Sign Language Recognition for Low Resource Languages Using Few Shot Learning

Chapter *in* Communications in Computer and Information Science · November 2023

DOI: 10.1007/978-981-99-8141-0_16

5 authors, including:

Sandareka Wickramanayake
University of Moratuwa
28 PUBLICATIONS **186** CITATIONS
SEE PROFILE

Thanuja D. Ambegoda
University of Moratuwa
26 PUBLICATIONS **142** CITATIONS
SEE PROFILE

Sign Language Recognition for Low Resource Languages Using Few Shot Learning

Kaveesh Charuka[0009−0008−8725−1000], Sandareka Wickramanayake*[0000−0003−0314−5988], Thanuja D. Ambegoda[0000−0002−5059−3479], Pasan Madhushan[0009−0007−8523−4154], and Dineth Wijesooriya[0009−0001−3996−7900]

Department of Computer Science & Engineering, University of Moratuwa, Sri Lanka
{kaveesh.18,sandarekaw,thanuja,pasan.18,dineth.18}@cse.mrt.ac.lk

Abstract. Sign Language Recognition (SLR) with machine learning is challenging due to the scarcity of data for most low-resource sign languages. Therefore, it is crucial to leverage a few-shot learning strategy for SLR. This research proposes a novel skeleton-based sign language recognition method based on the prototypical network [20] called ProtoSign. Furthermore, we contribute to the field by introducing the first publicly accessible dynamic word-level Sinhala Sign Language (SSL) video dataset comprising 1110 videos over 50 classes. To our knowledge, this is the first publicly available SSL dataset. Our method is evaluated using two low-resource language datasets, including our dataset. The experiments show the results in 95% confidence level for both 5-way and 10-way in 1-shot, 2-shot, and 5-shot settings.

Keywords: Sign Language Recognition · Few Shot Learning · Sign Language Recognition(SLR) · Prototypical Network

1 Introduction

Sign languages constitute the principal communication mechanism for approximately 5% of the hearing-impaired global population, as documented by the World Health Organization [1]. Each linguistic community worldwide utilizes a distinct sign language tailored to its cultural and regional context. Recognizing sign languages, especially those considered 'low-resource' like Sinhala Sign Language [26], presents unique challenges. Each sign language varies by region, with distinct gestures and meanings. This diversity, combined with the dynamic nature of sign languages and the limited availability of comprehensive datasets, makes Sign Language Recognition (SLR) a complex task.

In the field of SLR, numerous studies have been conducted, employing both vision-based [4,14] and contact-based approaches [2,10]. While these methods have shown promising results, they also face several limitations. Contact-based methods, for instance, require the user to wear specialized gloves equipped with

* Corresponding author

sensors, which can be intrusive and cost-prohibitive. On the other hand, vision-based methods often rely on extensive and rich datasets for training, which are not always available for low-resource sign languages. Few-shot learning is designed to build machine learning models that can understand new classes or tasks with minimal examples or training data [25]. Given the scarcity of sign language data, particularly for low-resource sign languages, few-shot learning could play a crucial role in SLR, enabling the development of robust models that can learn from fewer instances.

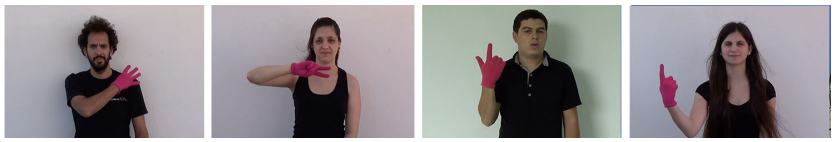
In this research, we introduce a new dynamic word-level Sinhala Sign Language dataset called SSL50 and propose a new framework called ProtoSign for low-resource SLR. ProtoSign comprises three core components: skeleton location extraction [4,22], a Transformer Encoder (TE) [23] equipped with a novel composite loss function, and the application of ProtoNet [20] for few-shot classification. The new SSL50 dataset comprises 50 classes with over 1000 sign videos, and it is the first publicly available dynamic Sinhala Sign Language dataset. The first step of our proposed ProtoSign is skeleton location extraction, the basis for data preprocessing. Next, we apply a TE, a deep learning model renowned for capturing intricate patterns and dependencies in data. Further, we introduce a composite loss function that combines triplet and classification loss, improving the whole approach's accuracy. Lastly, we employ ProtoNet for the few-shot classification task, which has state-of-the-art results. Experiment results on the newly introduced dataset and two publicly available datasets, LSA64 [15] and GSLL [21] demonstrate the effectiveness of the proposed ProtoSign framework for low-resource SSL.

