HandArch: A deep learning architecture for LIBRAS hand configuration recognition

1st Gabriel Peixoto de Carvalho
Center of Mathematics Computing and Cognition (CMCC)
Federal University of ABC
Santo Andre - SP - Brazil
Email: gabriel.carvalho@ufabc.edu.br

2nd André Luiz Brandão
Center of Mathematics Computing and Cognition (CMCC)
Federal University of ABC
Santo Andre - SP - Brazil
Email: brandao@daad-alumni.de

3rd Fernando Teubl Ferreira
Center of Mathematics Computing and Cognition (CMCC)
Federal University of ABC
Santo Andre - SP - Brazil
Email: fernando.teubl@ufabc.edu.br

***Abstract*—Despite the recent advancements in deep learning, sign language recognition persists as a challenge in computer vision due to its complexity in shape and movement patterns. Current studies that address sign language recognition treat hand pose recognition as an image classification problem. Based on this approach, we introduce HandArch, a novel architecture for real-time hand pose recognition from video to accelerate the development of sign language recognition applications. Furthermore, we present Libras91, a novel dataset of Brazilian sign language (LIBRAS) hand configurations containing 91 classes and 108,896 samples. Experimental results show that our approach surpasses the accuracy of previous studies while working in real-time on video files. The recognition accuracy of our system is 99% for the novel dataset and over 95% for other hand pose datasets.**

***Index Terms*—Sign Language Recognition, LIBRAS, Deep Learning, Software Architecture, Hand Configurations**

I. Introduction

Sign languages are natural languages and have five main components: (1) facial expression, (2) hand orientation, (3) hand movement, (4) gesture localization, and (5) hand configuration [1], [2]. Studies addressing Sign Language Recognition (SLR) focus on a single component due to the complexity of this tasks [3]. Different studies approach sign language recognition as a hand configuration classification problem by identifying the pose performed in an image or a sequence of images [3]. However, these studies do not address the applicability of this knowledge for accessibility applications because they work with hand images isolated with human assistance in controlled environments [4], [5]. For instance, there is no constant illumination or static background in a real-world scenario, making it difficult to achieve automatic hand isolation. Moreover, there is no specification on building SLR applications in the literature, creating a barrier for new researchers entering the field.

We present HandArch, a novel software architecture for sign language recognition that considers real-time and real-world scenarios to solve the aforementioned issues. Our architecture is modular, reconfigurable, and contains all the components necessary to build a real-time hand pose recognition system: (1) detection, (2) tracking, (3) segmentation, and (4) classification. With this design choice, we unify different approaches under a single software architecture overcoming the aforementioned limitations. We compare our proposal to past works using both classical and deep learning methods. Furthermore, we present Libras91, a novel dataset on Brazilian Sign Language (LIBRAS) containing 91 hand configurations [2] in different orientations and points of view. The dataset contains 108,896 images[1], which supports its applicability in deep learning training. We use different sign languages datasets to assess the applicability of the proposed architecture. The main contributions of this study are the following:

1) Modular and reconfigurable architecture for SLR;
2) Novel dataset for LIBRAS with 91 hand configurations;
3) Assessment of the applicability of our architecture.

II. Related Work

There are different taxonomies for the SLR systems in the survey studies [3]. The main common characteristics are: (1) the type of sign and (2) sensors used for hand pose/movement acquisition. The primary types of signs in literature are static and dynamic, where static signs consider a hand pose [4], [6]–[8] and dynamic signs consider the sequence of movements in combination with the hand poses [9]. For sensors, there are different sensors considered in previous works, from wearable devices [10] to depth cameras [4], [6]–[8]. This study considers static signs (poses) as hand configurations in images and videos (frames) from RGB cameras.

Filho *et al.* [4] present a novel dataset, feature descriptor, and classifier. The database has 12,200 samples divided in 61 classes of Brazilian sing language (LIBRAS) collected by ten volunteers. They classify depth masks summarized by Linear Discriminant Analysis (LDA) and Principal Component Analysis (PCA) with their novel filter classification and report an accuracy of 95%.

