M.; SANDYGULOVA, A.; KIMMELMAN, V. Using computer vision to analyze non-manual marking of questions in krsl. In: *Proceedings of the 1st International Workshop on Automatic Translation for Signed and Spoken Languages (AT4SSL)*. [S.l.: s.n.], 2021. p. 49–59.

LECUN, Y.; BOTTOU, L.; BENGIO, Y.; HAFFNER, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, IEEE, v. 86, n. 11, p. 2278–2324, 1998.

LECUN, Y.; BOTTOU, L.; ORR, G. B.; MÜLLER, K.-R. Efficient backprop. In: *Neural networks: Tricks of the trade*. [S.l.]: Springer, 1998. p. 9–50.

LEE, C. K.; NG, K. K.; CHEN, C.-H.; LAU, H. C.; CHUNG, S.; TSOI, T. American sign language recognition and training method with recurrent neural network. *Expert Systems with Applications*, Elsevier, v. 167, p. 114403, 2021.

LI, S.-Z.; YU, B.; WU, W.; SU, S.-Z.; JI, R.-R. Feature learning based on sae–pca network for human gesture recognition in rgbd images. *Neurocomputing*, Elsevier, v. 151, p. 565–573, 2015.

LIDDELL, S. K. et al. *Grammar, gesture, and meaning in American Sign Language*. [S.l.]: Cambridge University Press, 2003.

LIU, T.; ZHOU, W.; LI, H. Sign language recognition with long short-term memory. In: IEEE. *Image Processing (ICIP), 2016 IEEE International Conference on*. [S.l.], 2016. p. 2871–2875.

LÓPEZ-LUDEÑA, V.; BARRA-CHICOTE, R.; LUTFI, S.; MONTERO, J. M.; SAN-SEGUNDO, R. Lsespeak: A spoken language generator for deaf people. *Expert Systems with Applications*, Elsevier, v. 40, n. 4, p. 1283–1295, 2013.

LOWE, D. G. *Method and apparatus for identifying scale invariant features in an image and use of same for locating an object in an image*. [S.l.]: Google Patents, 2004. US Patent 6,711,293.

LUO, R. C.; WU, Y.; LIN, P. Multimodal information fusion for human-robot interaction. In: IEEE. *Applied Computational Intelligence and Informatics (SACI), 2015 IEEE 10th Jubilee International Symposium on*. [S.l.], 2015. p. 535–540.

MADHURI, Y.; ANITHA, G.; ANBURAJAN, M. Vision-based sign language translation device. In: IEEE. *Information Communication and Embedded Systems (ICICES), 2013 International Conference on*. [S.l.], 2013. p. 565–568.

MAHALANOBIS, P. C. On the generalised distance in statistics. *Proceedings of the National Institute of Sciences of India, 1936*, p. 49–55, 1936.

MARTÍNEZ, A. M.; KAK, A. C. Pca versus lda. *IEEE transactions on pattern analysis and machine intelligence*, IEEE, v. 23, n. 2, p. 228–233, 2001.

MERITY, S.; KESKAR, N. S.; SOCHER, R. Regularizing and optimizing lstm language models. *arXiv preprint arXiv:1708.02182*, 2017.

NAGI, J.; DUCATELLE, F.; CARO, G. A. D.; CIREŞAN, D.; MEIER, U.; GIUSTI, A.; NAGI, F.; SCHMIDHUBER, J.; GAMBARDELLA, L. M. Max-pooling convolutional neural networks for vision-based hand gesture recognition. In: IEEE. *Signal and Image Processing Applications (ICSIPA), 2011 IEEE International Conference on*. [S.l.], 2011. p. 342–347.

NEEDLEMAN, S. B.; WUNSCH, C. D. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, Elsevier, v. 48, n. 3, p. 443–453, 1970.

NEIVA, D. H.; ZANCHETTIN, C. A dynamic gesture recognition system to translate between sign languages in complex backgrounds. In: IEEE. *Intelligent Systems (BRACIS), 2016 5th Brazilian Conference on*. [S.l.], 2016. p. 421–426.