2 Related Work

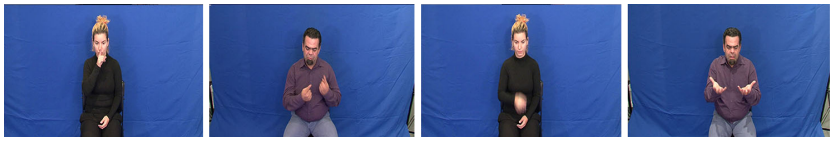
Vision-based Sign Language Recognition(SLR) methods use images or videos of hand gestures to recognize the signs. Vision-based SLR has dramatically improved with the advancement of deep learning. For example, Convolutional Neural Networks [9,13], Long Short-Term Memory Networks [6] and Transformers [17,5], have been used for input encoding in SLR.

However, using deep learning in low-resource SLR is challenging because of limited available data. A possible solution is to employ a few-shot learning approach. For instance, Santoro et al. [16] and Vinyals et al. [24] attempted to solve few-shot classification with end-to-end deep neural networks. Metric-based models are commonly used in meta-learning, one of the main types of few-shot learning approaches. Metric learning uses non-parametric techniques to model sample distance distributions, ensuring proximity between similar samples and maintaining distance between dissimilar ones. Core models embodying this principle are Matching Networks [24], Prototypical Networks [20], and Relation Network [19]. In their work on Prototypical Networks [20], Snell et al. expanded the concept from individual samples to a class-based metric. They grouped the descriptors of all samples from a specific class to establish class prototypes. These prototypes are then used for inference. Artem et al. in [7] introduced a meta-

(a) SSL50 dataset



(b) LSA64 dataset



(c) GSLL dataset

Fig. 1: Screenshots of sample videos from SSL50, LSA64 and GSLL Sign Language Recognition datasets.

learning-based network for American SLR, which acquires the ability to evaluate the similarity between pairs of feature vectors. Nevertheless, using metric-based models for low-resource SLR is not well-explored. In this paper, we explore using Prototypical Networks for low-resource SLR.

3 SSL50 Dataset

This paper introduces a diverse Sinhala Sign Language (SSL) dataset called SSL50 to facilitate recognizing dynamic SSL, addressing the lack of dynamic SSL datasets. SSL50 comprises over 1,000 videos, covering 50 classes of commonly used SSL words. We have ensured the SSL50 contains videos representing the most frequently used words by consulting with sign language professionals during the dataset creation and does not have any closely related signs.

Five signers, four female and one male, contributed videos to our dataset. All contributors were right-handed and between the ages of 21-35. Two signers learned SSL from their families, while the other three studied at educational institutions. To ensure a diverse and natural dataset, we conducted orientation

