👐 OpenHands: Making Sign Language Recognition Accessible with Pose-based Pretrained Models across Languages

Prem Selvaraj[*1], **Gokul NC**[*1,2], **Pratyush Kumar**[1,2,3], **Mitesh Khapra**[1,2]
[1]AI4Bharat, [2]IIT-Madras, [3]Microsoft Research
prem@ai4bharat.org, gokulnc@ai4bharat.org, pratyush@cse.iitm.ac.in, miteshk@cse.iitm.ac.in

Abstract

AI technologies for Natural Languages have made tremendous progress recently. However, commensurate progress has not been made on Sign Languages, in particular, in recognizing signs as individual words or as complete sentences. We introduce 👐 OpenHands[1], a library where we take four key ideas from the NLP community for low-resource languages and apply them to sign languages for word-level recognition. First, we propose using pose extracted through pretrained models as the standard modality of data in this work to reduce training time and enable efficient inference, and we release standardized pose datasets for different existing sign language datasets. Second, we train and release checkpoints of 4 pose-based isolated sign language recognition models across 6 languages (American, Argentinian, Chinese, Greek, Indian, and Turkish), providing baselines and ready checkpoints for deployment. Third, to address the lack of labelled data, we propose self-supervised pretraining on unlabelled data. We curate and release the largest pose-based pretraining dataset on Indian Sign Language (Indian-SL). Fourth, we compare different pretraining strategies and for the first time establish that pretraining is effective for sign language recognition by demonstrating (a) improved fine-tuning performance especially in low-resource settings, and (b) high crosslingual transfer from Indian-SL to few other sign languages. We open-source all models and datasets in 👐 OpenHands with a hope that it makes research in sign languages reproducible and more accessible.

1 Introduction

According to the World Federation of the Deaf, there are approximately 72 million Deaf people worldwide. More than 80% of them live in developing countries. Collectively, they use more than 300 different sign languages varying across different nations (UN, 2021). Loss of hearing severely limits the ability of the Deaf to communicate and thereby adversely impacts their quality of life. In the current increasingly digital world, systems to ease digital communication between Deaf and hearing people are important accessibility aids. AI has a crucial role to play in enabling this accessibility with automated tools for Sign Language Recognition (SLR). Specifically, transcription of sign language as complete sentences is referred to as Continuous Sign Language Recognition (CSLR), while recognition of individual signs is referred to as Isolated Sign Language Recognition (ISLR). There have been various efforts to build datasets and models for ISLR and CLSR tasks (Adaloglou et al., 2021; Koller, 2020). But these results are often concentrated on a few sign languages (such as the American Sign Language) and are reported across different research communities with few standardized baselines. When compared against text- and speech-based NLP research, the progress in AI research for sign languages is significantly lagging. This lag has been recently brought to notice of the wider NLP community (Yin et al., 2021).

For most sign languages across the world, the amount of labelled data is very low and hence they can be considered *low-resource languages*. In the NLP literature, many successful templates have been proposed for such low-resource languages. In this work, we adopt and combine many of these ideas from NLP to sign language research. We implement these ideas and release several datasets and models in an open-source library 👐 OpenHands with the following key contributions:

1. Standardizing on pose as the modality: We consider using pose-extractor as an encoder, which processes raw RGB videos and extracts the framewise coordinates for few keypoints. Pose-extractors are useful across sign languages and also other

[*] Equal contribution.

[1] https://github.com/AI4Bharat/OpenHands

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 2114 - 2133
May 22-27, 2022 ©2022 Association for Computational Linguistics

tasks such as action recognition (Yan et al., 2018; Liu et al., 2020), and can be trained to high accuracy. Further, as we report, pose as a modality makes both training and inference for SLR tasks efficient. We release pose-based versions of existing datasets for 5 sign languages: American, Argentinian, Greek, Indian, and Turkish.

2. Standardized comparison of models across languages: The progress in NLP has been earmarked by the release of standard datasets, including multilingual datasets like XGLUE (Liang et al., 2020), on which various models are compared. As a step towards such standardization for ISLR, we train 4 different models spanning sequence models (LSTM and Transformer) and graph-based models (ST-GCN and SL-GCN) on 7 different datasets across 6 sign languages mentioned in Table 1, and compare them against models proposed in the literature. We release all 28 trained models along with scripts for efficient deployment which demonstrably achieve real-time performance on CPUs and GPUs.

3. Corpus for self-supervised training: A defining success in NLP has been the use of self-supervised training, for instance masked-language modelling (Devlin et al., 2018), on large corpora of natural language text. To apply this idea to SLR, we need similarly large corpora of sign language data. To this end, we curate 1,129 hours of video data on Indian Sign Language. We pre-process these videos with a custom pipeline and extract keypoints for all frames. We release this corpus which is the first such large-scale sign language corpus for self-supervised training.

4. Effectiveness of self-supervised training: Self-supervised training has been demonstrated to be effective for NLP: Pretrained models require small amounts of fine-tuning data (Devlin et al., 2018; Baevski et al., 2020) and multilingual pretraining allows crosslingual generalization (Hu et al., 2020b). To apply this for SLR, we evaluate multiple strategies for self-supervised pretraining of ISLR models and identify those that are effective. With the identified pretraining strategies, we demonstrate the significance of pretraining by showing improved fine-tuning performance, especially in very low-resource settings and also show high crosslingual transfer from Indian SL to other sign languages. This is the first and successful attempt that establishes the effectiveness of self-supervised learning in SLR. We release the pretrained model and the fine-tuned models for 4 different sign languages.

Through these datasets, models, and experiments we make several observations. First, in comparing standardized models across different sign languages, we find that graph-based models working on pose modality define state-of-the-art results on most sign languages. RNN-based models lag on accuracy but are significantly faster and thus appropriate for constrained devices. Second, we establish that self-supervised pretraining helps as it improves on equivalent models trained from scratch on labelled ISLR data. The performance gap is particularly high if the labelled data contains fewer samples per label, i.e., for the many sign languages which have limited resources the value of self-supervised pretraining is particularly high. Third, we establish that self-supervision in one sign language (Indian SL) can be crosslingually transferred to improve SLR on other sign languages (American, Chinese, and Argentinian). This is particularly encouraging for the long tail of over 300 sign languages that are used across the globe. Fourth, we establish that for real-time applications, pose-based modality is preferable over other modalities such as RGB, use of depth sensors, etc. due to reduced infrastructure requirements (only camera), and higher efficiency in self-supervised pretraining, fine-tuning on ISLR, and inference. We believe such standardization can help accelerate dataset collection and model benchmarking. Fifth, we observe that the trained checkpoints of the pose-based models can be directly integrated with pose estimation models to create a pipeline that can provide real-time inference even on CPUs. Such a pipeline can enable the deployment of these models in real-time video conferencing tools, perhaps even on smartphones.

As mentioned all datasets and models are released with permissible licenses in 👐 `OpenHands` with the intention to make SLR research more accessible and standardized. We hope that others contribute datasets and models to the library, especially representing the diversity of sign languages used across the globe.

The rest of the paper is organized as follows. In section 2 we present a brief overview of the existing work. In section 3 we describe our efforts in standardizing datasets and models across six different sign languages. In section 4 we explain our pretraining corpus and strategies for self-supervised learning and detail results that establish its effectiveness. In section 5 we describe in brief the functionalities of the 👐 `OpenHands` library. In section 6, we summarize our work and also list potential follow-up work.

2 Background and Related Work

Significant progress has been made in Isolated Sign Language Recognition (ISLR) due to the release of datasets (Li et al., 2020; Sincan and Keles, 2020; Chai et al., 2014; Huang et al., 2019) and recent deep learning architectures (Adaloglou et al., 2021). This section reviews this work, with a focus on pose-based models.

2.1 Sign Language

A sign language (SL) is the visual language used by the Deaf and hard-of-hearing (DHH) individuals (and also by those who comunnicate with them), which involves usage of various bodily actions, like hand gestures and facial expressions, called signs to communicate. A sequence of signs constitutes a phrase or sentence in a SL. The signs can be transcribed into sign-words of any specific spoken language usually written completely in capital letters. Each such sign-word is technically called as a gloss and is the standardized basic atomic token of an SL transcript (Schembri and Crasborn, 2010). It is be noted that there is not (always) one-to-one relationships between glosses and spoken language words.

The task of converting each visual sign communicated by a signer into a gloss is called isolated sign language recognition (ISLR). The task of converting a continuous sequence of visual signs into serialized glosses is referred as continuous sign language recognition (CSLR). CSLR can either be modeled as an end-to-end task, or as a combination of sign language segmentation and ISLR. The task of converting signs into spoken language text is referred as sign language translation (SLT), which can again either be end-to-end or a combination of CLSR and gloss-sequence to spoken phrase converter.