Bastos *et al.* [8] propose a hand pose SLR system evaluated in two different datasets. They present a new hand pose dataset of 40 classes (26 alphabet letters, numbers, and words). Their SLR system uses Neural Networks (NN) for pixel-wise skin

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

[1]We obtained the dataset from the frames of 50 seconds videos.

segmentation and Histogram of Oriented Gradients (HOG) and Zernike moments descriptors with NN and reports an accuracy of 96.77% combining both feature descriptors.

Both Rahman *et al.* [5] and Sruthi *et al.* [11] propose a novel Deep Neural Network (DNN) for SLR. Rahman *et al.* [5] apply the DNN in four different American sign language (ASL) datasets containing 29 classes, and they obtain 100% accuracy. Sruthi *et al.* [11] apply the DNN in a new Indian sign language (ISL) dataset with 4,125 samples divided in 24 classes and reports an accuracy of 98,64%.

Tolentino *et al.* [12] propose a novel framework for teaching ASL through an SLR system. This study used three different datasets: 26 alphabets, 10 numbers, and 35 static words. They use a DNN architecture and silhouette segmented hand images for recognition and obtain a maximum accuracy of 97.52% in the static word dataset.

Hosoe *et al.* [13] present a framework for Japanese Sign Language (JSL) alphabet sign recognition. They created a dataset with 41 signs with 8,000 samples (expanded using 3D modeling). For classification, the authors propose the use of DNN trained from scratch to recognize the sign. The authors report a maximum accuracy of 98%.

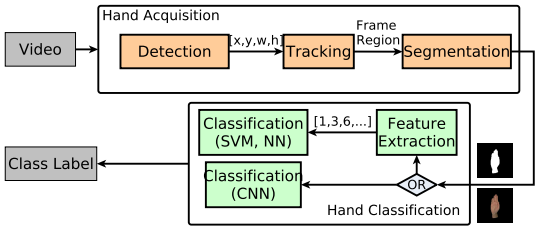
Most of the studies focus on the classification of static signs in the form of alphabets and numbers. Sign language alphabets and numbers are present in most datasets but have limited applicability. In contrast, hand configurations are present in every sign, thus having broader applicability. Furthermore, the aforementioned work focuses on image classification in controlled environments and does not consider realistic usage scenarios (e.g., complex background, different illumination) for hand image acquisition.

Our architecture extends the limitations of the aforementioned work both in terms of applicability and recognition accuracy. We compare our architecture accuracy with the aforementioned works that have datasets available. We also consider datasets from RGBD (color with depth information) cameras because the final result is a binary mask similar to the one obtained from an RGB camera after color segmentation. Thus, they are the same for the classification task.

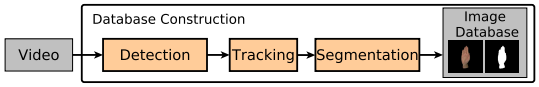
III. HandArch Architecture

The SLR studies we discussed in the previous section do not address the real-world applicability of SLR. They focus on improving recognition accuracy. Some works like Tolentino *et al.* [12] approach that topic by presenting a semi-supervised dataset creation tool. Carvalho *et al.* [14] present a similar approach, but the author's approach does not support hand detection, Deep Learning (DL) methods and is focused on hand pose recognition (not SLR but virtually the same application). The approaches mentioned earlier are not enough to realize a real-world capable SLR system. There are other variables to consider, such as illumination, occlusion, background, and skin tone differences.

To solve these issues, we present **HandArch** architecture that unifies different approaches for SLR by defining a standard design for an SLR and adding different layers enables



(a) **HandArch** architecture block diagram.



(b) **HandArch** Applied to dataset creation tasks.

Fig. 1: **HandArch** diagram and application example.

the applicability of SLR methods in real-world scenarios. The additional tasks make the system effective in different scenarios with occlusion, changing lighting conditions, and complex backgrounds. We design the architecture to be modular and reconfigurable to support different SLR applications, so it is possible to combine or remove layers to fit the application's requirements.