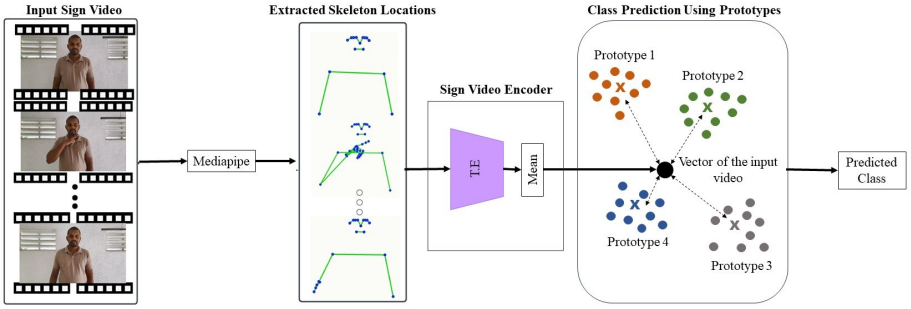


Fig. 2: Overview of the proposed ProtoSign framework

sessions with sign language professionals, providing them with an overview of our work. Each participant was requested to produce five sign videos per class using their mobile phones. We encouraged the signers to vary the backgrounds for each video of the same sign, aiming to capture a broader range of real-world scenarios rather than relying on lab-generated datasets. Figure 1a shows screenshots of sample videos in the SSL50 dataset, characterized by its inclusion of natural backgrounds. Conversely, Figure 1b and 1c showcase the screenshots of sample videos sourced from the LSA64 [15] and GSLL [21] datasets, respectively, created using static backgrounds. Hence, SSL50 better resembles real-world signing practices, incorporating natural variations and promoting inclusivity.

After collecting all the sign videos, we renamed each file in the format of `classId_signerId_variantId` (e.g., 001_002_001.mp4) to facilitate dataset annotation. Additionally, we converted all the videos into a uniform frame rate of 30fps. The file CSV file file contains comprehensive details about the dataset including signer details, gloss details (word, label, word in English), and video details (file name, signer ID, label, duration, fps, video width, video height). The dataset can be downloaded from here

4 Proposed ProtoSign Framework

The overview of the proposed ProtoSign framework is shown in Figure 2. The ProtoSign consists of three main steps. First, given the sign video, ProtoSign extracts the skeleton locations of the signer using the MediaPipe model [11]. Second, the extracted skeleton locations are sent to the transformer encoder to obtain a vector representing the input sign video. Finally, following ProtoNet [20], ProtoSign compares the obtained vector with the prototypes of different sign classes to determine the class of the given sign video.

4.1 Skeleton Locations Extraction

In the ProtoSign framework, the first step involves identifying the location of the skeleton in each video frame, including the face, body, and hand landmarks. The process of skeleton extraction is illustrated in Figure 3. To ensure accuracy, we

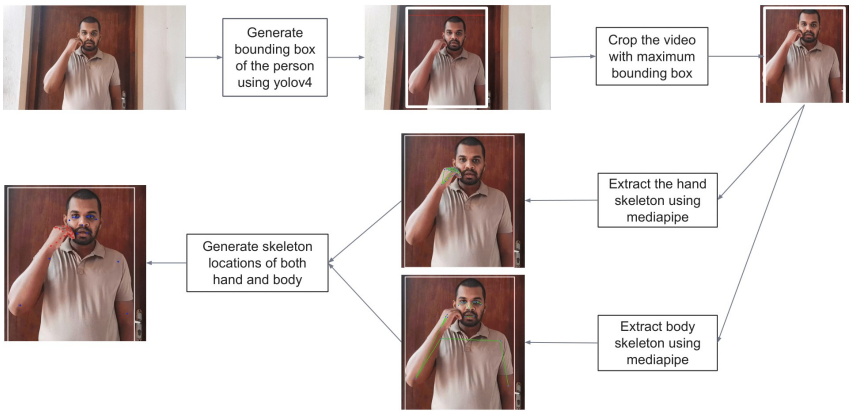


Fig. 3: The process pipeline for extracting and refining skeleton locations.

use the YOLOv4 framework [3] to detect the person in the video. This involves calculating the coordinates of the bounding box for each frame, which helps us determine the maximum bounding box that includes the person in each frame. Once we have the bounding box, we crop each frame using it. This step is essential in addressing the variability in the distance between the camera and the person. In real-world scenarios, we cannot expect the signer to be at a specific distance from the camera. By isolating the person, we eliminate the effects of distance and potential interference from other objects in the video background.

Next, we use a standard pose estimation algorithm from = MediaPipe [11] to extract skeleton locations. The algorithm utilizes two models: one for the hands and another for the whole body, resulting in a comprehensive extraction of skeleton locations. We then employ a refinement phase to remove any irrelevant locations. This method yields 57 skeleton locations, including 21 per hand, 4 for the body, and 11 for the face. Excluding the face locations, the remaining points represent the body's joints.