In terms of real-world applications, eventhough CSLR is more practically useful than ISLR, it does not still undermine the value in studying and implementing ISLR. The applications of ISLR include building sign spotting systems (Albanie et al., 2020), building alignment networks (Albanie et al., 2021) to aid in building CSLR datasets (or evaluate CSLR output), building CSLR systems on top of an automatic SL segmentation model (Farag and Brock, 2019) which identifies the frame boundaries for signs in videos to divide them into approximate meaningful segments (glosses), etc.

Although SL content is predominantly recorded as RGB (color) videos, it can also be captured using various other modalities like depth maps or point cloud, finger gestures recorded using sensors, skeleton representation of the signer, etc. In this work, we focus on ISLR using pose-skeleton modality. A pose representation, extracted using pose estimation models, provides the spatial coordinates at which the joints (such as elbows and knees), called keypoints, are located in a field or video. This pose information can be represented as a connected graph with nodes representing keypoints and edges may be constructed across nodes to approximately represent the human skeleton.

2.2 Models for ISLR

Initial methods for SLR focused on hand gestures from either video frames (Reshna et al., 2020) or sensor data such as from smart gloves (Fels and Hinton, 1993). Given that such sensors are not commonplace and that body posture and face expressions are also of non-trivial importance for understanding signs (Hu et al., 2020a), convolutional network based models have been used for SLR (Rao et al., 2018).

ISLR can be considered as a multiclass classification task and generally accuracy metric is used the to evaluate the performance of the models. The ISLR task is related to the more widely studied action recognition task (Zhu et al., 2020). Like in action recognition task, highly accurate pose recognition models like OpenPose (Cao et al., 2018) and MediaPipe Holistic (Grishchenko and Bazarevsky, 2020) are being used for ISLR models (Li et al., 2020; Ko et al., 2018), where frame-wise keypoints are the inputs. Although RGB-based models may outperform pose-based models (Li et al., 2020) narrowly, pose-based models have far fewer parameters and are more efficient for deployment if used with very-fast pose estimation pipelines like MediaPipe BlazePose. In this work, we focus on lightweight pose-based ISLR which encode the pose frames and classify the pose using specific decoders. We briefly discuss the two broad types of such models: sequence-based and graph-based.

Sequence-based models process data sequentially along time either on one or both directions. Initially, RNNs were used for pose-based action recognition to learn from temporal features (Du et al., 2015; Zhang et al., 2017; Si et al., 2018). Specifically, sequence of pose frames are input to GRU or LSTM layers, and the output from the final timestep is used for classification. Transformer architectures with encoder-only models like BERT (Vaswani et al., 2017) have also been studied for pose-based ISLR models (De Coster et al., 2020). The input is a sequence of pose frames along with positional embeddings. A special [CLS] token is prepended to the sequence, whose final embedding is used for classification.

Graph convolution networks (Kipf and Welling, 2017), which are good at modeling graph data have been used for skeleton action recognition to achieve state-of-the-art results, by considering human skeleton sequences as spatio-temporal graphs (Cheng et al., 2020a; Liu et al., 2020). Spatial-Temporal GCN (ST-GCN) uses human body joint connections for spatial connections and temporal connections across frames to construct a 3d graph, which is processed by a combination of spatial graph convolutions and temporal convolutions to efficiently model the spatio-temporal data (Lin et al., 2020). Many architectural improvements have been proposed over ST-GCN for skeleton action recognition (Zhang et al., 2020; Shi et al., 2019b,a; Cheng et al., 2020b,a; Liu et al., 2020). MS-AAGCN (Shi et al., 2020) uses attention to adaptively learn the graph topology and also proposes STC-attention module to adaptively weight joints, frames and channels. Decoupled GCN (Cheng et al., 2020a) improves the capacity of ST-GCN without adding additional computations and also proposes attention guided drop mechanism called DropGraph as a regularization technique. Sign-Language GCN (SL-GCN) (Jiang et al., 2021) combines STC-attention with Decoupled-GCN and extends it to ISLR achieving state-of-the-art results.

2.3 Pretraining strategies

Although there are works which use an already trained classifier (on a large dataset) to finetune for smaller datasets and obtain state-of-the-art results in the latter (Albanie et al., 2020), there are currently no works which study the value of pretraining on openly available unlabelled data. On this front, we now survey three broad classes of self-supervised pretraining strategies that we reckon could be applied to SLR.

Masking-based pretraining: In NLP, masked language modelling is a pretraining technique where randomly masked tokens in the input are predicted. This approach has been explored for action recognition (Cheng et al., 2021), where certain frames are masked and a regression task estimates coordinates of keypoints. In addition, a direction loss is also proposed to classify the quadrant where the motion vector lies.

Contrastive-learning based: Contrastive learning is used to learn feature representations of the input to maximize the agreement between augmented views of the data (Gao et al., 2021; Linguo et al., 2021). For positive examples, different augmentations of the same data item are used, while for negative samples randomly-chosen data items usually from a few last training batches are used. A variant of contrastive loss called InfoNCE (van den Oord et al., 2018) is used to minimize the distance between positive samples.

Predictive Coding: Predictive Coding aims to learn data representation by continuously correcting its predictions about data in future timesteps given data in certain input timesteps. Specifically, the training objective is to pick the future timestep's representation from other negative samples which are usually picked from recent previous timesteps of the same video. Similar to contrastive learning, a loss function based on NCE is used (Mikolov et al., 2013; van den Oord et al., 2018). This technique was explored for action recognition in a model called Dense Predictive Coding (DPC) (Han et al., 2019). Instead of predicting at the frame-level, DPC introduces coarse-prediction at the scale of non-overlapping windows.

3 Standardized Pose-based ISLR Models across Sign Languages

In this section, we describe our efforts to curate standardized pose-based datasets across multiple sign languages and benchmark multiple ISLR models on them.

3.1 ISLR Datasets

Multiple datasets have been created for the ISLR task across sign languages. However, the amount of data significantly varies across different sign languages, with American and Chinese having the largest datasets currently. With a view to cover a diverse set of languages, we study 7 different

Dataset	Language	Vocab	Signers	Videos	Hrs	Data
AUTSL (Sincan and Keles, 2020)	Turkish	226	43	38,336	20.5	RGBD
CSL (Huang et al., 2019)	Chinese	500	50	125,000	108.84	RGBD
DEVISIGN (Chai et al., 2014)	Chinese	2000	30	24,000	21.87	RGBD
GSL (Adaloglou et al., 2021)	Greek	310	7	40,785	6.44	RGBD
INCLUDE (Sridhar et al., 2020)	Indian	263	7	4,287	3.57	RGB
LSA64 (Ronchetti et al., 2016)	Argentinian	64	10	3,200	1.90	RGB
WLASL (Li et al., 2020)	American	2000	119	21,083	14	RGB

Table 1: The diverse set of existing ISLR datasets which we study in this work through pose-based models

datasets across 6 sign languages as summarised in Table 1. For each of these datasets, we generate pose-based data using the Mediapipe pose-estimation pipeline (Grishchenko and Bazarevsky, 2020), which enables real-time inference in comparison with models such as OpenPose (Cao et al., 2018). Mediapipe, in our chosen Holistic mode, returns 3d coordinates for 75 keypoints (excluding the face mesh). Out of these, we select only 27 sparse 2d keypoints which convey maximum information, covering upper-body, hands and face. Thus, each input video is encoded into a vector of size $F \times K \times D$, where F is the number of frames in the video, K is the number of keypoints (27 in our case), and D is the number of coordinates (2 in our case). In addition, we perform several normalizations and augmentations explained in Section 5.

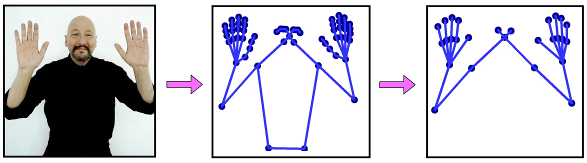

Figure 1: Illustration for RGB frame to pose keypoints conversion. The center skeleton shows the upper portion of the 75 keypoints returned by MediaPipe, from which we choose only 27 points as shown in right.

3.2 Standardized ISLR Models

On the 7 different datasets we consider, different existing ISLR models have been trained which are mentioned in Table 2 which produce their current state-of-the-art results. For INCLUDE dataset, an XGBoost model is used (Sridhar et al., 2020) with direct input as 135 pose-keypoints obtained using OpenPose. For AUTSL, SL-GCN is used (Jiang et al., 2021) with 27 chosen keypoints as input from HRNet pose estimation model. For GSL, the corresponding model (Parelli et al., 2020) is an attention-based encoder-decoder with 3D hand pose and 2D body pose as input. For WLASL, Temporal-GCN is used (Li et al., 2020) by passing 55 chosen keypoints from OpenPose. For LSA64, 33 chosen keypoints from OpenPose are used as input to an LSTM decoder (Konstantinidis et al., 2018). For DEVISIGN, RGB features are used (Yin et al., 2016) and the task is approached using a clustering-based classic technique called Iterative Reference Driven Metric Learning. For CSL dataset, an I3D CNN is used as encoder with input as RGBD frames and BiLSTM as decoder (Adaloglou et al., 2021). For DEVISIGN_L and CSL datasets, we report RGB model results in the table as there are no existing works using pose-based models.

