# Mutual Support of Data Modalities in the Task of Sign Language Recognition

Ivan Gruber Zdeněk Krňoul Marek Hrúz Jakub Kanis
University of West Bohemia, Faculty of Applied Sciences,
Department of Cybernetics and New Technologies for the Information Society
Technická 8, 301 00 Plzeň, Czech Republic
grubiv@ntis.zcu.cz zdkrnoul@ntis.zcu.cz mhruz@ntis.zcu.cz jkanis@ntis.zcu.cz

Matyáš Boháček
Gymnasium of Jan Kepler
Parléřova 2/118, 169 00 Prague 6 – Hradčany, Czech Republic
matyas.bohacek@matsworld.io

## Abstract

*This paper presents a method for automatic sign language recognition that was utilized in the CVPR 2021 ChaLearn Challenge (RGB track). Our method is composed of several approaches combined in an ensemble scheme to perform isolated sign-gesture recognition. We combine modalities of video sample frames processed by a 3D ConvNet (I3D), with body-pose information in the form of joint locations processed by a Transformer, hand region images transformed into a semantic space, and linguistically defined locations of hands. Although the individual models perform sub-par (60% to 93% accuracy on validation data), the weighted ensemble results in 95.46% accuracy.*

## 1. Introduction

Sign languages (SLs) are very complex visual languages. Cues of SLs are a combination of multi-modality and multi-semantic functions [31]. This makes SL recognition a very complex task. The main problem is to effectively extract representative cues, which in visual languages (ie SLs), must be extracted directly from pixels of the images. This is even more challenging for SL recognition in the wild [33, 34].

The undeniable success of Hidden Markov Models (HMMs) in automatic speech recognition has been the initial motivation for its use for SL recognition [37]. Following successful architectures for continuous SL recognition combine spatial models (2D CNN) with temporal models (LSTM or HMM) [23, 40]. Recently, 3D Convolutional Neural Networks (CNNs) and self-attention architectures have been used [28, 43]. Nowadays, Transformers reach state-of-the-art results in the areas of natural language processing (NLP) [15]. In addition, during the last year, Transformers established new state-of-the-art results in many tasks in the visual domain [2, 8, 16].

The main motivation for our approach was to use the state-of-the-art models for gesture/action recognition and augment it with other approaches based on different modalities. We wanted to analyze the performance of an ensemble scheme when different models utilizing different data modalities are used. Previous experiments suggest that the extraction of multi-modal and multi-semantic features (such as keypoints [14]) have a particular impact on class discrimination. As the state-of-the-art model for gesture/action recognition we use I3D [10] and finetune it from Kinetics400 dataset [21] to several data representations of AUTSL dataset [34].

To incorporate the motion of hands we detect the body joints using OpenPose [7] and predict the sign class using a Transformer model inspired by the Vision Transformer (ViT) [16]. The information about the pose of the hands is added by our Visual Language Embedding (VLE) model. In this work, we present a proof-of-concept method that transforms images of hands into a semantic space, where similar poses lie close to each other.

We use concepts from other vision tasks, that show that deep neural networks trained for the classification of images fulfill the requirement of embedded space in the penultimate layer. We finetune a MobileNet [18] architecture pre-trained on ImageNet [30] to classify our mined dataset of hand images. To add information about the location of hands, we developed an algorithm for computing linguistically defined locations of hands. We compute the location vectors from OpenPose detections [7] and together with VLE input them into a Transformer to classify the sign. Lastly, we compute weights of these different models using Covariance Matrix Adaptation Evolution Strategy (CMA-ES) optimization algorithm [3]. As a result of this pipeline, we provide an initial overview of how this modalities' conjunction contributes to SL recognition within the extensive isolated sign AUTSL dataset.

Our contributions are summarized as follows:

1. We have proposed a system for sign language recognition reaching competitive results.
2. We have shown the effectiveness of mutual support of different data modalities.
3. We have proposed a novel visual feature embedding for sign language recognition - VLE.

Through the paper, we provide several supportive images. Generally, red cylinders depict ”outside” data provided by the organizers; green cylinders are data produced by us during this challenge; purple rounded rectangles are 3rd party methods; blue rectangles are our own models/methods.

## 2. Related Work

We can distinguish three tasks in sign language recognition: classification of isolated signs [14], continuous SL recognition [1, 12, 23, 38, 43], and SL translation [5, 42]. Although a combined solution, such as recognition and translation, can improve the performance across both tasks [6].

Recently, the STMC network was introduced to model spatial and temporal cues in the task of continuous SL recognition. The approach learns spatial representation and explicitly decomposes visual features of different cues to make self-contained pose estimation [43]. Although there are common problems in sequence learning, isolated sign recognition is, due to its natural limitation, closer to the classification tasks in the video domain.

Deep learning methods prevail in this area. Especially, 2D convolutional neural networks (2D-CNN) and 3D convolutional neural networks (3D-CNN) are very successful [1, 20, 25, 29, 35]. Additionally, 2D-CNN with temporal convolutional layers [12] and 3D-CNN [26, 38] are adopted to learn dynamic features in SLs.