4.2 Sign Video Encoder

ProtoSign adapts Prototypical Networks (ProtoNet) [20] to deal with low-resource sign languages. ProtoNet makes the predictions based on prototype representations of each class. To create prototypes of classes, we develop a Sign Video Encoder based on a Transformer Encoder (TE). We use a modified version of the Transformer model [23] with a classification head as the final layer. This TE is first trained using a large sign language dataset, and in our implementation, we use the LSA64 dataset [15]. Next, we finetune the TE following the method used in [20] to create more discriminative prototypes for each sign class. In this section, we first describe how we pre-train the TE using a large sign language model and then detail how we fine-tune it.

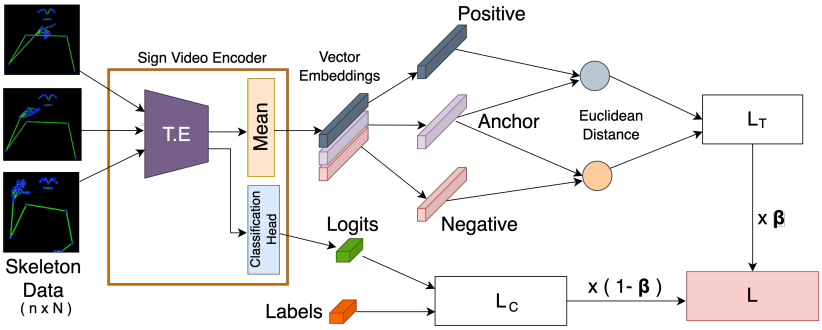


Fig. 4: Training phase of the Transformer Encoder used in ProtoSign

Given a sign video v of class y, the input to the transformer is a sequence of normalized skeleton locations x extracted in the previous step, $x \in \mathcal{R}^{n \times d}$ where n is the sequence length and d is the number of features. x is sent through positional encoding, self-attention, and feed-forward layers, mirroring the process in the original TE. Suppose the output of TE is $z \in \mathcal{R}^{n \times \bar{d}}$ where $\bar{d}$ is the output embedding size of the TE. We take the mean of z along the sequence length dimension to get the vector embedding of the input v_x:

$$v_x = \frac{1}{n} \sum_{z_i \in z} z_i \tag{1}$$

Finally, the classification head, another linear layer, is applied to v_x to get the predicted class $\bar{y}$.

We employ a composite objective function of triplet loss and classification loss to train the transformer encoder to generate discriminative vector representations for different sign classes. The classification loss forces the transformer encoder to learn discriminative features for each class. The triplet loss further enhances the discriminativeness of learned feature vectors by forcing higher intra-class and lower inter-class similarities. The training phase of the transformer encoder used in ProtoSign is shown in Figure 4.

Suppose a positive sample of x, which shares the same label, is denoted by x_p, and a negative sample with a different label is denoted by x_n. Let the output vector embedding of the transformer encoder for x_p and x_n be v_p and v_n, respectively. Then the triplet loss L_T is defined as

$$L_T = \|v_x - v_p\|_2^2 - \|v_x - v_n\|_2^2 + \alpha \tag{2}$$

Here, α is a margin enforced between positive and negative pairs. Through hyperparameter tuning, we set α to 2.0. We adopted an online triplet selection strategy in [18]. Although computationally intensive, the online strategy enhances robustness, expedites convergence, and performs better.

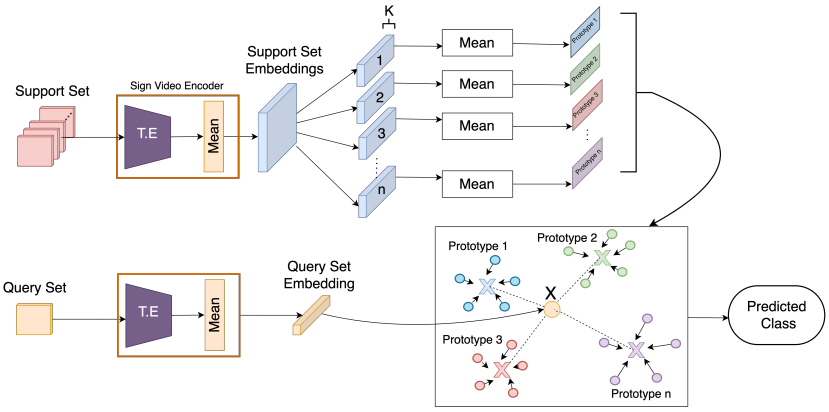


Fig. 5: The Prototypical Network Architecture.