The differences in the above models make it difficult to compare them on effectiveness, especially across diverse datasets. To enable standardized comparison of models, we train pose-based ISLR models on all datasets with similar training setups for consistent benchmarking. These models belong to two groups: sequence-based models and graph-based models. For sequence-based models we consider RNN and Transformer based architectures. For the **RNN model**, we use a 4-layered bidirectional LSTM of hidden layer dimension 128 which takes as input the framewise pose-representation of 27 keypoints with 2 coordinates each, i.e., a vector of 54 points per frame. We also use a temporal attention layer to weight the most effective frames for classification. For the **Transformer model**, we use a BERT-based architecture consisting of 5 Transformer-encoder layers with 6 attention heads and hidden dimension size 128, with a maximum sequence length of 256. For the graph-based models we consider ST-GCN (Yan et al., 2018) and SL-GCN (Jiang et al., 2021) models as discussed in section 2. For **ST-GCN model**, we use 10 spatio-

Dataset	State-of-the-art model		Model available in 👐 OpenHands			
	Model (Params)	Accuracy	LSTM	BERT	ST-GCN	SL-GCN
AUTSL	Pose-SL-GCN[2] (4.9M)	**95.02**	77.4	81.0	90.4	91.9
CSL	RGBD-I3D (27M)	95.68	75.1	88.8	94.2	**94.8**
DEVISIGN_L	RGB-iRDML	56.85	37.6	48.9	55.8	**63.9**
GSL	Pose-Attention (2.1M)	83.42	86.6	89.5	93.5	**95.4**
INCLUDE	Pose-XGBoost	63.10	86.3	90.4	91.2	**93.5**
LSA64	Pose-LSTM (1.9M)	93.91	90.2	92.5	94.7	**97.8**
WLASL2000	Pose-TGCN (5.2M)	23.65	20.6	23.2	21.4	**30.6**
	Average accuracy	→	67.7	73.5	77.3	81.1

Table 2: Accuracy of different models across datasets. The results in bold are the SOTA pose models.

temporal GCN layers with the spatial dimension of the graph consisting of the 27 keypoints. For the **SL-GCN model**, we use again 10 SL-GCN blocks with the same graph structure and hyperparameters as the ST-GCN model.

3.3 Experimental Setup and Results

We train 4 models - LSTM, BERT, ST-GCN, and SL-GCN - for each of the 7 datasets. We use PyTorch Lightning to implement the data processing and training pipelines. We use Adam Optimizer to train all the models. We search for optimal hyperparameters using grid search to find the best hyperparams for each model on a standard dataset, and report the best configuration per model. For the LSTM model, we set the batch size as 32 and initial learning rate (LR) as $5e-3$, while for BERT, we set a batch size 64, and LR of $1e-4$. For ST-GCN and SL-GCN, we use a batch size of 32 and LR of 0.001. We train all our models on a NVIDIA Tesla V100 GPU. Also for all datasets, we only train on the train-sets given and we use valid-sets to do early stopping, whereas some works (like AUTSL) train on combination of train-set and val-set to report the final test accuracy. We run each experiment around 3 times, and report the best accuracy, eventhough we do not see significant difference in accuracies across the runs. All trained models and the training configurations are open-sourced in 👐 OpenHands.

Accuracy We report the obtained test-set accuracy of detecting individual signs, for each model against each dataset in Table 2. On all datasets, graph-based models report the state-of-the-art results using pose data. Except for AUTSL[2], on 6 of the 7 datasets, models we train improve upon the accuracy reported in the existing papers sometimes significantly (e.g., over 10% on GSL). These uniform results across a diverse set of SLs confirm that graph-based models on pose modality data define the SOTA.

In summary, the standardized benchmarking of multiple models in terms of accuracy on datasets and, measurements of latency on devices (explained in appendix) informs model selection. Making the trade-off between accuracy and latency, we use the ST-GCN model for the pretrained model we discuss later. Our choice is also informed by the cost of the training step: The more accurate SL-GCN model takes 4× longer to train than ST-GCN.

4 Self-Supervised Learning for ISLR

In this section, we describe our efforts in building the largest corpus for self-supervised pretraining and our experiments in different pretraining strategies.

4.1 Indian SL Corpus for Self-supervised pretraining

Channel	Hours	Domain
NewzHook	615	News
MBM Vadodara	225	News
ISH-News	145	News
NIOS	115	Educational
SIGN Library	29	Educational
Total	**1129**	

Table 3: Source-wise statistics of the processed self-supervised dataset on Indian-SL

Large text corpora such as BookCorpus,

[2] SOTA AUTSL model is trained on high quality pose data from HRNet model with more keypoints.

Wikipedia dumps, OSCAR, etc. have enabled pretraining of large language models. Although there are large amounts of raw sign language videos available on the internet, no existing work has studied how such large volumes of open unlabelled data can be collected and used for SLR tasks. To address this, we create a corpus of Indian SL data by curating videos, pre-process the videos, and release a standardized pose-based dataset compatible with the models discussed in the previous section.

We manually search for freely available major sources of Indian SL videos. We restrict our search to a single sign language so as to study the effect of pretraining on same language and crosslingual ISLR tasks. We sort the sources by the number of hours of videos and choose the top 5 sources for download. All of these 5 sources, as listed in Table 3 are YouTube channels, totalling over 1,500 hours before preprocessing.

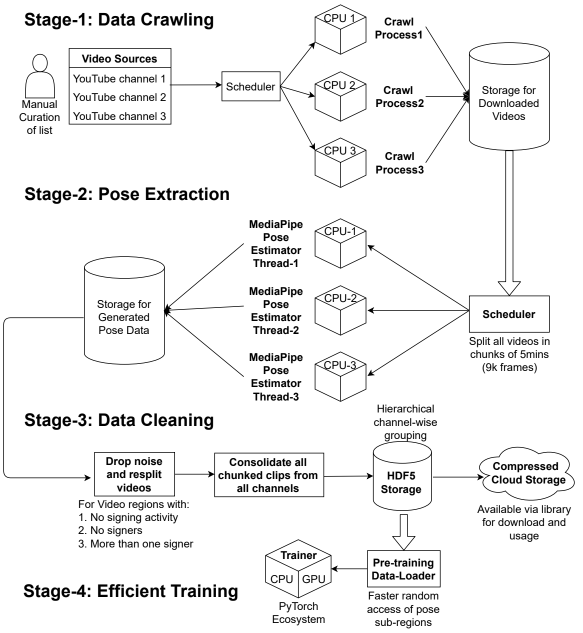
Stage-1: Data Crawling
Manual Curation of list
Video Sources
YouTube channel 1
YouTube channel 2
YouTube channel 3
Scheduler
CPU 1
CPU 2
CPU 3
Crawl Process1
Crawl Process2
Crawl Process3
Storage for Downloaded Videos
Stage-2: Pose Extraction
MediaPipe Pose Estimator Thread-1
MediaPipe Pose Estimator Thread-2
MediaPipe Pose Estimator Thread-3
CPU-1
CPU-2
CPU-3
Storage for Generated Pose Data
Scheduler
Split all videos in chunks of 5mins (9k frames)
Stage-3: Data Cleaning
Hierarchical channel-wise grouping
Drop noise and resplit videos
Consolidate all chunked clips from all channels
HDF5 Storage
Compressed Cloud Storage
For Video regions with:
1. No signing activity
2. No signers
3. More than one signer
Available via library for download and usage
Stage-4: Efficient Training
Trainer
CPU
GPU
PyTorch Ecosystem
Pre-training Data-Loader
Faster random access of pose sub-regions

Figure 2: Pipeline used to collect and process Indian SL corpus for self-supervised pretraining

We pass these videos through a processing pipeline as described in Figure 2. We initially dump the pose data for all videos, then process them to remove those which are noisy or contain either no person or more than 1 person. This resulted in 1,129 hours of Indian SL data, as detailed source-wise in Table 3. This is significantly larger than all the training sets in the datasets we studied which is on average 177 hours. We pass these videos through MediaPipe to obtain pose information as described earlier, i.e., 75 keypoints per frame. The resultant Indian SL corpus has more than 100 million pose frames. We convert this to the HDF5 format to enable efficient random access, as is required for training. We open-source this corpus of about 250 GB which is available in 👐 `OpenHands`.