These appearance-based methods directly create hierarchical representations of spatial-temporal data just like standard convolutional networks, but with spatial-temporal filters. The multi-modality fusion of RGB and optical flow [13] is considered, for example, in [9] the authors trained one I3D network on RGB inputs, and the other one on optical-flow inputs.

Recently, the integration of the attention mechanism shows very good results. Transformers, such as attention-based encoder-decoder models, were originally designed for machine translation [36] and subsequently for SLs [19]. These transformers need a huge amount of training data, however, same as in the areas of NLP [15], they recently achieve state-of-the-art results also in many image processing tasks [8, 16].

The classification task in the image/video domains differs from the task of continuous speech recognition or machine translation in a restriction to the self-attention mechanism. There is no need to learn sequence-to-sequence such as the correspondence between images/frames and sign glosses. Therefore, for the classification in the image/video domain, the original architecture is reduced just to the encoder part that processes the blocks of images or frames as one sequence. This spatial or spatial-temporal sequence is extended by the classification token at the input of the Transformer. The original Vision Transformer architecture was recently further adapted to model long-range sequences in the video domain [2]. These approaches make video classification exclusively on self-attention over space and time. Unfortunately, the authors did not provide codes and pre-trained models yet.

The problem arises in a reduction of image information in general. For Transformer input, 2D RGB pixel data need to be converted into a suitable 1D suitable embedding [27]. This reduction becomes even more significant in the video domain [14]. There exist two main groups of approaches that overcome this problem. The approaches from the first group utilize CNN backbone to extract embedding from image sub-block [39]. The second group of approaches employs hand-crafted features. Hand-crafted features have a long tradition in SL recognition and plenty of them were previously designed especially for SLs [4, 11, 41]. In more recent works, the outputs from the human body pose detector (OpenPose) were also used [24, 42].

## 3. Datasets

AUTSL dataset contains 32302 videos from 43 different speakers in total. On each video, one person is signing one of 226 signs, usually starting and ending in a neutral pose. The organizers of the completion divided the dataset into three subsets - train, development, and test. The train set contains 28142 videos from 31 signers, the development set contains 4418 videos from 6 signers and the test set contains 3742 videos also from 6 signers. Signs' classes in the train set have approximately uniform distribution with a minimum of 90 videos and a maximum of 127 videos per sign.

Considering the diversity within the classes and the fact, each video contains also a relatively big amount of background, we employed the following preprocessing pipeline.

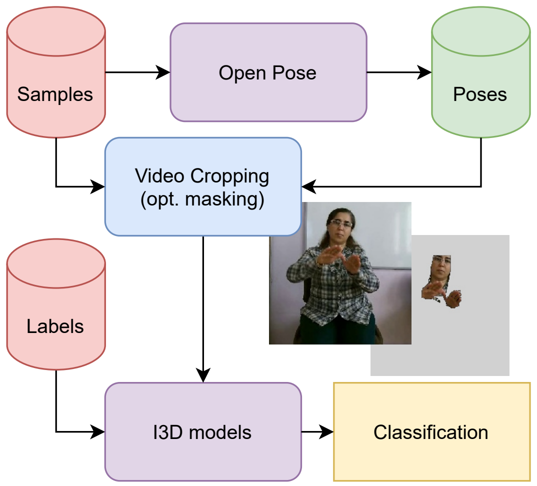

Figure 1. The processing of samples by OpenPose. The poses are then used to crop the video recordings so that we remove unimportant parts of the image. The crop is per-video constant. Optionally, we apply a mask, so that only regions with body-parts are visible. These videos are used to train the different I3D models.

The first step of processing is running OpenPose on all the video samples. We detect all bodies in the video and choose the largest one for further processing. We use the model BODY_25 (default model) to detect 8 body joints (face, neck, left/right shoulder, left/right elbow, and left/right wrist) and 21 joints per hand, meaning in total 50 body joint locations per frame. We also store the confidence of the detected joints.

Next, we prepare data for training the I3D models (Fig. 1). We crop the RGB video frames on a per-video basis. We use the body joints detected by OpenPose. Firstly we get a scale of the body in the sample according to Euclidean distance of the shoulder body joints in the first frame. Next, we crop all remaining frames of the sample relative to the scale (four times), centered in the x-axis on the neck joint and y-axis is below the neck joint by 30% of the shoulder distance. Finally, the crops are resized to the size of $256 \times 256$.

Furthermore, we prepare a masked version of these cropped videos. We start by preparing a binary mask by rendering the detected hand skeleton and face region. We repeat a $3 \times 3$ dilatation on the per-pixel hand skeleton binary mask, for metacarpal bones by a factor of 4, for reaming hand bones by a factor of 2, and for neck/hand bone by a factor of 20. These masks are then used on the original videos to produce the masked versions in which only the important body parts are visible.

We also represent the videos in terms of key-frames. We detect a constant number of frames per video, where there is minimal motion. In our experiments, we have chosen 16 key-frames per video. To obtain the key-frames we first compute the velocity vectors of the detected joints of the whole body as a simple difference in the $x$ and $y$ axis. Next, we compute the magnitude of these vectors and sum up all the joint velocities. The key-frames are then the $N = 16$ minimal locations in the velocity signal. To suppress the detection of nearby key-frames we use a non-minimum suppression with a window of $\pm 3$ frames.