We supplement the Triplet Loss with Classification Loss to provide comprehensive supervision to the TE during training. The Classification Loss aids the model in making accurate classification decisions for individual examples, thus facilitating the distinction between different classes. This, in turn, eases the task of Triplet Loss in refining the relative distances between classes, culminating in enhanced performance. The classification objective of the transformer encoder, L_C, is defined using the multi-class Classification Loss.

$$L_C = -\log P(\bar{y}|v_x) \tag{3}$$

The final objective function of the transformer encoder, L, is

$$L = \beta * L_T + (1 - \beta) * L_C \tag{4}$$

, where β is a hyper-parameter and in our experiment we set *beta* to be 0.9 through parameter tuning.

Next, following [20], we use episodic training to fine-tune the trained TE. Each episode consists of a support set of N samples $S = \{(x_1, y_1), ..., (x_N, y_N)\}$, where x_i is the sample video, $y_i \in \{1, ..., K\}$ is the corresponding label and K denotes the classes randomly selected from the training set comprising C classes ($|C| > |K|$). Suppose S_k is the set of samples belonging to class $k \in K$

We calculate the prototype for each class $k \in K$, p_k, by taking the average of the embeddings produced TE for S_k. Given a query point ($\bar{x} \in Q$, we obtain its vector representation from TE, $v_{\bar{x}}$ and calculate the Euclidean distance between $v_{\bar{x}}$ and each p_k. To determine the class of $\bar{x}$, we apply softmax over the calculated distances, producing a probability distribution over the classes. Figure 5 shows the fine-tuning phase of ProtoSign following the ProtoNet.

The same approach is employed during the inference, but data is randomly selected from the testing set instead.

Dataset	Num. of Classes	Num. of Signers	Average Num. of Videos per Class	Total Num. of Videos
LSA64	64	10	50	3200
SSL50	50	5	22	1110
GSLL	347	2	10	3464

Table 1: Datasets Summary

5 Experimental Study

5.1 Datasets

In addition to the newly introduced SSL50 dataset, our experiments use LSA64 [15] and GSLL [21] datasets.

- **LSA64:** The LSA64 dataset [15] is a comprehensive database developed for Argentinian Sign Language (LSA). The dataset comprises 3200 videos featuring 64 unique signs performed by ten non-expert, right-handed subjects five times each. The chosen signs, a mix of common verbs and nouns, were recorded in two separate sessions under distinct lighting conditions - outdoors with natural light and indoors with artificial light, providing variety in illumination across the videos.
- **GSLL:** The Greek Sign Language (GSLL) dataset [21] comprises 3,464 videos encapsulating a total of 161,050 frames, with each video representing one of 347 distinct sign classes. Two signers perform these signs, repeating each sign 5-17 times to offer variations.

A summary of the datasets used in our experiments is given in Table 1.

5.2 Implementation Details

We implement ProtoSign using PyTorch framework [12], and it is trained on an NVIDIA Tesla T4 GPU or an NVIDIA GeForce RTX 2040 GPU. We use the ADAM optimizer [8] for training.

We first train the TE in ProtoSign using the LSA64 dataset. We performed a grid search for hyperparameter tuning to optimize the overall performance. We set the number of attention heads to 16, batch size to 32 and learning rate to 0.002 and trained the model for 70 epochs. Further, we experimented with different values for β and set it to 0.9, which gives the best results.

Next, use ProtoSign for few-shot classification of signs by finetuning TE on each low-resource language dataset, SSL50 and GSLL. Here, for each 1-shot, 2-shot and 5-shot scenario for a given low-resource dataset, we create train, validation and test datasets separately. For example, let's consider the SSL50 dataset under the 1-shot scenario. We select two instances from each of the 50 classes for training and the remaining for testing. One sample is assigned as the support set of two chosen for training, whereas one is assigned as the query set.