We derive the architecture blocks, tasks, and methods from the standard SLR methodology found in other surveys [3], [15]. The main blocks of the architecture are (1) Detection, (2) Tracking, (3) Segmentation, (4) Classification. Each block has standardized inputs and outputs, which enables the usage of a block on its own or combines any number of blocks to create a system. Figures 1a, and 1b illustrate two applications of the architecture: (1) a complete SLR system, and a (2) semi-automatic dataset collection tool.

We integrate different methods in the architecture, from more classic (e.g., Viola-jones) methods to more complex methods, including DL. With these diverse methods, the architecture supports prototyping different applications, from embedded applications to more accurate and high-performance sign language recognition applications.

The hand detection block applies object detection methods to locate the hand in an image. HandArch currently supports single-hand detection and tracking, which is a design decision due to the use case of the architecture for the recognition of single-hand LIBRAS hand configuration. Therefore, we can support dual hand tracking by adding this feature to the detection and tracking methods. HandArch supports both classical and DL methods. We use Viola-jones [16], hand color detection (Gaussian Mixture Models — GMM), hand motion detection (Optical Flow — OF), and DL object detection (YOLO, SDD, FRCNN) [17] for detection methods. Detection is the heaviest and slowest task in the system because the algorithm needs to search for candidates in the whole picture. Thus we decide to apply object tracking with detection to ease the computational burden of the system. The hand detection block outputs a Region of Interest (ROI) around the hand in the format $[x, y, w, h]$, where x, y are the coordinates of the

top right corner of the ROI and w, h are the width and height of the ROI rectangle, respectively. Hand tracking continues to provide this ROI updated in subsequent frames.

The hand tracking block receives an initial ROI and follows its movement on the subsequent frames. There are different ways to achieve object tracking, so algorithms use characteristics such as color and movement. For color detection, our architecture supports CAMshift [16] and Meanshift [16] algorithms. For movement tracking, we use OF [16]. Furthermore, we apply Bayesian filters to refine the initial estimates for the tracking methods, and the architecture supports Kalman [18] and particle filters for this task. We also consider successive detections as a form of object tracking, so it is possible to bypass the tracking block entirely if we apply this approach. However, successive detection is usually more resource-intensive and slower than tracking.

The hand segmentation block isolates the hand region from the background. This block works on the cropped ROI it receives from the detection/tracking block and outputs a binary mask where the white pixels (255) are the hand pose, and black pixels (0) are the background. We use the same approach as the previous blocks, i.e., we consider both classic and DL approaches. We integrate the architecture methods for color segmentation using a basic threshold in HSV color space and pixel binary classification supported by machine learning algorithms (e.g., binary trees — BT, Support Vector Machines — SVM, GMM) and Simple Linear Iterative Clustering (SLIC) [19] for speedup. We also consider DL methods for segmentation by integrating UNET [20] into the architecture.

The hand detection, tracking, and segmentation blocks compose the hand acquisition part of the architecture. The hand acquisition goal is to isolate the hand image in real-world and unpredictable scenarios, with complex background and partial occlusions. This part of the architecture simplifies the hand pose image by removing the background so that the classification algorithms can focus only in critical information, given the similarity between poses.

The last part in the architecture is hand sign classification. The Machine Learning (ML) methods classify the isolated hand images in their respective sign meaning in the image classification block. The classification block receives the output of the segmentation block (mask or segmented pose with color) and outputs a class label. In the architecture, we integrate two types of ML methods: classic and DL. Classic methods rely on hand-crafted features for training and classification. These methods require less data for training and can reach a satisfactory accuracy but have problems in generalizing. For classic methods, we integrate HOG [21], Hu Moments [21], Scale Invariant Feature Transform (SIFT) [21], and Speed-up Robust Features (SURF) [21] for feature descriptors and shallow multi-layer perceptron (MLP) [17] and SVM [17] for classification algorithms.

From 2015 to 2021, DL methods have been the standard in computer vision studies. This approach does not require hand-crafted features, and it obtains a high-level image description by a sequence of trainable filters resulting in a better generalization capability than the classic methods. However, to reach this generalization, DL requires a massive quantity of samples. We integrated different DNN architectures to the HandArch architecture ranging from LeNet [17] to ResNet [17].