NEIVA, D. H.; ZANCHETTIN, C. Gesture recognition: A review focusing on sign language in a mobile context. *Expert Systems with Applications*, Elsevier, 2018.

NEVEROVA, N.; WOLF, C.; TAYLOR, G. W.; NEBOUT, F. Multi-scale deep learning for gesture detection and localization. In: SPRINGER. *Workshop at the European conference on computer vision*. [S.l.], 2014. p. 474–490.

NGUYEN, T. D.; RANGANATH, S. Facial expressions in american sign language: Tracking and recognition. *Pattern Recognition*, Elsevier, v. 45, n. 5, p. 1877–1891, 2012.

OTSU, N. A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics*, IEEE, v. 9, n. 1, p. 62–66, 1979.

PAN, T.-Y.; LO, L.-Y.; YEH, C.-W.; LI, J.-W.; LIU, H.-T.; HU, M.-C. Real-time sign language recognition in complex background scene based on a hierarchical clustering classification method. In: IEEE. *Multimedia Big Data (BigMM), 2016 IEEE Second International Conference on*. [S.l.], 2016. p. 64–67.

PAPINENI, K.; ROUKOS, S.; WARD, T.; ZHU, W.-J. Bleu: a method for automatic evaluation of machine translation. In: *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*. [S.l.: s.n.], 2002. p. 311–318.

PARK, C.-B.; LEE, S.-W. Real-time 3d pointing gesture recognition for mobile robots with cascade hmm and particle filter. *Image and Vision Computing*, Elsevier, v. 29, n. 1, p. 51–63, 2011.

PATTANAWORAPAN, K.; CHAMNONGTHAI, K.; GUO, J.-M. Signer-independence finger alphabet recognition using discrete wavelet transform and area level run lengths. *Journal of Visual Communication and Image Representation*, Elsevier, v. 38, p. 658–677, 2016.

PAULSON, B.; CUMMINGS, D.; HAMMOND, T. Object interaction detection using hand posture cues in an office setting. *International Journal of Human-Computer Studies*, Elsevier, v. 69, n. 1, p. 19–29, 2011.

PIGOU, L.; DIELEMAN, S.; KINDERMANS, P.-J.; SCHRAUWEN, B. Sign language recognition using convolutional neural networks. In: SPRINGER. *Workshop at the European Conference on Computer Vision*. [S.l.], 2014. p. 572–578.

PRASUHN, L.; OYAMADA, Y.; MOCHIZUKI, Y.; ISHIKAWA, H. A hog-based hand gesture recognition system on a mobile device. In: IEEE. *2014 IEEE International Conference on Image Processing (ICIP)*. [S.l.], 2014. p. 3973–3977.

PUGEAULT, N.; BOWDEN, R. Spelling it out: Real-time asl fingerspelling recognition. In: IEEE. *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*. [S.l.], 2011. p. 1114–1119.

RAHEJA, J. L.; SUBRAMANIYAM, S.; CHAUDHARY, A. Real-time hand gesture recognition in fpga. *Optik-International Journal for Light and Electron Optics*, Elsevier, v. 127, n. 20, p. 9719–9726, 2016.

RAO, G. A.; KISHORE, P. Selfie video based continuous indian sign language recognition system. *Ain Shams Engineering Journal*, Elsevier, 2017.

RASAMOELINA, A. D.; ADJAILIA, F.; SINČÁK, P. A review of activation function for artificial neural network. In: IEEE. *2020 IEEE 18th World Symposium on Applied Machine Intelligence and Informatics (SAMI)*. [S.l.], 2020. p. 281–286.

RODRIGUES, A. J. *V-LIBRASIL:Uma base de dados com sinais na Língua Brasileira de Sinais (Libras)*. Dissertação (Mestrado) — Centro de Informática-CIN, Universidade Federal de Pernambuco-UFPE, 2021.

ROSENBLATT, F. *Principles of neurodynamics. perceptrons and the theory of brain mechanisms*. [S.l.], 1961.

ROZADO, D.; RODRIGUEZ, F. B.; VARONA, P. Gaze gesture recognition with hierarchical temporal memory networks. In: SPRINGER. *International Work-Conference on Artificial Neural Networks*. [S.l.], 2011. p. 1–8.