4.2 Pretraining Setup and Experiments

We explore the three major pretraining strategies as described in Section 2.3 and explain how and why certain self-supervised settings are effective for ISLR. We pretrain on randomly sampled consecutive input sequences of length 60-120 frames (approximating 2-4 secs with 30fps videos). After pretraining, we fine-tune the models on the respective ISLR dataset with an added classification head.

4.2.1 Masking-based pretraining

We follow the same hyperparameter settings as described in Motion-Transformer (Cheng et al., 2021), to pretrain a BERT-based model with random masking of 40% of the input frames. When using only the regression loss, we find that pretraining learns to reduce the loss as shown in appendix. However, when fine-tuned on the INCLUDE dataset, we see no major contribution of the pretrained model to increasing the accuracy as shown in Table 4. We posit that while pretraining was able to approximate interpolation for the masked frames based on the surrounding context, it did not learn higher-order features relevant across individual signs. We also experiment with different masking ratios (20% and 30%) as well as different length of random contiguous masking spans (randomly selected between 2-10), and obtain similar results.

4.2.2 Contrastive-learning based

Inspired from the work by Gao et al. (2021), we consider Shear, Scaling and Rotation augmentations to generate the 2 augmented copies of the input pose sequence and we pretrain the model and observe that it converges on reducing the InfoNCE loss (see appendix for plot). We then fine-tune on INCLUDE and again did not observe any gain over the baseline of training from scratch as seen in Table 4. To understand this, we analyzed the embeddings of data from the pretrained model and observed two facts: (a) Embeddings of different augmentations of a video clip are similar indicating successful pretraining, but (b) Embeddings of different videos from the INCLUDE dataset do not show any clustering based on the class (see visualization in appendix). Again, we posit that pretraining did not learn higher order semantics that could be helpful for ISLR.

4.2.3 Predictive-coding based

Our architecture is inspired from Dense Predictive Coding (Han et al., 2019), but using pose modality. The architecture is represented in Figure 3. The pose frames from a video clip will be partitioned into multiple non-overlapping windows with equal number of frames in each window. The encoder f takes each window of pose keypoints as input and embeds into the hidden space z. We use ST-GCN as the encoder. The ST-GCN encoder embeds each input window x_i, and the direct output is average pooled across the spatial and temporal dimensions to obtain the output embedding z_i for each window. The embeddings are then fed to a Gated Recurrent Unit (GRU) as a temporal sequence and the future timesteps $\hat{z}_i$ are predicted sequentially using the past timestep representations from GRU, with an affine transform layer ϕ. We use 4 windows of data as input to predict the embeddings of the next 3 windows, each window spanning 10 frames, which we empirically found to be the best setting. For pretraining, we used a batch size of 128 and for finetuning, we used a batch size of 64. For both pretraining and finetuning, we used Adam optimizer with an initial learning rate of 1e-3. The pretraining was done for 200k iterations on a NVIDIA V100 GPU, taking around 26 hours (on Microsoft platform's Azure NC6s_v3 machine).

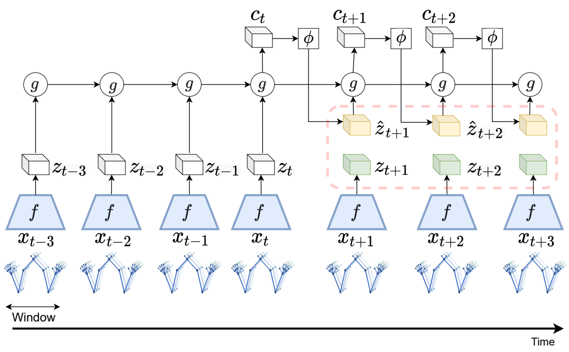

Figure 3: Model architecture for DPC pretraining

Upon fine-tuning on INCLUDE, DPC provides a significant improvement of 3.5% over the baseline. We include a plot comparing the validation accuracy between baseline and finetuned model in appendix. We posit that Sign Language DPC (SL-DPC) is successful, while previous methods were not, as it learns coarse-grained representations across multiple frames and thereby captures motion semantics of actions in SL.

To the best of our knowledge, this is the first comparison of pretraining strategies for SLR.

Training of ST-GCN	Accuracy
No pretraining + Fine-tune	91.2
Masked-based + Fine-tune	91.3
Contrastive learning + Fine-tune	90.8
Predictive-coding + Fine-tune	94.7

Table 4: Effectiveness of pretraining strategies as measured on ISLR accuracy on INCLUDE

4.3 Evaluation on low-resource and crosslingual settings

We demonstrated that DPC-based pretraining is effective. We now analyze the effectiveness of such pretraining in two constrained settings - (a) when fine-tuning datasets are small, and (b) when fine-tuning on sign languages different from the sign language used for pretraining. The former captures in-language generalization while the latter crosslingual generalization.

Dataset	Samples/class	STGCN	SLDPC
INCLUDE (Indian)	Full (Avg. 17)	91.2	94.7
	10	79.7	86.27
	5	45	57.35
	3	15.2	35.42
WLASL2k (American)	Full (Avg. 10)	21.4	27.4
	5	3.1	5.74
	3	1.6	2.78
DEVISign_L (Chinese)	Full (8)	55.8	59.5
	5	33.0	40.26
	3	8.46	18.65
LSA64 (Argentinian)	Full (50)	94.7	96.25
	5	64.7	75.32
	3	39.7	57.19

Table 5: Effectiveness of pretraining for in-language (first row) and crosslingual transfer (last three rows)

4.3.1 In-language generalization

The INCLUDE dataset contains an average of 17 samples per class. For this setting, we observed a gain of 3.5% with DPC-based pretraining over training from scratch. How does this performance boost change when we have fewer samples per class? We present results for 10, 5, and 3 samples per class in Table 5. We observe that as the number of labels decreases the performance boost due to pretraining is higher indicating effective in-language generalization.

4.3.2 Crosslingual transfer

Does the pretraining on Indian sign language provide a performance boost when fine-tuning on other sign languages? We study this for 3 different sign languages - American, Chinese, and Argentinian - and report results in Table 5. We see that crosslingual transfer is effective leading to gains of about 6%, 4%, and 2% on the three datasets, similar to the 3% gain on in-language accuracy. The increase in accuracy varies with datasets - For Argentianian and Indian datasets which already have 90+% accuracy, there are small improvements. However, WLASL which is scraped from web and has a lot more variations, sees a much higher improvement due to pretraining. Further, we also observe that these gains extend to low-resource settings of fewer labels per sign. For instance on Argentinian SL, with 3 labels, pretraining on Indian SL given an improvement of about 18% in accuracy. To the best of our knowledge this is the first successful demonstration of crosslingual transfer in ISLR.

In summary, we discussed different pretraining strategies and found that only SL-DPC learns semantically relevant higher-order features. With DPC-based pretraining we demonstrated both in-language and crosslingual transfer.

5 The 👐 OpenHands library

As mentioned in the main paper, we open-source all our contributions through the 👐 `OpenHands` library. This includes the pose-based datasets for 5 SLs, 4 ISLR models trained on 7 datasets, the pretraining corpus on Indian SL with over 1,100 hours of pose data, pretrained models on this corpus using self-supervised learning, and models fine-tuned for 4 different SLs on top of the pretrained model. We also provide scripts for efficient deployment using MediaPipe pose estimation and our trained ISLR models.

In addition, the library provides utilities that are helpful specifically for pose-based data. This includes methods to normalize keypoints by width and height of frames, to normalize all of the pose data to be in the same scale and reference coordinate system by using a constant feature of body as reference, and to fill missing keypoints. The library also includes utilities to create data augmentations such as *ShearTransform* to displace the joints in a random direction, *RotatationTransform* to simulate the viewpoint changes of the camera, *ScaleTransform* to simulate different scales of the pose data to account for relative zoomed-in or zoomed-out view of signers, *PoseRandomShift* to move a significant portion of the video by a time offset so as to make the ISLR models robust to inaccurate segmentation of real-time video, *UniformTemporalSubsample* to uniformly sample frames from the video instead of considering only the initial frames, in cases where the number of frames in a video clip exceeds a maximum limit, and *RandomTemporalSubsample* to sample a random fixed contiguous window of required size covering a maximum number of frames.

We encourage researchers to contribute datasets, models, and other utilities to make sign language research more accessible. All the aspects of the toolkit are well-documented online[3] for anyone to get started easily.

6 Conclusion

In this work, we make several contributions to make sign language research more accessible. We release pose-based datasets and 4 different ISLR models across 6 sign languages. This evaluation enabled us to identify graph-based methods such as ST-GCN as being accurate and efficient. We release the first large corpus of SL data for self-supervised pretraining. We evaluated different pretraining strategies and found DPC as being effective. We also show that pretraining is effective both for in-language and crosslingual transfer. All our models, datasets, training and deployment scripts are open-sourced in 👐 `OpenHands`.