Table 2: Few-shot classification accuracies of ProtoSign on SSL and GSLL datasets

	5 way			**10 way**		
	1 shot	**2 shot**	**5 shot**	**1 shot**	**2 shot**	**5 shot**
SSL50	61.2%	81.66%	93.08%	42.75%	69.24%	87.47%
GSLL	73.20%	84.38%	-	62.5%	79.65%	-

Table 3: Few-shot learning accuracies of ProtoSign for different scenarios on SSL50 dataset

Model	**5 way**			**10 way**		
	1 shot	**2 shot**	**5 shot**	**1 shot**	**2 shot**	**5 shot**
ProtoSign - CL	53.2%	74.3%	87.47%	37.7%	59.8%	84.06%
ProtoSign - TL	57.80%	77.47%	92.47%	41.45%	63.78%	86.73%
Matching Networks	**63.4%**	78.4%	88.45%	**46.87%**	69.09%	83.34%
VE + ProtoNet	41%	43.6%	41.22%	21%	22%	22.25%
ProtoSign	61.21%	**81.66%**	**93.08%**	42.75%	**69.24%**	**87.47%**

In each episode, we randomly select 5 (in the 5-way scenario) or 10 (in the 10-way scenario) classes from the training. Each epoch comprises 1000 such episodes. The code is available at `https://github.com/ProtoSign`

5.3 Experimental Results

Table 2 shows the few-shot classification accuracies of ProtoSign on SSL50 and GSLL datasets under different settings.

To show the effectiveness of our approach, we conducted four main ablation studies on the SSL50 and GSLL datasets. The final model we proposed integrates a combination of classification loss and triplet loss, including an extra classification layer nested within the Transformer Encoder. In the ablation studies, we evaluated the performance of our model by training only using Triplet Loss (TL) or classification loss (CL). Further, we considered replacing the ProtoNet with Matching Networks, allowing for a comparative analysis between these approaches. Furthermore, we assess the performance of ProtoSign when using a Video Encoder (VE) directly instead of skeleton location extractions followed by a Transformer Encoder. In our experiments, we use the r3d-18 video encoder for this evaluation.

Table 3 shows the performance of variants of ProtoSign for various conditions using the SSL50 dataset. We observe that ProtoSign has achieved the best performance among different variants for all the scenarios except the 1-shot scenario, whereas replacing ProtoNet with Matching Networks has yielded the best results in the 1-shot scenario. Table 4 displays the few-shot learning accuracies of variants of ProtoSign for the GSLL dataset. Here, our ProtoSign model surpasses all other variants in all the scenarios. The results of these ab-

Table 4: N-way k-shot accuracies for different scenarios on GSLL dataset

Model	5 way		10 way	
	1 shot	2 shot	1 shot	2 shot
ProtoSign - CL	66%	74.3%	37.7%	59.8%
ProtoSign - TL	70%	82.02%	60.10%	74.37%
Matching Networks	70.19%	82.78%	61.8%	75.61%
VE + ProtoNet	40.64%	46.45%	36.65%	43.12%
ProtoSign	**73.20%**	**84.38%**	**62.5%**	**79.65%**

lations studies demonstrate the effectiveness of each component of the proposed ProtoSign framework.

6 Conclusion

This paper presents a new architecture for few-shot learning in low-resource languages called ProtoSign, along with a dynamic dataset of Sinhala Sign Language at the word level called SSL50. The ProtoSign architecture consists of three main steps. Firstly, it extracts the skeleton locations of the signer from the sign video. Secondly, the extracted skeleton locations are sent to the transformer encoder to obtain a vector representing the input sign video. Finally, ProtoSign compares the obtained vector with prototypes of different sign classes to determine the class of the given sign video. The proposed framework's effectiveness is demonstrated through experimental results on two low-resource sign language datasets, the newly introduced SSL50 and GSLL.