ROZADO, D.; RODRIGUEZ, F. B.; VARONA, P. Extending the bioinspired hierarchical temporal memory paradigm for sign language recognition. *Neurocomputing*, Elsevier, v. 79, p. 75–86, 2012.

ROZADO, D.; RODRIGUEZ, F. B.; VARONA, P. Low cost remote gaze gesture recognition in real time. *Applied Soft Computing*, Elsevier, v. 12, n. 8, p. 2072–2084, 2012.

SANTOS, D. G.; FERNANDES, B. J.; BEZERRA, B. L. Hagr-d: a novel approach for gesture recognition with depth maps. *Sensors*, Multidisciplinary Digital Publishing Institute, v. 15, n. 11, p. 28646–28664, 2015.

SCHELP, P. P. Letramento e alunos surdos: práticas pedagógicas em escola inclusiva. In: *Anais do IX Congresso Nacional de Educação (EDUCERE), III Encontro Sul Brasileiro de Psicopedagogia*. [S.l.: s.n.], 2009.

SEYMOUR, M.; TŠOEU, M. A mobile application for south african sign language (sasl) recognition. In: IEEE. *AFRICON, 2015*. [S.l.], 2015. p. 1–5.

SHAH, F.; SHAH, M. S.; AKRAM, W.; MANZOOR, A.; MAHMOUD, R. O.; ABDELMINAAM, D. S. Sign language recognition using multiple kernel learning: A case study of pakistan sign language. *Ieee Access*, IEEE, v. 9, p. 67548–67558, 2021.

SHANABLEH, T.; ASSALEH, K. User-independent recognition of arabic sign language for facilitating communication with the deaf community. *Digital Signal Processing*, Elsevier, v. 21, n. 4, p. 535–542, 2011.

SHARMA, R. P.; VERMA, G. K. Human computer interaction using hand gesture. *Procedia Computer Science*, Elsevier, v. 54, p. 721–727, 2015.

SHARMA, S.; KUMAR, K. Asl-3dcnn: American sign language recognition technique using 3-d convolutional neural networks. *Multimedia Tools and Applications*, Springer, v. 80, n. 17, p. 26319–26331, 2021.

SHUKOR, A. Z.; MISKON, M. F.; JAMALUDDIN, M. H.; ALI, F. bin; ASYRAF, M. F.; BAHAR, M. B. bin et al. A new data glove approach for malaysian sign language detection. *Procedia Computer Science*, Elsevier, v. 76, p. 60–67, 2015.

SRUTHI, S. A. L.; ANGAYARKANNI, A.; PRIYANGA, K.; SRUTHI, S. A. L.; ANGAYARKANNI, A.; PRIYANGA, K. Gaze gesture recognition using optimized ohtm. *International Journal*, IJIRST (International Journal for Innovative Research in Science & Technology), v. 4, p. 57–62, 2017.

TAN, M.; LE, Q. Efficientnet: Rethinking model scaling for convolutional neural networks. In: PMLR. *International Conference on Machine Learning*. [S.l.], 2019. p. 6105–6114.

THAKKER, U.; DASIKA, G.; BEU, J.; MATTINA, M. Measuring scheduling efficiency of rnns for nlp applications. *arXiv preprint arXiv:1904.03302*, 2019.

THALANGE, A.; DIXIT, S. Cohst and wavelet features based static asl numbers recognition. *Procedia Computer Science*, Elsevier, v. 92, p. 455–460, 2016.

THEPADE, S. D.; NARKHEDE, A.; KELVEKAR, P.; KULKARNI, G.; TATHE, S. Novel technique for background removal from sign images for sign language recognition system. *International Journal of Computer Applications*, Citeseer, v. 78, n. 10, 2013.

TIAN, Z.; SHEN, C.; CHEN, H. Conditional convolutions for instance segmentation. In: SPRINGER. *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*. [S.l.], 2020. p. 282–298.

TIEDEMANN, J. The tatoeba translation challenge–realistic data sets for low resource and multilingual mt. *arXiv preprint arXiv:2010.06354*, 2020.