Several directions for future work emerge such as evaluating alternative graph-based models, experimenting with varying sequence lengths of input data, efficiently sampling the data from the raw dataset for pretraining such that the samples are diverse enough, using better pose estimator models and more keypoints, and quantized inference for 2×-4× reduced latency. On the library front, we aim to release updated versions incorporating more SL datasets, better graph-based models, studying the performance on low FPS videos (like 2-4 FPS), effect of pretraining using other high-resource SL datasets, extending to CSLR, and improving deployment features.

[3] https://openhands.readthedocs.io

Acknowledgements

We would like to thank *Aravint Annamalai* from IIT Madras for preparing the list of potential YouTube channels that can be crawled and for his help in downloading them. We would like to thank the entire *AI4Bharat Sign Language Team*[4] for their support and feedback for this work, especially from *Rohith Gandhi Ganesan* for his insights on code structuring, and *Advaith Sridhar* for managing the overall project. We would also like to extend our immense gratitude to Microsoft's *AI4Accessibility* program (via Microsoft Philanthropies India) for granting us the compute required to carry out all the experiments in this work, through Microsoft Azure cloud platform. Our extended gratitude also goes to Zenodo, who helped us with hosting our large datasets (NC and Selvaraj, 2021). Finally, we thank all the content creators and ISLR dataset curators without whose data this work would have been impossible.

References

Julien Abadji, Pedro Ortiz Suarez, Laurent Romary, and Benoît Sagot. 2022. Towards a Cleaner Document-Oriented Multilingual Crawled Corpus. *arXiv e-prints*, page arXiv:2201.06642.

Nikolaos M. Adaloglou, Theocharis Chatzis, Ilias Papastratis, Andreas Stergioulas, Georgios Th Papadopoulos, Vassia Zacharopoulou, George Xydopoulos, Klimis Antzakas, Dimitris Papazachariou, and Petros none Daras. 2021. A comprehensive study on deep learning-based methods for sign language recognition. *IEEE Transactions on Multimedia*, page 1–1.

Samuel Albanie, Gül Varol, Liliane Momeni, Triantafyllos Afouras, Joon Son Chung, Neil Fox, and Andrew Zisserman. 2020. BSL-1K: Scaling up co-articulated sign language recognition using mouthing cues. In *European Conference on Computer Vision*.

Samuel Albanie, Gül Varol, Liliane Momeni, Hannah Bull, Triantafyllos Afouras, Himel Chowdhury, Neil Fox, Bencie Woll, Rob Cooper, Andrew McParland, and Andrew Zisserman. 2021. BOBSL: BBC-Oxford British Sign Language Dataset.

Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations.

Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. 2018. Openpose: Realtime multi-person 2d pose estimation using part affinity fields.

Xiujuan Chai, Hanjie Wang, and Xilin Chen. 2014. The devisign large vocabulary of chinese sign language database and baseline evaluations. *Technical report VIPL-TR-14-SLR-001. Key Lab of Intelligent Information Processing of Chinese Academy of Sciences (CAS), Institute of Computing Technology, CAS*.

Ke Cheng, Yifan Zhang, Congqi Cao, Lei Shi, Jian Cheng, and Hanqing Lu. 2020a. Decoupling gcn with dropgraph module for skeleton-based action recognition. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Ke Cheng, Yifan Zhang, Xiangyu He, Weihan Chen, Jian Cheng, and Hanqing Lu. 2020b. Skeleton-based action recognition with shift graph convolutional network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yi-Bin Cheng, Xipeng Chen, Dongyu Zhang, and Liang Lin. 2021. *Motion-Transformer: Self-Supervised Pre-Training for Skeleton-Based Action Recognition*. Association for Computing Machinery, New York, NY, USA.

Mathieu De Coster, Mieke Van Herreweghe, and Joni Dambre. 2020. Sign language recognition with transformer networks. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 6018–6024, Marseille, France. European Language Resources Association.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding.

Yong Du, Wei Wang, and Liang Wang. 2015. Hierarchical recurrent neural network for skeleton based action recognition. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1110–1118.

Iva Farag and Heike Brock. 2019. Learning motion disfluencies for automatic sign language segmentation. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7360–7364.

S.S. Fels and G.E. Hinton. 1993. Glove-talk: a neural network interface between a data-glove and a speech synthesizer. *IEEE Transactions on Neural Networks*, 4(1):2–8.

Xuehao Gao, Yang Yang, and Shaoyi Du. 2021. Contrastive self-supervised learning for skeleton action recognition. In *NeurIPS 2020 Workshop on Pre-registration in Machine Learning*, volume 148 of *Proceedings of Machine Learning Research*, pages 51–61. PMLR.

Ivan Grishchenko and Valentin Bazarevsky. 2020. Mediapipe holistic — simultaneous face, hand and pose prediction, on device.

[4] https://sign-language.ai4bharat.org

https://ai.googleblog.com/2020/12/mediapipe-holistic-simultaneous-face.html. (Accessed on 08/23/2021).

Tengda Han, Weidi Xie, and Andrew Zisserman. 2019. Video representation learning by dense predictive coding.

Hezhen Hu, Wen gang Zhou, Junfu Pu, and Houqiang Li. 2020a. Global-local enhancement network for nmf-aware sign language recognition. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 17:1 – 19.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020b. Xtreme: A massively multilingual multi-task benchmark for evaluating cross-lingual generalization.

Jie Huang, Wengang Zhou, Houqiang Li, and Weiping Li. 2019. Attention-based 3d-cnns for large-vocabulary sign language recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 29(9):2822–2832.

Songyao Jiang, Bin Sun, Lichen Wang, Yue Bai, Kunpeng Li, and Yun Fu. 2021. Skeleton aware multi-modal sign language recognition.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks.

Sang-Ki Ko, Jae Gi Son, and Hyedong Jung. 2018. Sign language recognition with recurrent neural network using human keypoint detection. In *Proceedings of the 2018 Conference on Research in Adaptive and Convergent Systems*, RACS '18, page 326–328, New York, NY, USA. Association for Computing Machinery.

Oscar Koller. 2020. Quantitative survey of the state of the art in sign language recognition.

Dimitrios Konstantinidis, K. Dimitropoulos, and P. Daras. 2018. Sign language recognition based on hand and body skeletal data. *2018 - 3DTV-Conference: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON)*, pages 1–4.

Dongxu Li, Cristian Rodriguez, Xin Yu, and Hongdong Li. 2020. Word-level deep sign language recognition from video: A new large-scale dataset and methods comparison. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 1459–1469.

Yaobo Liang, Nan Duan, Yeyun Gong, Ning Wu, Fenfei Guo, Weizhen Qi, Ming Gong, Linjun Shou, Daxin Jiang, Guihong Cao, Xiaodong Fan, Ruofei Zhang, Rahul Agrawal, Edward Cui, Sining Wei, Taroon Bharti, Ying Qiao, Jiun-Hung Chen, Winnie Wu, Shuguang Liu, Fan Yang, Daniel Campos, Rangan Majumder, and Ming Zhou. 2020. Xglue: A new benchmark dataset for cross-lingual pre-training, understanding and generation.

Lilang Lin, Sijie Song, Wenhan Yang, and Jiaying Liu. 2020. Ms2l. *Proceedings of the 28th ACM International Conference on Multimedia*.

Li Linguo, Wang Minsi, Ni Bingbing, Wang Hang, Yang Jiancheng, and Zhang Wenjun. 2021. 3d human action representation learning via cross-view consistency pursuit. In *CVPR*.

Ziyu Liu, Hongwen Zhang, Zhenghao Chen, Zhiyong Wang, and Wanli Ouyang. 2020. Disentangling and unifying graph convolutions for skeleton-based action recognition.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality.

Gokul NC and Premkumar Selvaraj. 2021. Openhands v1 : Raw slr pose datasets.

M. Parelli, Katerina Papadimitriou, G. Potamianos, G. Pavlakos, and P. Maragos. 2020. Exploiting 3d hand pose estimation in deep learning-based sign language recognition from rgb videos. In *ECCV Workshops*.

G. Anantha Rao, K. Syamala, P. V. V. Kishore, and A. S. C. S. Sastry. 2018. Deep convolutional neural networks for sign language recognition. In *2018 Conference on Signal Processing And Communication Engineering Systems (SPACES)*, pages 194–197.

S. Reshna, A. Sajeena, and Madhavan Jayaraju. 2020. Recognition of static hand gestures of indian sign language using cnn. volume 2222, page 030012.

Franco Ronchetti, Facundo Quiroga, Cesar Estrebou, Laura Lanzarini, and Alejandro Rosete. 2016. Lsa64: A dataset of argentinian sign language. *XX II Congreso Argentino de Ciencias de la Computación (CACIC)*.