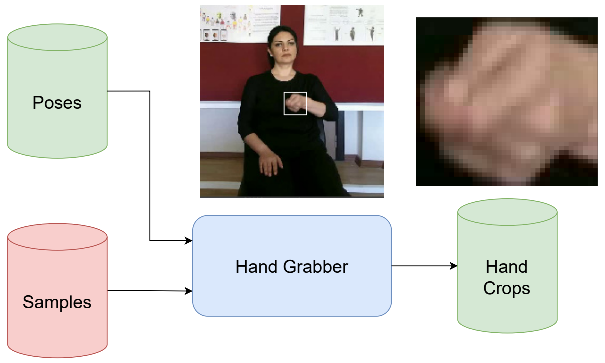

Figure 2. Extraction of hand images (crops) from samples. We extract square regions enclosing all hand joints (with a 10% border) and resize them to $70 \times 70$ pixels.

## 4. Methods

In this section, we describe our pipeline and its individual parts in detail. Firstly, we describe our novel Visual Language Embedding, then the location vectors also used for the training, and lastly the recognition models.

### 4.1. Visual Language Embedding

In this work, we propose a novel visual feature embedding - VLE. VLEs are used as the input for the VLE-Transformer. The premise of the system is to train a deep neural network that will transform an input image into an embedded vector space. This space should have a property that similar hand poses are close to each other. To train such a model we first needed to obtain images of hands in the same pose. The input hand images are obtained using the algorithm depicted in Fig. 2.

We use two consecutive algorithms. The first one finds representative hand poses for each sign. We consider only the dominant hand. We set a parameter of how many representative hand poses per video sample we want to detect at most (we set it experimentally to 5). We sort the detected OpenPose hand joints by the mean confidence. We consider only hand-poses with a minimum confidence of 0.6. We observed that hands with lower confidence are blurred and not suitable to be representatives. We apply a non-maximum suppression of $\pm 5$ frames to suppress the detection of the same hand-pose from the same video sample. For each other video sample, we find all hand-poses that have a distance smaller than 0.42. The threshold was set empirically, by observing what maximal distance is between similar hand-poses. The distance measure is defined as:

$$D(p_1, p_2) = \min_{\mathbf{A}} \sum_{j=1}^{21} \left\| \mathbf{A} p_j^1 - p_j^2 \right\|, \quad (1)$$

where $p^1$ and $p^2$ are the two hand-poses represented as a set of 21 2D vectors provided by OpenPose. The matrix $\mathbf{A}$ is a similarity transform (i.e. restricted to scale, rotation and translation). In practice, we estimate the matrix $\mathbf{A}$ using a least-squares method. The result is in the metric of pixels. Since $p^1$ and $p^2$ can stem from images of different resolutions a normalization needs to be performed. In this step of the solution, we normalize images to the length of the shoulder of the signer with hand $p^2$. This normalization is not perfect and hence the function is not symmetric and thus is not a real distance. But for the purpose of our solution, it is sufficient. This algorithm produces the detection of per-sign representative hand-poses. Next, we want to cluster these representatives, because one sign can be composed of more representative hand-poses. During experimentation with the clustering we modified the distance measure (Eq. 1) so that the similarity transform is found only on palm joints:

$$\mathbf{A}^\star = \min_{\mathbf{A}} \sum_{k} \left\| \mathbf{A} p_k^1 - p_k^2 \right\| \quad (2)$$

where $k$ represents the palm joints (MCPs, CMC, and wrist). And the distance is weighted:

$$D(p_1, p_2) = \sum_{j=1}^{21} \omega_j \left\| \mathbf{A}^\star p_j^1 - p_j^2 \right\|, \quad (3)$$

where $\omega_j$ is weighting different finger joints. Precisely, fingertips have the weight of 3.0, DIPs have the weight of 2.0, PIPs have weights of 1.5. The rest of the weights are 1.0. Finally, they are normalized to a sum of 1. The idea behind these modified formulas is that we want to emphasize the important parts of the hands. Equation 2 finds the orientation, translation, and scale of the hand whilst ignoring the configuration of fingers. Equation 3 reflects the fact that changes of locations of different joints affect the perceived hand-pose differently. We employ an agglomerative clustering with the distance function from Eq. 3. The actual values of the weights in Equation 3 are set based on observation of the results of the clustering. The clustering is stopped when we would merge samples that have a distance of 1.0 or more. The threshold was set experimentally.

In the second algorithm, we want to join the clusters from different signs to obtain the definite hand-pose clusters. We find representative hand-poses for each per-sign cluster. Those are the hand-poses that have a minimal sum of distances to all the other hand-poses from the same sub-cluster. This algorithm leaves us with 52 final hand-pose clusters. When observing the data, we found some errors that were a result of the imperfections in the distance measure. The main problem is that we are limited to 2D distance computation and perspective plays a significant role. Hence, we corrected the errors manually and ended up with 39 hand clusters. Unfortunately, the clusters were heavily imbalanced.