HandArch is modular and reconfigurable. Each block or task can work independently, so it is possible to apply only hand detection to images or classification to the hand image dataset. Each block supports different methods, and all the methods follow the same input/output standard to change the methods in each block without any significant repercussions to the rest of the system. The same goes for adding new methods and features to the system; as long as it follows the input-output standards of the architecture, the method should work fine with the others. Figure 1a contains details of the input/output standard of the architecture. With this design, we unify different approaches for SLR present in the literature, making it easier and faster to prototype and develop new SLR applications compared to the previous work we discussed in the last section.

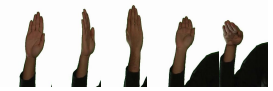
For this study, we consider three possible applications of the proposed SLR architecture: (1) semi-automatic dataset creation, (2) benchmark tool, and (3) prototyping tool for real-world real-time SLR systems. The first application is the semi-automatic dataset creation, where we apply the blocks of hand detection, tracking, and segmentation to extract the hand images from a video (Figure 1b). We use this application to create our dataset. The second application results from the reconfigurable design, where we can use the architecture as a benchmark tool for methods, i.e., we can quickly compare different methods using the same conditions for the whole system. For example, we can quickly use the classification block to compare different SLR classification methods in a single dataset. The third application is a real-time real-world SLR system that incorporates all the methods of the architecture (Figure 1a).

IV. LIBRAS91 DATASET

We propose a new dataset in LIBRAS hands configurations called **Libras91** (our novel dataset). Previous datasets consider 61 hand configurations [4]. **Libras91** contains 91 hand configurations [2], and as far as we know, this is the first dataset that considers this amount of hand configurations. Furthermore, **Libras91** presents different hand orientations for each configuration,

There are no standard guidelines for collecting sign languages dataset in the literature, and each work considers slightly different characteristics, such as pose angle and hand scale [4], [8]. We follow a similar approach to previous works by collecting the dataset in a uniform background and a constant illumination. We consider different hand orientations and angles for each hand configuration, which is beyond any condition considered by the previous datasets in LIBRAS. We had two volunteers collecting the dataset, and each volunteer performed the hand configuration in a short video of 50 seconds where they change its orientation and angle considerably. The volunteers were not at a fixed distance from the camera.

(a) Similar hand configurations (configurations 16 to 20).



(b) Different orientations in the same gesture.

Fig. 2: Samples of our dataset (**Libras91**).

Fig. 3: Example of the tracking dataset in natural light.

Figures 2b present an example of the classes. Our dataset contains 108,896 samples (approximately 1200 samples per class), in which we use 76,155 (70%) samples for training and 32,741 (30%) for validation. This dataset also exceeds the previous ones in the number of classes and samples, which describes LIBRAS configurations better and allows application with deep learning methods.

Libras91 offers several challenges because there are similarities between hand configurations, and depending on the orientation, the hand configurations can also be similar. Figure 2b presents some similar hand configuration examples.

Furthermore, we also notice a lack of hand tracking datasets in the literature. Therefore, we create a simple hand tracking dataset to test our architecture tracking methods. This dataset has three video scenarios of approximately 5 minutes each with manual hand ROI annotation. The first scenario is a garden with natural light and complex background, the second scenario is a kitchen with uniform background and artificial light, and the third one is a living room with complex background and a mix of natural and artificial light. All the videos contain self-occlusion in hand and require re-tracking because the hand leaves the frame. Figure 3 shows a frame of the garden scenario that illustrates the dataset characteristics.

V. Experiments

We conduct two experimental studies to validate our architecture (HandArch): single method experiments (Experiment 1) and architecture experiments (Experiment 2). We evaluate each block alone on the single method experiments and build upon the architecture by adding blocks together. Therefore, we certify that each block of the architecture can properly work standalone. Finally, we combine the best methods in the architecture and use them to evaluate the videos we recorded while collecting our dataset because this presents a more realistic scenario where the hand location is unknown, and motion blur can cause errors in classification. We use specific evaluation metrics for each type of problem so that we can assess their performance reasonably.