Adam Schembri and Onno Crasborn. 2010. Issues in creating annotation standards for sign language description. In *Proceedings of the LREC2010 4th Workshop on the Representation and Processing of Sign Languages: Corpora and Sign Language Technologies*, pages 212–216, Valletta, Malta. European Language Resources Association (ELRA).

Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. 2019a. Skeleton-based action recognition with directed graph neural networks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7904–7913.

Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. 2019b. Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In *CVPR*.

Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. 2020. Skeleton-based action recognition with multi-stream adaptive graph convolutional networks. *IEEE Transactions on Image Processing*, 29:9532–9545.

Chenyang Si, Ya Jing, Wei Wang, Liang Wang, and Tieniu Tan. 2018. Skeleton-based action recognition with spatial reasoning and temporal stack learning.

Ozge Mercanoglu Sincan and Hacer Yalim Keles. 2020. Autsl: A large scale multi-modal turkish sign language dataset and baseline methods. *IEEE Access*, 8:181340–181355.

Advaith Sridhar, Rohith Gandhi Ganesan, Pratyush Kumar, and Mitesh Khapra. 2020. Include: A large scale dataset for indian sign language recognition. MM '20. Association for Computing Machinery.

UN. 2021. International day of sign languages.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need.

Sijie Yan, Yuanjun Xiong, and Dahua Lin. 2018. Spatial temporal graph convolutional networks for skeleton-based action recognition.

Fang Yin, Xiujuan Chai, and Xilin Chen. 2016. Iterative reference driven metric learning for signer independent isolated sign language recognition. In *Computer Vision – ECCV 2016*, pages 434–450, Cham. Springer International Publishing.

Kayo Yin, Amit Moryossef, Julie Hochgesang, Yoav Goldberg, and Malihe Alikhani. 2021. Including signed languages in natural language processing.

Pengfei Zhang, Cuiling Lan, Junliang Xing, Wenjun Zeng, Jianru Xue, and Nanning Zheng. 2017. View adaptive recurrent neural networks for high performance human action recognition from skeleton data. *2017 IEEE International Conference on Computer Vision (ICCV)*.

Pengfei Zhang, Cuiling Lan, Wenjun Zeng, Junliang Xing, Jianru Xue, and Nanning Zheng. 2020. Semantics-guided neural networks for efficient skeleton-based human action recognition. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yi Zhu, Xinyu Li, Chunhui Liu, Mohammadreza Zolfaghari, Yuanjun Xiong, Chongruo Wu, Zhi Zhang, Joseph Tighe, R. Manmatha, and Mu Li. 2020. A comprehensive study of deep video action recognition.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 19–27.

APPENDIX

A Ethical considerations

All models trained in this work only use pose or skeletal data. Consequently all released datasets and models do not have any personally identifiable information (PII), thereby addressing privacy concerns of those who contributed to these datasets. Furthermore, such a standardization eliminates all visually distinguishing features of individuals like color, gender, ethnicity/race, etc., thereby overcoming any potential biases pertaining to subpopulations.

We address the licensing-related aspects of the datasets in the subsequent sub-sections.

A.1 Release of pose ISLR datasets

Our work builds on existing ISLR datasets across languages, by processing them to retain only pose data. Complying with the respective licenses of the datasets, we release our generated poses only for the openly available datasets with permissive licenses. Out of the 7 datasets we evaluate in the paper, we find that we can release the pose data for 5 of the datasets (AUTSL, WLASL, GSL, LSA64, and INCLUDE), covering 5 sign languages (respectively: Turkish, American, Greek, Argentinian and Indian). The other 2 datasets are CSL and DEVISIGN, belonging to Chinese sign language. The licenses of each original dataset is shown in Table 6.

Dataset	License
AUTSL	Permissive
CSL	Proprietary
DEVISIGN	Proprietary
GSL	Creative Commons 4.0
INCLUDE	MIT
LSA64	Creative Commons 4.0
WLASL	C-UDA

Table 6: Licenses of each ISLR dataset

We do not claim ownership over any of the original ISLR datasets, and release the pose data under the same licensing terms as the original datasets.

A.2 Release of raw pose data

We also open-source the pretraining dataset on Indian Sign Language (ISL) that we explained in Section 4.1. The detailed datasheet of this dataset, including motivation, composition, collection process, preprocessing, distribution, maintenance, and ethical considerations is included after the appendices.

B Inference Benchmarking

In this section, we explain how we achieve over 23fps real-time inference, by using MediaPipe Holistic for generating poses (as an ISLR encoder) and our pose-based models (as decoder) that recognizes the sign at any given window.

B.1 MediaPipe Inference

For pose-estimation, MediaPipe offers 3 variants of models: *heavy*, *full* and *lite* in decreasing order of accuracy but increasing order of inference-speed. The latency of these variants on Intel Xeon E5-2690 v4 CPU with a frame-size of 640x480 were 142.59ms, 55.28ms, and 35.37ms respectively per frame. For all training and testing in this work, we used the heavy model to get the best quality results.

For real-time inference, depending on one's CPU, either of the 3 variants can be used with the trained models, since all the 3 BlazePose models are trained on the same dataset to return same number of keypoints. Based on our experience, we prefer only *lite* or *full* variants depending on the CPU-type, and we find the *heavy* model only suitable if we employ frame-skipping and use decoder models that also work at a lower FPS (below 8fps).

B.2 ISLR Model Inference

Given that SLR is an interactive application, deployability atleast at 23 FPS without noticeable latency is essential. We thus study the latency of our models on various CPU configurations so as to target ubiquitous deployment. For each of the 4 models, we report the model size and latency measured on 4 different CPUs in Table 7. The LSTM model is an order of magnitude faster across all devices than the most accurate SL-GCN model, and is a good candidate when speed is essential at the cost of about 10% accuracy drop that we observed in Table 2. Amongst the graph-based methods, ST-GCN provides a good trade-off being about 2× faster than SL-GCN at the cost of only 3% lower average accuracy across datasets.

The benchmarking is done with a batch size of 1 with complete serial processing (without any data

Model →	**LSTM**	**Transformer**	**ST-GCN**	**SL-GCN**	**SLDPC**
Params →	1.6M	3.8M	2.3M	4.9M	4.0M
CPU	**Latency in milliseconds**				
Xeon E5-2690 v4 (2.60GHz)	08.05	30.64	23.02	52.8	47.60
AMD Ryzen 7 3750H (2.30GHz)	12.94	76.41	86.97	225.3	147.28
Xeon Platinum 8168 (2.70GHz)	05.38	23.76	51.64	112.66	112.52
Xeon E5-2673 v4 (2.30GHz)	09.03	43.69	99.39	201.31	188.43

Table 7: Number of parameters and average latency of different model architectures

loading parallelization). The latencies reported in the table corresponds to average inference time per video using the test set of the INCLUDE dataset, for both freshly trained models and pretrained sign language DPC (SLDPC) model.

Note that encoder (pose estimation) and decoder (classifier) are parallelized such that the former is a producer of skeletons for window of live frames, and the latter is a consumer which recognizes glosses.

C Additional notes on pretraining

In this section, we briefly present a few of the artifacts pertaining to the different configurations of the self-supervised training that we experiment.

C.1 Masking-based pretraining

Figure 4 shows the pretraining loss-plot for masked-language learning, to show that although the model converges, due to the reasons mentioned in the main paper, it does not learn useful representations for the downstream tasks.

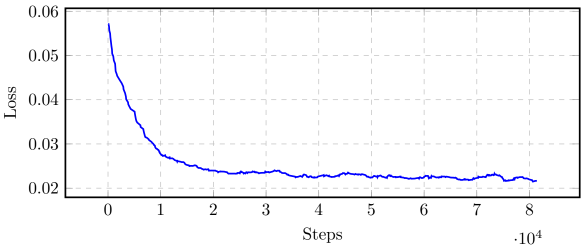

Figure 4: Loss curve for masked pretraining with regression loss

To explain this behaviour, we analyzed the input data as well as the outputs by the model. We find that the model was able to converge because learning to perform an approximate linear interpolation for the masked frames based on the surrounding context was sufficient reduce the loss significantly. However, we posit that such interpolation does not learn any high-level features. This is illustrated in Figure 5, where for each masking span length, we plot the sum of absolute differences between each consecutive masked frames F_i and F_{i-1}, for both predictions from the model as well as the actual frame keypoints. The numbers shown are averaged across all videos in the INCLUDE test set, in which the masking is done around the center region of each video. The plot shows that as masking length is increased, the gap between the predicted values and the actual values diverges indicating an inability to learn longer-range patterns that may be necessary to classify signs.