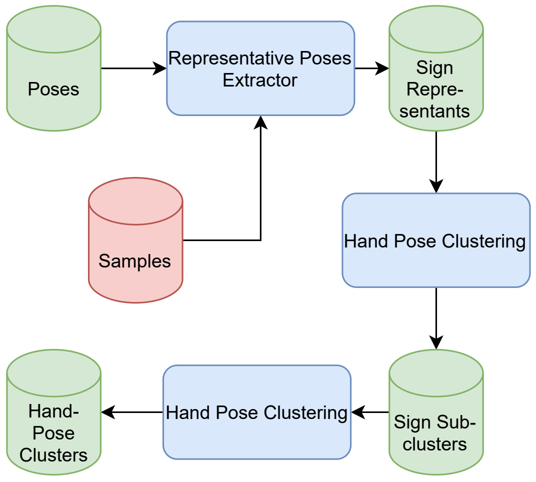

Figure 3. Hand pose clustering. The clustering is based on the computation of weighted pose distance. First, we find per-sign clusters and then we cluster these across all the signs.

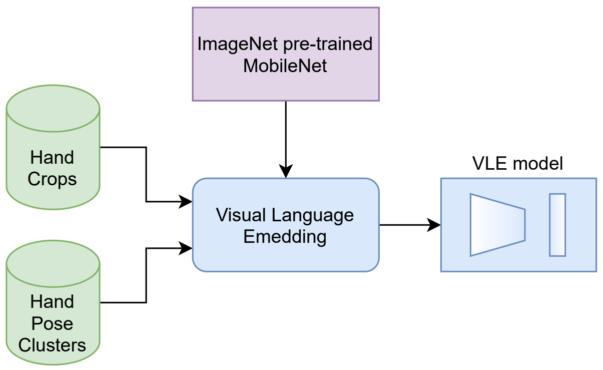

Figure 4. We train the VLE model from the hand crops. Each crop has an assigned cluster (class) from the prior processing. The VLE is a MobileNet pre-trained on ImageNet and fine-tuned on our hand pose clusters.

The last stage was the training of the deep neural network. We performed experiments with ResNet-18 [17], MobileNet and a custom model. We tested randomly initialized models and models pre-trained on ImageNet. The algorithms are depicted in Figures 3, 4 and 5.

### 4.2. Location Vectors

We wanted to incorporate knowledge from the field of sign language linguistics [31], namely the location of the performed sign. We define 15 locations: Neutral space (fallback), Above the head, Upper part of the face, Eyes, Nose,

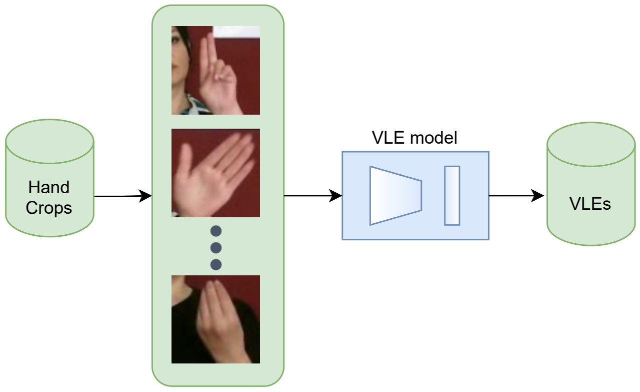

Figure 5. To obtain the VLE representation of test data, we have to obtain the hand crops from test video samples using the process depicted in Fig. 2. Then the model produces VLEs that are a semantic representation of the hand images.

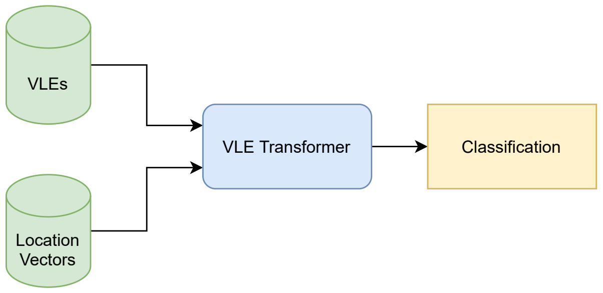

Figure 6. The VLE-transformer takes as input a concatenation of the VLEs and Location Vectors. It has a form of Vision Transformer.

Mouth, Lower part of the face (chin), Cheeks, Ears, Neck, Shoulders, Chest, Waist, Arm, Wrist (of the other hand). The regions are depicted in Fig. 8. To obtain the body regions we utilize the OpenPose joints locations. For each region, we compute an enclosing box with pre-defined relative sizes around the relevant joints. For the face regions, we use the dlib [22] face landmark detector and generate the boxes in a similar manner. The algorithm computes the location of both hands for every frame. First, it detects whether the hand is in a "pointing" gesture by computing the extension of the index finger and bending of the other fingers. The extension and bending are computed as normalized distances of TIP and MCP of the relevant fingers and compared to a threshold.