We use the average Intersection Over Union (IOU) evaluation metric to compare the hand detection and tracking methods. To evaluate hand detection, we apply the different algorithms in two datasets: Egohands [22] and VIVA [23]. There was only one dataset available for hand tracking, a single video for the Visual Object Tracking (VOT) challenge [24] challenge. We also use our tracking dataset to evaluate the methods in challenging conditions.

Hand segmentation is more straightforward to evaluate. We use a simple pixel-wise comparison between the resulting mask of our method and the ground truth mask and run time. To evaluate segmentation methods, we focus on skin segmentation datasets. We use a Polish Sign Language dataset [25] with skin region masks, the SFA dataset [26] with facial skin segmentation masks, and Sttötinger *et. al.* skin dataset [27], which has a more generic and challenging skins segmentation scenario. Both [26], [27] have a wide range of skin tone samples but are not specialized in hands, where [25] is focused on hand skin segmentation but has smaller skin tone variation.

For the classification blocks of the architecture, we use mainly the standard accuracy metric ($\frac{Correct\ Predictions}{Total\ Predictions}$). We separate the classic methods from the DL methods in the evaluation for simplicity. In our search for datasets, we verified that the datasets created by [5], [11], [12] were not available. Therefore, we compare our results with previous works in different sign languages such as Irish Sign Language [28], JSL [13], and others [4], [8], [29]–[31].

VI. Results

Experiment 1 goal is to validate each block of the hand Arch architecture individually, so we tested each block individually (hand detection, tracking, segmentation, and classification). We evaluated the hand detection in two hand detection datasets. Table I presents the result of the methods we integrated on the hand detection task. We noticed that the CNN-based detectors have the best results in this problem, so we considered them for experiment 2.

TABLE I: Detection experiments results.

Dataset	Method	IOU (avg)	Time (s)
[22]	Cascade	0.013	0.013
[22]	GMM	0.9	6.86
[22]	**SSD**	**0.82**	**0.007**
[22]	YOLO	0.22	0.008
[22]	FRCNN	0.34	0.25
[23]	Cascade	0.0007	0.016
[23]	Color	0.18	7.16
[23]	SSD	0.12	0.007
[23]	**YOLO**	**0.940**	**0.013**
[23]	FRCNN	0.12	0.21

Table II shows that the performance of the tracking algorithms was suboptimal given the complexity of the hand movements and background. Meanshift method obtained the best result, so we selected it for experiment 2. Furthermore, for

TABLE II: Tracking experiments results.

	VOT [24]	Garden	Kitchen	Living Room
Method	IOU(avg)	IOU(avg)	IOU(avg)	IOU(avg)
MeanShift	**0.25**	**0.50**	**0.24**	**0.04**
CAMSHIFT	0.08	0.14	0.29	0.07
OF	0.19	0.07	0.21	0.14
Kalman Filter	0.07	0.16	0.25	0.05
Particle Filter	0.08	0.13	0.27	0.07

TABLE III: Skin segmentation results.

	SFA [26]		Polish SL [25]		[27]	
Method	A(%)	T(s)	A(%)	T(s)	A(%)	T(s)
Static	**90.4**	**0.009**	**77.9**	**0.0008**	**39.9**	**0.0005**
GMM	89.1	1.89	82.1	0.79	50.9	0.62
BT	**82.6**	**0.006**	**86.3**	**0.001**	**62.1**	**0.003**
NB	51.3	0.03	49.2	0.01	26.6	0.01
MLP	86.4	0.87	72.7	0.75	28.9	0.26
SVM	62.4	17.0	93.1	7.2	58.4	8.1
SLIC+NB	55.1	0.83	52.7	0.21	28.3	0.24
SLIC+MLP	61.1	0.87	65.2	0.21	45.4	0.23
UNET	92.3	0.122	84.6	0.106	51.1	0.07

experiment 2, we improve tracking accuracy by periodically restarting the tracker with a new hand location we obtain by executing the hand detection block every 3 seconds. With this approach, we do not sacrifice speed and performance and still maintain high accuracy.