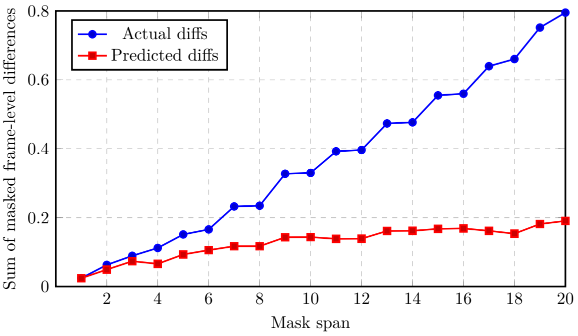

Figure 5: Differences in the output range of masked predictions of pretrained model and corresponding actual keypoints

We also experiment with pretraining using direction loss as explained in background, which essentially is an objective to classify which quadrant the motion vector for each frame will lie. We find that the pretraining does not converge. Upon checking the labels, we see that at the fine-grained level of each frames, the approximately discretized quadrant for each motion vector were seemingly almost random because of the slightly jittery predictions for each frame by the pose estimation model. Also, since the quadrant-type classification encodes only 4 directions, it fails to capture static motion (keypoints which do not move much temporally), which accounts for more than half of the total motion vectors. We thus posit that the direction classification targets are noisy and do not allow the pretraining loss to converge. Figure 6 shows the visualization of quadrants for a randomly-selected joint from a random video in the INCLUDE dataset, to visually verify how noisy the targets for direction loss are.

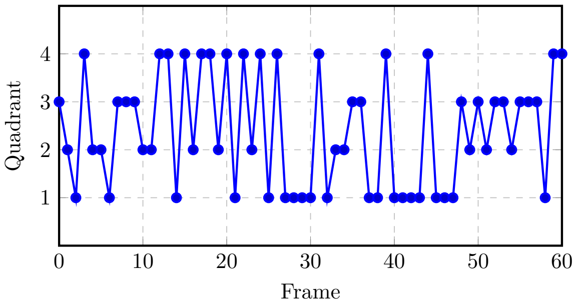

Figure 6: Sample visualization of direction labels for keypoint-15 from the frames of a random INCLUDE video (*Adjectives/4. sad/MVI_9720*)

C.2 Contrastive-learning based

The training setup for this experiment is: For pretraining, we used a batch size of 128 and for finetuning, we used a batch size of 64. For both pretraining and finetuning, we used Adam optimizer with an initial LR of 1e-3. To obtain negative samples, we use a Memory Bank to obtain the embeddings from samples of recent previous batches, which is essentially a FIFO queue of fixed size. We use Facebook's *MoCo codebase* to implement the setup, by plugging-in our ST-GCN as the encoder.

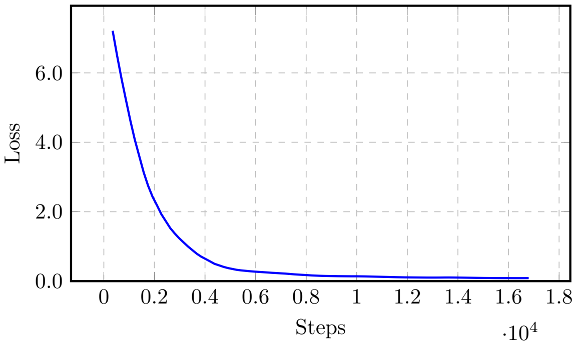

Figure 7: Loss curve for contrastive pretraining

Figure 7 shows the pretraining loss-plot for contrastive learning, to show that although the model converges, as explained in the main paper, the representations learnt do not signify any semantic relationships in the signs. To illustrate this, we take a standard subset of the INCLUDE dataset, called INCLUDE50 (containing 50 classes) and visualize the embeddings of all signs using PCA clustering. Note that each class is uniquely colored to identify if similar signs are grouped together. Figure 8 shows that the learnt embeddings do not discriminate the classes, suggesting that the embeddings may not be informative for the downstream sign recognition task.

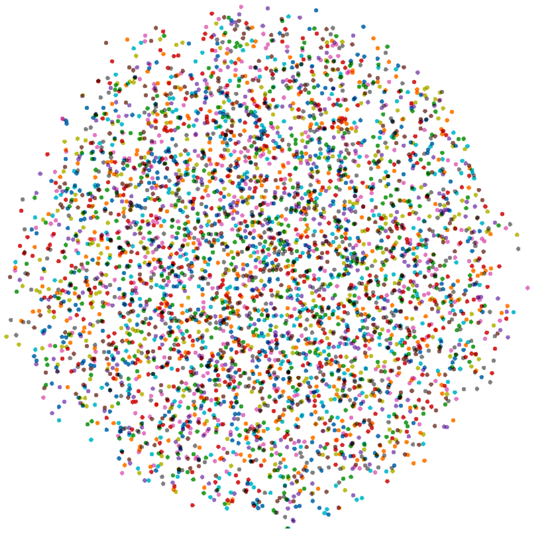

Figure 8: PCA visualization of INCLUDE50 embeddings obtained from Contrastive-Learning model

C.3 Predictive-coding based pretraining

The training setup for this experiment is: For pretraining, we used a batch size of 128 and for finetuning, we used a batch size of 64. For both pretraining and finetuning, we used Adam optimizer with a learning rate of 1e-3.

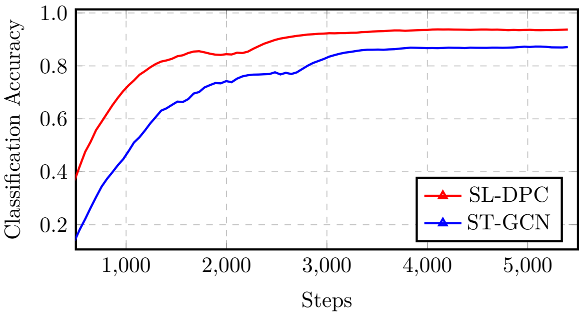

Figure 9: DPC Fine-tuning (orange) vs fresh training (light-green) validation accuracy plot

Figure 9 shows the performance gap between fine-tuning of a DPC pretrained model and an ST-GCN model being trained from scratch. This clearly demonstrates that self-supervised learning produces a significant boost in performance for downstream tasks.

D Sample usage snippets from 👐 OpenHands library

```
import omegaconf
from openhands.apis import ClassificationModel
from openhands.core import get_trainer

cfg = omegaconf.OmegaConf.load("path/to/config.yaml")
trainer = get_trainer(cfg)
model = ClassificationModel(cfg=cfg, trainer=trainer)
model.fit()
```

Figure 10: Example 👐 OpenHands code for running the model training.

```
data:
  modality: "pose"                      #modality to use
  train_pipeline:
    dataset:
      _target_: "dataset_class"         #dataset to use
      split_file: "path"                #labels file path
      root_dir: "path"                  #path to pose data

    transforms:                         #train augmentations
      - RotatationTransform:
          rotation_std: 0.1             #params for each transform

    dataloader:                         #dataloader parameters
      batch_size: 32
      shuffle: true
model:                                  #model parameters
    encoder:
        type: "encoder-to-use"
        params: ...                     #encoder parameters
    decoder:
        type: "decoder-to-use"
optim:                                  #optimizer and loss params
    loss: "CrossEntropyLoss"
    optimizer:
        name: Adam
        lr: 1e-3
trainer:                                #training settings
    gpus: 1
    max_epochs: 100
exp_manager:                            #logging and checkpointing
    create_tensorboard_logger: true     #tensorboard logging
    create_checkpoint_callback: true
    early_stopping_callback: false
```

Figure 11: Example 👐 OpenHands config.

Datasheet for Raw Indian SL corpus

This is the detailed datasheet, including ethical considerations, of the unlabelled pretraining dataset proposed in Section 4.1.

Motivation For Datasheet Creation

Why was the datasheet created? (e.g., was there a specific task in mind? was there a specific gap that needed to be filled?)
There were no large-scale unlabelled datasets available for experimenting with self-supervised learning for sign languages, like we have for NLP, eg. bookcorpus (Zhu et al., 2015) and Common-Crawl (Abadji et al., 2022). Like in NLP, it is expected that such a dataset may help reduce the need for labelled dataset. This dataset was collected with a specific focus on Indian Sign Language (ISL) which has limited labelled resources.

Who created the dataset (e.g., which team, research group) and on behalf of which entity (e.g., company, institution, organization)?
The dataset was programatically created by crawling, cleaning, and preprocessing video data by the main authors of this paper, who are researchers at AI4Bharat group of IIT Madras.

Who funded the creation dataset?
The work was funded by Microsoft Philanthropies India through Microsoft AI4Accessibility program, via AI4Bharat.

Datasheet Composition

What are the instances?(that is, examples; e.g., documents, images, people, countries) Are there multiple types of instances? (e.g., movies, users, ratings; people, interactions between them; nodes, edges)
An instance in our dataset is a bundle of sequence of pose keypoints extracted from the videos of a specific source, in HDF5 format. The original videos are not part of the dataset.