This approach ignores the effect of perspective transform, which can lead to some misclassifications. If the pointing gesture is recognized, the fingertip of the index finger is used for relative location computation. If not, then both the index fingertip and mean joint location computed from all hand joints are considered. The hand location(s) is then compared to the defined regions. If the hand lies inside a region, the closeness to the center of the region is computed. A normalized vector representing the relative

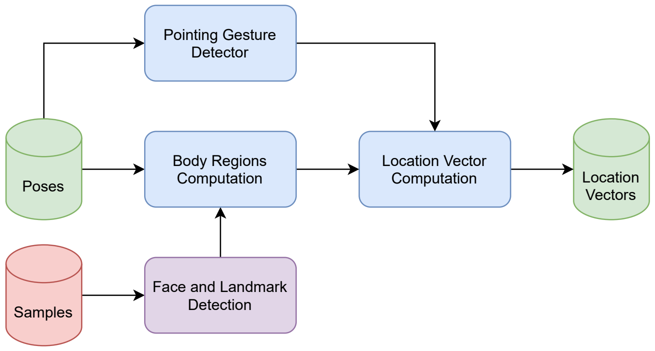
Pointing Gesture Detector
Poses
Body Regions Computation
Location Vector Computation
Location Vectors
Samples
Face and Landmark Detection

Figure 7. The location vectors represent the location of individual hands in individual frames relative to other body-parts. We selected 15 locations depicted in Fig. 8. Since many locations represent facial landmarks, we compute them using the dlib library.

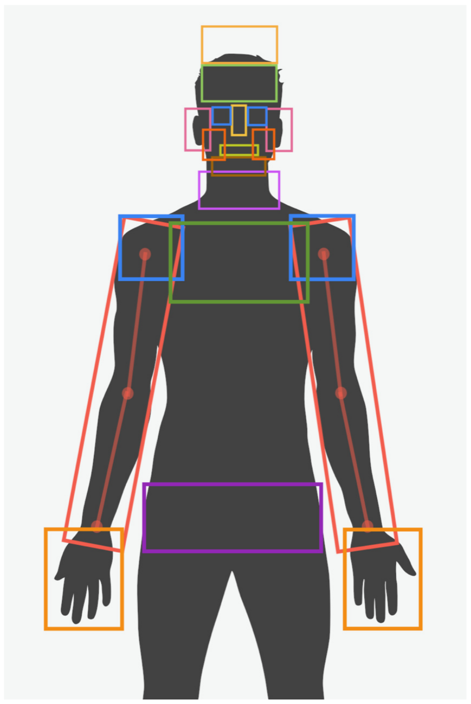

Figure 8. Regions representing different body-locations, from which the location vectors are computed.

closeness to each region is constructed for each hand and concatenated into a 30-dimensional vector (see Fig. 7).

### 4.3. Body-Pose parametrization

Body-Pose parametrization is based on selected 2D keypoints provided by OpenPose. These skeleton data are further pre-processed per frame for each video sample. The whole dataset is normalized to have a uniform distance of the shoulders of signing persons. This distance is calculated from the first frame of each sample. Furthermore, the skeleton data are centered according to the position of the neck. The final pre-processing step corrects hand poses for low confidence images. These hand poses are replaced with poses from surrounding frames.

### 4.4. Recognition Models

Our main recognition model is based on I3D architecture, which provides state-of-the-art results for gesture/action recognition tasks, and therefore it is a very strong baseline. To be more specific, we decided to utilize ResNet50-I3D, which has a very similar architecture as the classical ResNet50, nevertheless, all 2D convolutions are replaced with 3D convolutions. The inputs into this model are RGB video frames.

Other than I3D models, we also used Transformer models. To be more specific, we utilized ViT architecture. The difference from the original architecture lies in the embedding layer that prepares the input data. We utilized feed-forward multi-layer perceptron, which transforms the input vectors of the given parametrization into the transformer input dimension. In our experiments, we used two different parametrizations (data modalities) as an input into the models: Body-Pose, and a combination of VLE and Location vectors.

## 5. Experiments

In this section, we present experimental settings for the training of classification models, an optimization approach for obtaining the final model ensemble, an ablation study of the individual model's importance, and finally the results on the development and the test set.

### 5.1. I3D Training

In our final ensemble, we utilize 13 ResNet50-I3D models in total. Their implementation is based on https://github.com/IBM/action-recognition-pytorch. During the experiment, we also tested I3D-ResNet-101 architecture, however, it provides inferior results. Before the training, each RGB video was cropped (with optional masking) based on detected poses, see Fig. 1. Furthermore, 16 frames with a size of $256 \times 256$ pixels were selected per video. The selection of these frames is based on two different methods. The first method is a pseudo-random choice from the original repository (denoted as random). The second method is based on our key-frames (denoted as key-frames), see Section 3. All I3D models can be divided into three groups.

The first group of four models was trained before the start of the competition test phase. The validation set for these models was signers number 40, 41, and 42. The rest of the signers were in the training set. The models in the first groups were trained during 50 epochs using SGD optimizer with starting learning rate $lr = 0.01$ and cosine learning schedule. These models were fine-tuned using the whole training set after the start of the competition test phase during 20 additional epochs using SGD with starting learning rate $lr = 0.001$ and cosine learning rate schedule.

The second group of four models (denoted as *new*) was trained after the start of the competition test phase. The validation set for these models was competition development data. The models were trained during 80 epochs using SGD optimizer with starting learning rate $lr = 0.01$ and cosine learning schedule.

The third group of five models (denoted as *Cros*) was trained under a 5-fold cross-validation protocol using a competition training set only. Each fold was selected manually with respect to different signers. The models were trained during 50 epochs using SGD optimizer with starting learning rate $lr = 0.01$ and cosine learning schedule again.