UDDIN, M. T. An ada-random forests based grammatical facial expressions recognition approach. In: IEEE. *Informatics, Electronics & Vision (ICIEV), 2015 International Conference on*. [S.l.], 2015. p. 1–6.

VANĚK, J.; MACHLICA, L.; PSUTKA, J. Estimation of single-gaussian and gaussian mixture models for pattern recognition. In: SPRINGER. *Iberoamerican Congress on Pattern Recognition*. [S.l.], 2013. p. 49–56.

VASWANI, A.; SHAZEER, N.; PARMAR, N.; USZKOREIT, J.; JONES, L.; GOMEZ, A. N.; KAISER, Ł.; POLOSUKHIN, I. Attention is all you need. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2017. p. 5998–6008.

VASWANI, A.; SHAZEER, N.; PARMAR, N.; USZKOREIT, J.; JONES, L.; GOMEZ, A. N.; KAISER, Ł.; POLOSUKHIN, I. Attention is all you need. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2017. p. 5998–6008.

VINTIMILLA, M. G.; ALULEMA, D.; MOROCHO, D.; PROANO, M.; ENCALADA, F.; GRANIZO, E. Development and implementation of an application that translates the alphabet and the numbers from 1 to 10 from sign language to text to help hearing impaired by android mobile devices. In: IEEE. *Automatica (ICA-ACCA), IEEE International Conference on*. [S.l.], 2016. p. 1–5.

VIOLA, P.; JONES, M. J. Robust real-time face detection. *International journal of computer vision*, Springer, v. 57, n. 2, p. 137–154, 2004.

VOEGTLIN, T. Recursive principal components analysis. *Neural Networks*, Elsevier, v. 18, n. 8, p. 1051–1063, 2005.

WADHAWAN, A.; KUMAR, P. Deep learning-based sign language recognition system for static signs. *Neural Computing and Applications*, Springer, p. 1–12, 2020.

WINSTON, P. H. *Artificial intelligence*. [S.l.]: Addison-Wesley Longman Publishing Co., Inc., 1992.

XIE, R.; SUN, X.; XIA, X.; CAO, J. Similarity matching-based extensible hand gesture recognition. *IEEE sensors journal*, IEEE, v. 15, n. 6, p. 3475–3483, 2015.

XU, D.-H.; KURANI, A. S.; FURST, J. D.; RAICU, D. S. Run-length encoding for volumetric texture. *Heart*, v. 27, p. 25, 2004.

YANG, X. Error resilient multiple description coding for mobile sign language communication. In: IEEE. *Image and Signal Processing (CISP), 2013 6th International Congress on*. [S.l.], 2013. v. 1, p. 21–25.

YIN, K.; READ, J. Better sign language translation with stmc-transformer. In: *Proceedings of the 28th International Conference on Computational Linguistics*. [S.l.: s.n.], 2020. p. 5975–5989.

YU, Y.; SI, X.; HU, C.; ZHANG, J. A review of recurrent neural networks: Lstm cells and network architectures. *Neural computation*, MIT Press One Rogers Street, Cambridge, MA 02142-1209, USA journals-info . . . , v. 31, n. 7, p. 1235–1270, 2019.

ZEYER, A.; BAHAR, P.; IRIE, K.; SCHLÜTER, R.; NEY, H. A comparison of transformer and lstm encoder decoder models for asr. In: IEEE. *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. [S.l.], 2019. p. 8–15.

ZHANG, A.; LEI, X.; CHEN, X.; CHEN, L. Fast segmentation of sign language video based on cellular neural network. *Jisuanji Yingyong/ Journal of Computer Applications*, Chengdu Institute of Computer Applications, 16 Donghuangchenggen North Street Beijing 100717 bibomputerapplications. com. c www. computerapplications. com. cn, v. 33, n. 2, p. 503–506, 2013.

ZHOU, H.; ZHOU, W.; ZHOU, Y.; LI, H. Spatial-temporal multi-cue network for continuous sign language recognition. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. [S.l.: s.n.], 2020. v. 34. no. 7, p. 13009–13016.