References

1. Deafness and hearing loss (February 27, 2023), `https://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss`, accessed May 29, 2023
2. Amin, M.S., Rizvi, S.T.H., Hossain, M.M.: A comparative review on applications of different sensors for sign language recognition. Journal of Imaging **8**(4), 98 (2022)
3. Bochkovskiy, A., Wang, C.Y., Liao, H.Y.M.: Yolov4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934 (2020)
4. Boháček, M., Hrúz, M.: Sign pose-based transformer for word-level sign language recognition. In: Proceedings of the IEEE/CVF winter conference on applications of computer vision. pp. 182–191 (2022)
5. Camgoz, N.C., Koller, O., Hadfield, S., Bowden, R.: Multi-channel transformers for multi-articulatory sign language translation. In: Computer Vision–ECCV 2020 Workshops: Glasgow, UK, August 23–28, 2020, Proceedings, Part IV 16. pp. 301–319. Springer (2020)
6. 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. pp. 7361–7369 (2017)
7. Izutov, E.: Asl recognition with metric-learning based lightweight network. arXiv preprint arXiv:2004.05054 (2020)

8. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: andYann LeCun, Y.B. (ed.) Proceedings of the 3rd International Conference on Learning Representations, ICLR (2015)
9. Koller, O., Zargaran, O., Ney, H., Bowden, R.: Deep sign: Hybrid cnn-hmm for continuous sign language recognition. In: Proceedings of the British Machine Vision Conference 2016 (2016)
10. Lee, B.G., Lee, S.M.: Smart wearable hand device for sign language interpretation system with sensors fusion. IEEE Sensors Journal **18**(3), 1224–1232 (2017)
11. Lugaresi, C., Tang, J., Nash, H., McClanahan, C., Uboweja, E., Hays, M., Zhang, F., Chang, C.L., Yong, M.G., Lee, J., et al.: Mediapipe: A framework for building perception pipelines. arXiv preprint arXiv:1906.08172 (2019)
12. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (eds.) Proceedings of the Advances in Neural Information Processing Systems, pp. 8024–8035 (2019)
13. Rao, G.A., Syamala, K., Kishore, P., Sastry, A.: Deep convolutional neural networks for sign language recognition. In: 2018 conference on signal processing and communication engineering systems (SPACES). pp. 194–197. IEEE (2018)
14. Rastgoo, R., Kiani, K., Escalera, S.: Hand sign language recognition using multi-view hand skeleton. Expert Systems with Applications **150**, 113336 (2020)
15. Ronchetti, F., Quiroga, F., Estrebou, C.A., Lanzarini, L.C., Rosete, A.: Lsa64: An argentinian sign language dataset. In: XXII Congreso Argentino de Ciencias de la Computación (CACIC 2016). (2016)
16. Santoro, A., Bartunov, S., Botvinick, M., Wierstra, D., Lillicrap, T.: Meta-learning with memory-augmented neural networks. In: International conference on machine learning. pp. 1842–1850. PMLR (2016)
17. Saunders, B., Camgoz, N.C., Bowden, R.: Continuous 3d multi-channel sign language production via progressive transformers and mixture density networks. International journal of computer vision **129**(7), 2113–2135 (2021)
18. Schroff, F., Kalenichenko, D., Philbin, J.: Facenet: A unified embedding for face recognition and clustering. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 815–823 (2015)
19. Si, J., Zhang, H., Li, C.G., Kuen, J., Kong, X., Kot, A.C., Wang, G.: Dual attention matching network for context-aware feature sequence based person re-identification. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 5363–5372 (2018)
20. Snell, J., Swersky, K., Zemel, R.: Prototypical networks for few-shot learning. Advances in neural information processing systems **30** (2017)
21. Theodorakis, S., Pitsikalis, V., Maragos, P.: Dynamic–static unsupervised sequentiality, statistical subunits and lexicon for sign language recognition. Image and Vision Computing **32**(8), 533–549 (2014)
22. Tunga, A., Nuthalapati, S.V., Wachs, J.: Pose-based sign language recognition using gcn and bert. In: Proceedings of the IEEE/CVF winter conference on applications of computer vision. pp. 31–40 (2021)
23. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)

24. Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al.: Matching networks for one shot learning. Advances in neural information processing systems **29** (2016)
25. Wang, Y., Yao, Q., Kwok, J.T., Ni, L.M.: Generalizing from a few examples: A survey on few-shot learning. ACM computing surveys (csur) **53**(3), 1–34 (2020)
26. Weerasooriya, A.A., Ambegoda, T.D.: Sinhala fingerspelling sign language recognition with computer vision. In: 2022 Moratuwa Engineering Research Conference (MERCon). pp. 1–6. IEEE (2022)