All the I3D models were pretrained on the Kinectics400 dataset. Data were normalized to the ImageNet mean and standard deviation. We used batch size $bs = 10$ for all the experiments. Moreover, group center crop is used during the training. A comparison of the models can be found in Table 1. The models with the highest validation recognition rate were selected for the final evaluation.

### 5.2. Pose-transformer Training

We used Body-Pose as an input parametrization for the Pose-transformer, see Section 4.3. The pre-processing step corrects hand poses with confidence $< 0.3$. During the training of the model, we utilized the following augmentations:

- a random drop of the first 10-15 frames from the beginning and the end of the video;
- a random selection of even/odd frames;
- a random horizontal flip of the data (simulation of left/right handed signing);
- Gaussian noise addition to wrist locations and hand-pose scale.

In our final ensemble we used one model based on the body-pose parametrization with the following parameters: max length of the sequence: 120, size of the input vector: 84, N-stages: 2, transformer size: 1024, size of feed-forward layer: 2048, number of heads: 2.

The Pose-transformer model was trained for 100 epochs using SGD optimizer with starting learning rate $lr = 0.1$ and learning rate exponential shift $ex = 0.95$. The model with the highest validation recognition rate was selected for the final evaluation, see Table 1.

### 5.3. VLE-transformer Training

Firstly, we train a deep neural network that transforms an input image into a semantic vector space, see Section 4.1. We performed experiments with three different

Table 1. Details of the models. The recognition rate is calculated on the competition development set. Weights are found via the CMA-ES optimization algorithm.

| **Model** | **Data** | **Frames Slc.** | **Rec. Rate** | **Weight** |
|---|---|---|---|---|
| I3D-Crop, I3D-Crop_new | crops | random | 0.923, **0.929** | 0.04762968, **0.13529561** |
| I3D-Key, I3D-Key_new | | key-frames | 0.909, 0.915 | 0.05915348, 0.04292918 |
| I3D-Mask, I3D-Mask_new | masked | random | 0.918, 0.919 | -0.02789492, 0.07635797 |
| I3D-Key_mask, I3D-Key_mask_new | | key-frames | 0.906, 0.904 | 0.12705846, 0.03582622 |
| I3D-Cros-1 | crops | random | 0.903 | 0.05322100 |
| I3D-Cros-2 | | | **0.912** | **0.12860186** |
| I3D-Cros-3 | | | 0.910 | 0.02714533 |
| I3D-Cros-4 | | | 0.900 | -0.05312429 |
| I3D-Cros-5 | | | 0.895 | -0.04703171 |
| Pose-transformer | Body-Pose | whole video | **0.866** | **0.17253467** |
| VLE-transformer-1 | VLE-Locations | key-frames | 0.603 | 0.08502941 |
| VLE-transformer-2 | | | 0.605 | **0.11312029** |
| VLE-transformer-3 | | | **0.652** | 0.02414777 |

models, both, randomly initialized and pre-trained on ImageNet. From these experiments, we selected the best performing MobileNet pre-trained on ImageNet to serve as the VLE extractor. We used SGD for optimization with a learning rate of 0.001 and momentum of 0.9. We augment the training data using color jitter, horizontal flip (to accommodate for the right hand), per-pixel Gaussian noise, grid distortion, motion blur, random brightness, and contrast transform, RGB shift, rotation of max $\pm 10$ degrees, random crop and resize. Categorical cross-entropy was used as the optimization criterion. After 40 iterations of finetuning we obtain a 95% training accuracy and 65% validation accuracy. The validation data were from two left-out signers. This shows the high sensitivity to the hand shape, and perspective transformations of the hands (since signers perform the signs with different hand orientations). As result, the VLE is the 1280 dimensional vector produced by the penultimate layer of MobileNet.

Secondly, we define VLE-Location parametrization as an input into the VLE-transformer. VLE and Location Vectors, Section 4.2, are concatenated into a single vector. Only key-frames are selected from the whole sequence, which forms the input sequence for the embedding layer. We train the VLE-transformer model without using any additional augmentations during this training phase.

In our final ensemble we used three models based on the VLE-Locations parametrization (VLE-transformer) with the following parameters:

1. max length of the sequence: 120, size of the input vector: 2590, N-stages: 6, transformer size: 1024, size of feed-forward layer: 2048, number of heads: 2;

2. max length of the sequence: 120, size of the input vector: 2590, N-stages: 2, transformer size: 512, size of feed-forward layer: 2048, number of heads: 2;

3. identical as 2) but with additional learn-able positional encoding.

The VLE-transformer models did follow the same training protocol as the Pose-transformer model.

### 5.4. Final Ensemble

The final ensemble is composed of 13 I3D models, one Pose transformer model, and three VLE transformer models. We tackle the problem of model weights optimization as a black-box optimization problem. Therefore, we utilized CMA-ES (Covariance Matrix Adaptation Evolution Strategy) optimization algorithm to compute the weights of these models. CMA-ES is a stochastic optimizer for robust non-linear, non-convex, derivative- and function-value-free numerical optimization. The inputs into the algorithm are predicted soft-max values for the development set, whereas the algorithm should maximize the development set recognition rate. As a starting optimization point, we used an equally weighted ensemble.