# APPENDIX A – SENTENCES CREATED BY COMBINING WORDS IN LIBRAS

- 'acidente', 'destruir', 'cama'
- 'acidente', 'destruir', 'computador'
- 'acidente', 'destruir', 'cadeira de balanço'
- 'criança', 'crescer', 'desassistido'
- 'criança', 'crescer', 'avô'
- 'carpinteiro', 'culpa, culpar','bombeiro'
- 'chefe', 'culpa, culpar','autoridade'
- 'biscoito', 'cair', 'de novo'
- 'corsa, veado', 'assistir', 'cavalo'
- 'devemos', 'desconectar, desligar', 'computador'
- 'computador', 'desconectar, desligar'
- 'bombeiro', 'aceitar', 'acompanhar', 'criança'
- 'bombeiro', 'aceitar', 'acompanhar', 'autoridade'
- 'avô', 'colocar', 'cobertor', 'acima', 'cama'
- 'chefe', 'chegar', 'cheio', 'culpa, culpar'
- 'criança', 'comer', 'com frequência'
- 'carpinteiro', 'comunicar', 'com frequência', 'desapontamento'
- 'avô', 'deprimir', 'de novo'
- 'almofada', 'azul'
- 'avô', 'doente', 'de novo'
- 'criança', 'doente', 'de novo'

- 'depósito', 'disponível'
- 'depósito', 'continuar', 'disponível'
- 'avô', 'continuar', 'com medo', 'cadeira de balanço'
- 'autoridade', 'agir, fazer', 'bem'
- 'ambos', 'bater papo', 'com frequência'
- 'calculadora', 'colaborar', 'com frequência'
- 'autoridade', 'agressivo', 'de novo'
- 'bolsa', 'azul', 'cair', 'cerca, barreira'
- 'autoridade', 'discriminar', 'ambos'
- 'bomba', 'destruir', 'com frequência', 'cidade, comunidade'
- 'bomba', 'destruir', 'cidade, comunidade', 'acidente'
- 'autoridade', 'colaborar', 'bem'
- 'bombeiro', 'agir, fazer', 'com medo'
- 'criança', 'comemorar, celebrar', 'aniversário', 'animado'
- 'borboleta', 'congelar', 'de novo'
- 'avô', 'cego', 'cair', 'de novo'
- 'correspondente', 'comunicar', 'com medo'
- 'carpinteiro', 'desistir', 'divórcio', 'de novo'
- 'asno', 'agressivo', 'brigar, porrada', 'com frequência'
- 'deus', 'abençoar', 'criança'
- 'desafio', 'agora', 'abrir a janela'
- 'criança', 'comer', 'biscoito'
- 'desafio', 'agora', 'abrir a porta'

- 'boi, búfalo', 'confrontar, enfrentar', 'corsa, veado'
- 'ar', 'acima', 'agradável, limpo'
- 'ambos', 'aceitar', 'deus'
- 'autoridade', 'cancelar', 'benefício, vantagem'
- 'brinco', 'chique', 'decorar, ornamentar', 'campeão'
- 'botão, broche', 'brilho, brilhar', 'coroa'
- 'cálculo', 'difícil', 'congelar', 'amigo'
- 'barba', 'crescer', 'com frequência'
- 'autoridade', 'demitir, demitido', 'carpinteiro'
- 'carpinteiro', 'consertar', 'barco', 'amanhã'
- 'deus', 'decorar, ornamentar', 'céu', 'azul'
- 'céu', 'azul', 'agradável, limpo', 'agora'
- 'chocolate', 'delicioso'
- 'aniversário', 'animado', 'de novo'
- 'doente', 'desassistido', 'desistir'
- 'capacete', 'confortável'
- 'criança', 'cópia', 'avô'
- 'criança', 'comemorar, celebrar', 'aniversário', 'amanhã'
- 'associação', 'construir', 'de novo', 'departamento'
- 'adulto', 'aceitar', 'desafio', 'cálculo', 'difícil'
- 'dólar', 'cair', 'de novo'
- 'calouro, novato', 'confrontar, enfrentar', 'desafio'
- 'céu', 'azul', 'brilho, brilhar', 'com frequência'
- 'chefe', 'avaliar', 'cultura', 'departamento', 'de novo'