The segmentation results in Table III were unexpected because the simple HSV threshold for skin color still obtained better results than more complex methods. However, in challenging scenarios, such as [27], it fell short. Also, UNET presented the best results in terms of accuracy, but its run time is over ten times larger than the run time of the BT and Static methods. The method that presented the best accuracy and run time is the Binary Tree (BT) method we trained with samples from the SFA dataset, so we considered it for experiment 2.

For hand pose/configuration classification, we considered a classical approach with feature descriptors and classification algorithms and the CNN approach. We obtained satisfactory results on the datasets we analyzed, surpassing previous work's accuracy on the same datasets. We performed hyperparameter optimization on both SVM and MLP parameters (e.g., Kernel, hidden layers) to achieve this result. Tables IV presents the results for the SVM and MLP. We obtained better results using the HOG feature descriptor combined with SVM on the dataset but had difficulty generalizing. At the same time, MLP performed better in a real-world scenario.

We performed tests in two steps to evaluate CNN architectures: (1) we assessed different architectures without hyperparameter optimization or data augmentation, (2) we selected the best architecture for the datasets we have and then execute hyperparameter optimization and data augmentation to improve the results. Initially, the datasets got unsatisfactory results in some architectures due to the relationship between the number of samples and the CNN size. However, after hyperparameter optimization and data augmentation, the results improved due to increasing the number of samples and sample

TABLE IV: Accuracy results for SVM and MLP.

	HOG		HOG+PCA		Hu		SIFT		SURF	
Dataset	SVM	MLP	SVM	MLP	SVM	MLP	SVM	MLP	SVM	MLP
[8]	**100**	**100**	100	100	28	36	98	98	100	100
[30]	**71**	**70**	71	70	57	50	74	75	73	73
[13]	**99**	**98**	99	97	26	32	95	93	97	95
[31]	**100**	**100**	100	100	70	69	100	99	100	100
[28]	**95**	**94**	95	96	12	15	89	85	88	83
[29]	**91**	**90**	91	88	-	-	51	36	61	40
[4]	**99**	**97**	99	96	15	24	34	34	46	44
Libras91	**97**	**94**	97	95	10	24	81	70	89	81

TABLE V: Classification results for CNN architectures.

Dataset	LeNet	Alexnet	VGG16	Inception	Resnet	**Alexnet**
[8]	2.44%	99.9%	2.52%	37.6%	43.15%	**99%**
[4]	1.6%	91.78%	1.63%	2.11%	9.09%	**99.8%**
[13]	3.19%	98.40%	2.56%	20.46%	16.29%	**96%**
[30]	13.75%	61.38%	4.25%	28.37%	17.25%	**84%**
[29]	9.55%	84.8%	9.6%	65.1%	54.4%	**94%**
[28]	78.11%	85%	4%	10%	10.29%	**100%**
[31]	15.55%	98.72%	12.02%	32.88%	27.52%	**96%**
Libras91	72.95%	80.27%	1.28%	1.48%	5%	**99%**

variability in the datasets. Table V contains the summary of this experiment, where in the last column, we have the Alexnet results after optimizations.

In experiment 2, we combined all the blocks in the **HandArch** and used the videos we recorded to generate the dataset. We evaluate the **HandArch** in 182 videos of 50 seconds. We tested the **HandArch** in a Core i7-7500U (dual-core) 2.7 GHz computer with a discrete graphics card and 16GB RAM. We compare the classic classification approach with the CNN classification. Both used the same hand acquisition blocks (SDD, Meanshift, BT segmentation). The architecture processed the video in 10 Frames Per Second (FPS)(10 FPS using the classical approach and 4 FPS using CNN). For classification, we applied the system on both subject videos while measuring the final accuracy on the classified frames, which resulted in a mean accuracy of 81.5% for SVM and 86.5% for CNN. Our architecture (HandArch) is available online at [32].

Finally, we compared the current study with related previous studies in Table VI. Most of the previous studies used 61 hand configurations [4], [6] or other approaches with less number of classes (alphabet) [7], [8]. The present study considers 91 hand configurations, which makes classification more challenging. However, even with increased complexity, this study achieved better accuracy than the previous works, outperforming studies that use RGB and RGBD data.

TABLE VI: Comparison with previous work.