## 6. Results and Discussion

The final weight for each model can be found in Table 1. It can be seen that, in this ensemble point, models from all three categories (I3Ds, Pose-transformer, and VLEs) receive relatively high weights (e.g. 0.135, 0.173, and 0.113). Thus, the models of all three categories are important for the final score. Rather surprisingly the highest weight receives the Pose-transformer model (0.173) which is not the best single prediction model (86.6% accuracy vs 92.9% accuracy of the best single model I3D-Crop_new model). But it is the only model working with the whole video clip. The effect of adding models of all particular categories is in Table 2. The listed recognition rate results are for the competition development data.

Table 2. Recognition rates of the different model ensembles. Higher is better. The equal ensemble is the ensemble with the same weights for each model, whereas the optimized ensemble is ensemble with weights obtained via the CMA-ES optimization.

| **Models** | **Equal ens.** | **Optimized ens.** |
|---|---|---|
| all I3D models | 0.9414 | 0.9475 |
| + Pose-transformer | 0.9441 | 0.9518 |
| + all VLE models | **0.9466** | **0.9556** |

| **Final Ensemble** | Dev. Data | Test Data |
|---|---|---|
|  | 0.9556 | **0.9546** |

## 6.1. Final Results

The final results for the AUTSL datasets (development+test) can be found in Table 2. Given the nature of the competition, our method is based on combining many models that perform well on the development data. The work needs to be polished and when the testing phase will be open for experimenting we can perform additional ablation studies. We plan to focus on individual components of the system and also the possibility of training them dependently on each other.

## 6.2. Other Experiments

In this section, we describe our most relevant failed experiments, because we believe that reporting approaches, which did not lead to the goal successfully can also be very beneficial for the scientific community.

### 6.2.1 CNN+LSTM

Inspired by a baseline method [34], we tested the conjunction of convolutional neural network (CNN) with LSTM. CNN's task is to extract spatial features from each frame, whereas these features are sent as inputs into the LSTM part. In the baseline method, the authors employed the VGG model [32]. In our first experiment, we decided to test ResNet-50 instead, which is arguably better architecture for visual classification tasks overall. Unfortunately, this change stems into a minor improvement only. Moreover, the training of the model was unstable. Motivated by the recent success of vision transformers, we experimented also with ViT in place of CNN. Nevertheless, this change also did not lead to any significant improvements.

### 6.2.2 Confusion Matrix

When we observed the results of our individual models, we noticed that some classes are frequently interchanged. We computed the confusion matrix for each of the models. We then changed the inputs into the ensemble in the following way. For each sample and model, we detected the top-1 decision. We computed a confidence threshold as half of the top-1 confidence. All classes above this threshold contribute to the decision (with a confidence value of one) and all classes below this threshold are set to zero. Next, we take the confusion vector from the confusion matrix of the top-1 decision, set the diagonal value to zero (it's the confusion value for the top-1 decision, i.e. the correct classification), and add the other confusion values multiplied by 5.25 to the final ensemble input. Effectively this means that if model A often confuses classes I and II when model A recognizes class I it automatically adds a prediction of II with confidence 5.25 times the confusion between classes I and II. We called these modified inputs hard-soft-max. Although we were able to produce superior accuracy on the development set (slightly above 97%), the test set resulted in lower accuracy of 93.56%. This shows, that this approach leads to overfitting on the validation set.

## 7. Conclusion

Automatic sign language recognition is still a challenging problem even for modern approaches. In this work, we propose the method based on the state-of-the-art algorithm for action recognition I3D and augment its decision, which is based on the RGB modality only, with two Vision Transformers with their inputs from two different modalities - Body-Pose parametrization, and our novel Visual Language Embedding with Location Vectors. Despite the fact, that these two additional models did not provide competitive results on their own, their conjunction with the main model significantly improves the final results.

In future work, we would like to firstly polish our final ensemble. In the second step, we believe that incorporating an additional model working with the optical flow (i.e. with additional data modality) can further improve our results.

**Acknowledgements** This research was supported by the Technology Agency of the Czech Republic, project No. TN01000024 and by the European Regional Development Fund under the project AI&Reasoning (reg. no. CZ.02.1.01/0.0/0.0/15 003/0000466).

## References

[1] Samuel Albanie, Gül Varol, Liliane Momeni, Triantafyllos Afouras, Joon Son Chung, Neil Fox, and Andrew Zisserman. Bsl-1k: Scaling up co-articulated sign language recognition using mouthing cues. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 35–53, Cham, 2020. Springer International Publishing.

[2] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? *arXiv preprint arXiv:2102.05095*, 2021.

[3] Hans-Georg Beyer and Bernhard Sendhoff. Covariance matrix adaptation revisited – the cmsa evolution strategy –. In

Günter Rudolph, Thomas Jansen, Nicola Beume, Simon Lucas, and Carlo Poloni, editors, *Parallel Problem Solving from Nature – PPSN X*, pages 123–132, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[4] P. Buehler, A. Zisserman, and M. Everingham. Learning sign language by watching tv (using weakly aligned subtitles). In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2961–2968, 2009.

[5] Necati Cihan Camgoz, Oscar Koller, Simon Hadfield, and Richard Bowden. Multi-channel transformers for multi-articulatory sign language translation. In Adrien Bartoli and Andrea Fusiello, editors, *Computer Vision – ECCV 2020 Workshops*, pages 301–319, Cham, 2020. Springer International Publishing.