How many instances are there in total (of each type, if appropriate)? The dataset consists of pose keypoints from 7 YouTube channels. Hence there are 7 instances in total.

What data does each instance consist of ? “Raw” data (e.g., unprocessed text or images)? Features/attributes?
Each instance contains the raw pose keypoints data (i.e., without any label data) extracted from the videos of a specific YouTube channel. The features extracted are obtained directly from MediaPipe Holistic tool (Grishchenko and Bazarevsky, 2020), which provides human skeletons for any given set of frames with a person. Our video sources are from news and educational domains. Around 87% of the total data is from 3 news channels. The Education domain channels are *National Institute of Open Schooling*, an intiative by Government of India and SIGN Library channel, an initiative to make educational content in Indian SL.

Is there a label or target associated with each instance? If so, please provide a description.
No, this is an unlabeled raw dataset used for self-supervised pretraining.

Is any information missing from individual instances? If so, please provide a description, explaining why this information is missing (e.g., because it was unavailable). This does not include intentionally removed information, but might include, e.g., redacted text.
We are releasing only the pose keypoints derived from the videos, and not the videos. For reproducibility, we also provide the YouTube video URLs of the corresponding pose keypoints.

Are relationships between individual instances made explicit (e.g., users’ movie ratings, social network links)? If so, please describe how these relationships are made explicit.
All instances are independent of each other and are not linked directly in anyway.

Are there recommended data splits (e.g., training, development/validation, testing)? If so, please provide a description of these splits, explaining the rationale behind them.
No. We recommend to use the entire raw data for pretraining purposes, and as a validation set to compare losses and perform early stopping, we recommend to use the open-sourced INCLUDE dataset (Sridhar et al., 2020). In case if a researcher wants to split the raw data to derive their own train-test split, we recommend them to use the data from "SIGN Library" source for test/development set and the remaining for training.

Are there any errors, sources of noise, or redundancies in the dataset? If so, please provide a

description.

We did our best to ensure there are no redundancies by crawling videos with unique video IDs and titles. There are no labels so there is no error due to labelling. One source of error could be inaccuracy in pose extraction by the MediaPipe library. We did not manually evaluate this accuracy across all datasets, but in manual checks we found MediaPipe to be highly accurate especially since most videos consist of one prominently featured signer with limited or no occlusion.

Is the dataset self-contained, or does it link to or otherwise rely on external resources (e.g., websites, tweets, other datasets)? If it links to or relies on external resources, a) are there guarantees that they will exist, and remain constant, over time; b) are there official archival versions of the complete dataset (i.e., including the external resources as they existed at the time the dataset was created); c) are there any restrictions (e.g., licenses, fees) associated with any of the external resources that might apply to a future user? Please provide descriptions of all external resources and any restrictions associated with them, as well as links or other access points, as appropriate.

The dataset we release is self-contained as used for pretraining in this work. As discussed, to recreate or process the original videos, we provide links to the original videos. However, as this data is hosted on YouTube with rights owned by the respective video creators, these videos may not be available indefinitely.

Collection Process

What mechanisms or procedures were used to collect the data (e.g., hardware apparatus or sensor, manual human curation, software program, software API)? How were these mechanisms or procedures validated?

The data collection pipeline is explained in Section 4 of the main paper. In summary, videos were automatically crawled from the web to collect openly available resources under permissible licenses. These videos were then processed with MediaPipe to obtain pose data as explained in the main paper.

How was the data associated with each instance acquired? Was the data directly observable (e.g., raw text, movie ratings), reported by subjects (e.g., survey responses), or indirectly inferred/derived from other data (e.g., part-of-speech tags, model-based guesses for age or language)? If data was reported by subjects or indirectly inferred/derived from other data, was the data validated/verified? If so, please describe how.

The (pose) data was derived from the crawled videos without any human intervention as stated above. The automatic validation/cleaning of the dataset is explained in the main paper's section 4.

If the dataset is a sample from a larger set, what was the sampling strategy (e.g., deterministic, probabilistic with specific sampling probabilities)?

We release all good quality data in the dataset after a very minimal cleaning process described in the main paper. There was no subjective sampling of the data.

Who was involved in the data collection process (e.g., students, crowdworkers, contractors) and how were they compensated (e.g., how much were crowdworkers paid)?

The code for the automatic crawling and cleaning processes were written by full-time researchers at AI4Bharat (authors of paper), with some help from a volunteer (a full-time student), who has been thanked in the acknowledgements section.

Over what timeframe was the data collected? Does this timeframe match the creation timeframe of the data associated with the instances (e.g., recent crawl of old news articles)? If not, please describe the timeframe in which the data associated with the instances was created.

The crawling was done in the month of June 2021 to include all the videos from the YouTube channels till then. It took over a month to crawl the videos.

Data Preprocessing

Was any preprocessing/cleaning/labeling of the data done (e.g., discretization or bucketing, tokenization, part-of-speech tagging, SIFT feature extraction, removal of instances, processing of missing values)? If so, please provide a description. If not, you may skip the remainder of the questions in this section.

The steps and pipeline used to create the dataset is explained in section 4 of the main paper. No further preprocessing is done before releasing. In addition, the augmentations and normalization performed for training different models are explained in the paper.

Dataset Distribution

How will the dataset be distributed? (e.g., tarball on website, API, GitHub; does the data have a DOI and is it archived redundantly?)
The dataset is released as zipped HDF5 files, one zip for each YouTube channel, and available via Zenodo hosting platform. The dataset has an DOI which is cited in the Acknowledgements section. A mirror link to the dataset can be availed on request (in-case the platform is down or other issues).

When will the dataset be released/first distributed? What license (if any) is it distributed under?
The dataset is released along with the camera-ready version of the paper submitted finally to the conference. It is licensed under the Creative Commons Attribution 4.0 International license.

Are there any copyrights on the data?
The original videos are the copyright of the respective YouTube channels, and are not released. We only release the pose data with no Personally Identifiable Information (PII).

Are there any fees or access/export restrictions?
No.

Dataset Maintenance

Who is supporting/hosting/maintaining the dataset?
The dataset is being hosted at Zenodo, an open-access repository to store and distribute scientific artifacts. The dataset is being maintained by AI4Bharat, a research lab in the CSE department of IIT Madras, India.

Will the dataset be updated? If so, how often and by whom?
If there are any errors found or if any required data is missing, we take responsibility to update/rectify the same.

How will updates be communicated? (e.g., mailing list, GitHub)
It would be conveyed via the changelogs in the GitHub repository.

If the dataset becomes obsolete how will this be communicated?
We do not expect this to happen. But in such a rare case, it will be notified in the GitHub repository as an important note.

Is there a repository to link to any/all papers/systems that use this dataset?
All research works that use our dataset are requested to cite this paper. If this is followed, by viewing the list of citations for this paper (for example on Google Scholar) one could track all papers/systems using the dataset.

If others want to extend/augment/build on this dataset, is there a mechanism for them to do so? If so, is there a process for tracking/assessing the quality of those contributions. What is the process for communicating/distributing these contributions to users?
We would greatly appreciate others adding to the dataset - hence the name of the hosting repository is 🙌 `OpenHands`. Those intending to extend the dataset can contact us on our email addresses or on the Github repository. Quality of the extended dataset would be measured by performance of sign language recognition systems built with the dataset.

Legal and Ethical Considerations

Does the dataset contain data that might be considered confidential (e.g., data that is protected by legal privilege or by doctorpatient confidentiality, data that includes the content of individuals non-public communications)? If so, please provide a description.
No, the dataset does not contain any confidential or personal data.

Does the dataset contain data that, if viewed directly, might be offensive, insulting, threatening, or might otherwise cause anxiety? If so, please describe why
No, the dataset does not contain any offensive or inappropriate data. No audio/speech/image data is included in the data.

Does the dataset relate to people? If not, you may skip the remaining questions in this section.
Yes, but the dataset has only the skeleton information of signers without any PII.

Does the dataset identify any subpopulations (e.g., by age, gender)? If so, please describe how these subpopulations are identified and provide

a description of their respective distributions within the dataset.

No, it does not identify any subpopulations.

Is it possible to identify individuals (i.e., one or more natural persons), either directly or indirectly (i.e., in combination with other data) from the dataset? If so, please describe how.

No, it is not possible to identify the individuals behind the dataset, because the data released contains only the pose points of the signers.

Does the dataset contain data that might be considered sensitive in any way (e.g., data that reveals racial or ethnic origins, sexual orientations, religious beliefs, political opinions or union memberships, or locations; financial or health data; biometric or genetic data; forms of government identification, such as social security numbers; criminal history)? If so, please provide a description.

No, the dataset does not include any sensitive data.

Did you collect the data from the individuals in question directly, or obtain it via third parties or other sources (e.g., websites)?

We obtain it via publicly available YouTube channels released by the respective groups, and not from any individuals.