Study	# Classes	# Samples	Acc. (Orig)	Acc. (Our)
[4]	61	12,200	95%	99.8%
[8]	40	9,600	96.77%	99%
[13]	41	5,000	98%	96%
[28]	26	50,000	99%	100%
Libras91	91	108,896	-	99%

VII. Final Considerations

This study presents the HandArch software architecture with multiple possible applications from data acquisition to SLR prototyping. We also present Libras91, which consists of 91 hand configurations and enough samples to apply in CNN efficiently. We believe this architecture and dataset can contribute to developing accessibility applications for sign language translation. We make both architecture and dataset freely available [32].

We also perform thorough testing to evaluate the applicability of the architecture in different scenarios. We compare classic methods with state-of-the-art methods (until September 2021) in challenging scenarios. The objective of this comparison is that the architecture can fulfill different application requirements (e.g., background, illumination, occlusion). By comparing the architecture with previous studies, we prove that it is flexible enough to work with different datasets (a different model for each dataset) and surpass previous work accuracy by using DNN.

The results in the final test present a similar accuracy in both classification methods, but CNN was able to generalize better than SVM with HOG. The videos contain samples that were not used for training since we used only 25% of the samples for training and contained blurred images and noisy images. Therefore, we consider this last test a challenging test for the classifiers, and we consider the accuracy above 80% satisfactory due to the considerable variation in the test data. For future works, we aim at expanding the architecture beyond poses and consider dynamic gesture modeling with subunits [15].

References

[1] R. M. de Quadros, *Lingua de Sinais Brasileira: Estudos Linguisticos*. Artmed Editora, 2004.

[2] K. M. O. Kumada, I. R. Silva, J. M. D. Martino, and V. R. R. da Nóbrega, "Desafios para a tradução de um livro didático de ciências com uso de avatares expressivos," in *I encontro do Centro de ensino, pesquisa e extensão sobre educação de surdos e Libras (Ceslibras)*, 2015.

[3] M. J. Cheok, Z. Omar, and M. H. Jaward, "A review of hand gesture and sign language recognition techniques," *International Journal of Machine Learning and Cybernetics*, Aug 2017.

[4] C. F. F. C. Filho, R. S. de Souza, J. R. dos Santos, B. L. dos Santos, and M. G. F. Costa, "A fully automatic method for recognizing hand configurations of brazilian sign language," *Research on Biomedical Engineering*, vol. 33, no. 1, pp. 78–89, mar 2017.

[5] M. M. Rahman, M. S. Islam, M. H. Rahman, R. Sassi, M. W. Rivolta, and M. Aktaruzzaman, "A new benchmark on american sign language recognition using convolutional neural network," in *2019 International Conference on Sustainable Technologies for Industry 4.0 (STI)*, 2019.

[6] A. J. Porfirio, K. L. Wiggers, L. E. Oliveira, and D. Weingaertner, "LIBRAS sign language hand configuration recognition based on 3d meshes," in *2013 IEEE International Conference on Systems, Man, and Cybernetics*. IEEE, oct 2013.

[7] R. Hartanto and A. Kartikasari, "Android based real-time static indonesian sign language recognition system prototype," in *2016 8th International Conference on Information Technology and Electrical Engineering (ICITEE)*. IEEE, oct 2016.

[8] I. L. Bastos, M. F. Angelo, and A. C. Loula, "Recognition of static gestures applied to brazilian sign language (libras)," in *2015 28th SIBGRAPI Conference on Graphics, Patterns and Images*, 2015.

[9] N. C. Camgöz, O. Koller, S. Hadfield, and R. Bowden, "Sign language transformers: Joint end-to-end sign language recognition and translation," *CoRR*, vol. abs/2003.13830, 2020.

[10] N. Escudeiro, P. Escudeiro, F. Soares, O. Litos, M. Norberto, and J. Lopes, "Recognition of hand configuration: A critical factor in automatic sign language translation," in *2017 12th Iberian Conference on Information Systems and Technologies (CISTI)*. IEEE, jun 2017.

[11] S. C.J. and L. A., "Signet: A deep learning based indian sign language recognition system," in *2019 International Conference on Communication and Signal Processing (ICCSP)*, 2019, pp. 0596–0600.