[6] Necati Cihan Camgöz, Oscar Koller, Simon Hadfield, and Richard Bowden. Sign language transformers: Joint end-to-end sign language recognition and translation. *CoRR*, abs/2003.13830, 2020.

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

[8] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pages 213–229. Springer, 2020.

[9] J. Carreira and A. Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4724–4733, 2017.

[10] Chun-Fu Chen, Rameswar Panda, Kandan Ramakrishnan, Rogerio Feris, John Cohn, Aude Oliva, and Quanfu Fan. Deep analysis of cnn-based spatio-temporal representations for action recognition. *arXiv preprint arXiv:2010.11757*, 2020.

[11] Helen Cooper and Richard Bowden. Learning signs from subtitles: A weakly supervised approach to sign language recognition. In *CVPR*, pages 2568–2574, 2009.

[12] R. Cui, H. Liu, and C. Zhang. Recurrent convolutional neural networks for continuous sign language recognition by staged optimization. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1610–1618, 2017.

[13] R. Cui, H. Liu, and C. Zhang. A deep neural framework for continuous sign language recognition by iterative training. *IEEE Transactions on Multimedia*, 21(7):1880–1891, 2019.

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

[15] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[16] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2020.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[18] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications, 2017.

[19] Jie Huang, Wengang Zhou, Qilin Zhang, Houqiang Li, and Weiping Li. Video-based sign language recognition without temporal segmentation. In *AAAI*, pages 2257–2264, 2018.

[20] S. Ji, W. Xu, M. Yang, and K. Yu. 3d convolutional neural networks for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):221–231, 2013.

[21] Will Kay, João Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustafa Suleyman, and Andrew Zisserman. The kinetics human action video dataset. *CoRR*, abs/1705.06950, 2017.

[22] Davis E. King. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 10:1755–1758, 2009.

[23] O Koller, O Zargaran, H Ney, and R Bowden. Deep sign: Hybrid cnn-hmm for continuous sign language recognition. 2016.

[24] Dongxu Li, Cristian Rodriguez, Xin Yu, and Hongdong Li. 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, 2020.

[25] Dongxu Li, Xin Yu, Chenchen Xu, Lars Petersson, and Hongdong Li. Transferring cross-domain knowledge for video sign language recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[26] P. Molchanov, X. Yang, S. Gupta, K. Kim, S. Tyree, and J. Kautz. Online detection and classification of dynamic hand gestures with recurrent 3d convolutional neural networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4207–4215, 2016.

[27] Niki J. Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. In *International Conference on Machine Learning (ICML)*, 2018.

[28] L. Pigou, M. Van Herreweghe, and J. Dambre. Gesture and sign language recognition with temporal residual networks. In *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 3086–3093, 2017.

[29] Zhaofan Qiu, Ting Yao, and Tao Mei. Learning spatio-temporal representation with pseudo-3d residual networks. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 5534–5542. IEEE Computer Society, 2017.

[30] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy,

Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[31] Wendy Sandler and Diane Lillo-Martin. *Sign Language and Linguistic Universals*. Cambridge University Press (CUP), The Edinburgh Building, Cambridge CB2 2RU, UK, 02 2006.

[32] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[33] Ozge Mercanoglu Sincan, Julio C. S. Jacques Junior, Sergio Escalera, and Hacer Yalim Keles. Chalearn LAP large scale signer independent isolated sign language recognition challenge: Design, results and future research. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2021.

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

[35] Hamid Vaezi Joze and Oscar Koller. MS-ASL: A large-scale data set and benchmark for understanding american sign language. In *The British Machine Vision Conference (BMVC)*, September 2019.

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

[37] C. Vogler and D. Metaxas. Adapting hidden markov models for asl recognition by using three-dimensional computer vision methods. In *1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation*, volume 1, pages 156–161 vol.1, 1997.

[38] C. Wei, W. Zhou, J. Pu, and H. Li. Deep grammatical multi-classifier for continuous sign language recognition. In *2019 IEEE Fifth International Conference on Multimedia Big Data (BigMM)*, pages 435–442, 2019.

[39] Bichen Wu, Chenfeng Xu, Xiaoliang Dai, Alvin Wan, Peizhao Zhang, Masayoshi Tomizuka, Kurt Keutzer, and Peter Vajda. Visual transformers: Token-based image representation and processing for computer vision. *CoRR*, abs/2006.03677, 2020.

[40] Y. Ye, Y. Tian, M. Huenerfauth, and J. Liu. Recognizing american sign language gestures from within continuous videos. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2145–214509, 2018.

[41] Fang Yin, Xiujuan Chai, and Xilin Chen. Iterative reference driven metric learning for signer independent isolated sign language recognition. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, pages 434–450, Cham, 2016. Springer International Publishing.

[42] Jan Zelinka and Jakub Kanis. Neural sign language synthesis: Words are our glosses. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, March 2020.

[43] Hao Zhou, Wengang Zhou, Yun Zhou, and Houqiang Li. Spatial-temporal multi-cue network for continuous sign language recognition. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 13009–13016. AAAI Press, 2020.