[12] L. K. S. Tolentino, , R. O. S. Juan, A. C. Thio-ac, M. A. B. Pamahoy, J. R. R. Forteza, and X. J. O. Garcia, "Static sign language recognition using deep learning," *International Journal of Machine Learning and Computing*, vol. 9, no. 6, pp. 821–827, Dec. 2019.

[13] H. Hosoe, S. Sako, and B. Kwolek, "Recognition of jsl finger spelling using convolutional neural networks," in *2017 Fifteenth IAPR International Conference on Machine Vision Applications (MVA)*, 2017.

[14] G. P. d. Carvalho, F. T. Ferreira, and A. L. Brandão, "Comparing pose recognition algorithms and introducing a new approach," in *Proceedings...* Conference on Graphics, Patterns and Images, 30. (SIBGRAPI), 2017.

[15] R. Elakkiya, "Machine learning based sign language recognition: a review and its research frontier," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 7, pp. 7205–7224, Aug. 2020.

[16] A. Kaehler and G. Bradski, *Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library*. O'Reilly Media, 1 2017.

[17] A. Géron, *Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems, 2nd Edition*. O'Reilly Media, 2019.

[18] S. Challa, M. R. Morelande, and D. Musicki, *Fundamentals of Object Tracking*. CAMBRIDGE UNIV PR, 2011.

[19] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "Slic superpixels compared to state-of-the-art superpixel methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012.

[20] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," 2015.

[21] M. Nixon, *Feature Extraction and Image Processing for Computer Vision*. Academic Press, 2012.

[22] S. Bambach, S. Lee, D. J. Crandall, and C. Yu, "Lending a hand: Detecting hands and recognizing activities in complex egocentric interactions," in *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[23] LISA, "The Vision for Intelligent Vehicles and Applications (VIVA) Challenge, Laboratory for Intelligent and Safe Automobiles, UCSD," http://cvrr.ucsd.edu/vivachallenge/index.php/hands/hand-detection/, 2019, accessed: 2019-09-10.

[24] M. Kristan, J. Matas, A. Leonardis, T. Vojir, R. Pflugfelder, G. Fernandez, G. Nebehay, F. Porikli, and L. Čehovin, "A novel performance evaluation methodology for single-target trackers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, Nov 2016.

[25] M. Kawulok, J. Kawulok, J. Nalepa, and B. Smolka, "Self-adaptive algorithm for segmenting skin regions," *EURASIP Journal on Advances in Signal Processing*, vol. 2014, no. 170, pp. 1–22, 2014.

[26] J. P. B. Casati, D. R. Moraes, and E. L. L. Rodrigues, "Sfa: a human skin image database based on feret and ar facial images," in *Workshop de Visão Computacional - WVC*, 2013.

[27] J. Stöttinger, A. Hanbury, C. Liensberger, and R. Khan, "Skin paths for contextual flagging adult videos," in *Advances in Visual Computing*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 303–314.

[28] M. Oliveira, H. Chatbri, S. Little, N. E. O'Connor, and A. Sutherland, "A comparison between end-to-end approaches and feature extraction based approaches for sign language recognition," in *2017 International Conference on Image and Vision Computing New Zealand (IVCNZ)*.

[29] P. K. Pisharady, P. Vadakkepat, and A. P. Loh, "Attention based detection and recognition of hand postures against complex backgrounds," *International Journal of Computer Vision*, vol. 101, Feb 2013.

[30] F. Ronchetti, F. Quiroga, L. Lanzarini, and C. Estrebou, "Handshape recognition for argentinian sign language using probsom," *Journal of Computer Science and Technology*, vol. 16, no. 1, pp. 1–5, 2016.

[31] D. Núñez Fernández, B. Kwolek, and S. Velastín, "Hand posture recognition using convolutional neural network," in *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*. Cham: Springer International Publishing, 2018, pp. 441–449.

[32] G. P. de Carvalho, "Handarch: A deep learning architecture for sign language hand configuration recognition," https://github.com/gabrielpeixoto-cvai/handarch, 2021, accessed: 2021-11-13.