Proceedings of the 9th Workshop on the Representation and Processing of Sign Languages, pages 123–126
Language Resources and Evaluation Conference (LREC 2020), Marseille, 11–16 May 2020

Recognition of Static Features in Sign Language Using Key-Points

Ioannis Koulierakis[1], Georgios Siolas[1], Eleni Efthimiou[2], Stavroula-Evita Fotinea[2], Andreas-Georgios Stafylopatis[1]

[1]National Technical University of Athens, School of Electrical and Computer Engineering,

Intelligent Systems Laboratory,

[2]Sign Language Technologies Team, Department of Embodied Interaction and Robotics,

Institute for Language and Speech Processing (ILSP) / ATHENA RC

Zografou Campus, 9, Iroon Polytechniou str, 15780 Zografou, Greece,

Artemidos 6 & Epidavrou, 15125 Maroussi, Greece

koulyia@gmail.com, gsiolas@islab.ntua.gr, {eleni_e, evita}@athenarc.gr, andreas@cs.ntua.gr

Abstract

In this paper we report on a research effort focusing on recognition of static features of sign formation in single sign videos. Three sequential models have been developed for handshape, palm orientation and location of sign formation respectively, which make use of key-points extracted via OpenPose software. The models have been applied to a Danish and a Greek Sign Language dataset, providing results around 96%. Moreover, during the reported research, a method has been developed for identifying the time-frame of real signing in the video, which allows to ignore transition frames during sign recognition processing.

Keywords: Sign language recognition algorithm, deep neural networks, key-point extraction

1. Introduction

One of the problems relating to sign language recognition is the lack of appropriate datasets for algorithm training, since most datasets are recorded for academic purposes and as such, they concentrate in human learning rather than machine learning. Therefore, most data collections contain a very large number of different glosses with very few repetitions of each. This characteristic makes it very unlikely for these datasets to be used as training sets for classification algorithms in sign recognition level. Thus, we developed a system in the direction of "phonological" features recognition. This way we can extract a dataset with a lot of examples for every handshape, palm orientation and hand location out of the video collections.

2. Datasets

For the purposes of the project two collections of single gloss videos were used as datasets.

The first one is "Noema +" which was developed by the Greek Institute for Language and Speech Processing (ILSP), Athena. It contains approximately 3000 lemmas of the Greek Language were signed by one native Greek signer and many of them are recorded two or three times. The total amount of videos is 3195 annotated with HamNoSys (Hanke, 2004).

The second one is the "Danish Sign Language Dictionary". It was developed and edited at the Centre for Sign Language and Sign Supported Communication – KC in close cooperation with the Danish Deaf Association (DDL) Centre for Sign Language as a dictionary of the Danish Sign Language (DSL). The dictionary is consisted by single-sign videos as well as videos including short sentences in DSL. We used the single gloss videos which are 2714 in total, signed by several different signers. All these videos are annotated with a variation of HamNoSys that uses only one descriptor per instance (handshape, location, etc). For the description of the handshapes, 69 different names were used. Most of them are named after a letter of the Danish fingerspelling alphabet. Based on them all videos were annotated. This feature is making the whole process easier when trying to split the dictionary into handshape classes.

3. Openpose

OpenPose is a software freely distributed by Carnegie Melon University, Perceptual Computing Lab (Cao et al., 2018). It is used as a tool of human body keypoints extraction from a single image or video frame. It offers an estimation of 25 body/foot keypoints, 2x21 hand keypoints and 70 face keypoints. In the case of a 2D video input, for each keypoint it returns a vector containing 3 elements. The first 2 correspond to the (x,y) coordinates with reference to the upper left corner of the image. The third is a value in the range [0,1] which is quantification of the confidence given by the

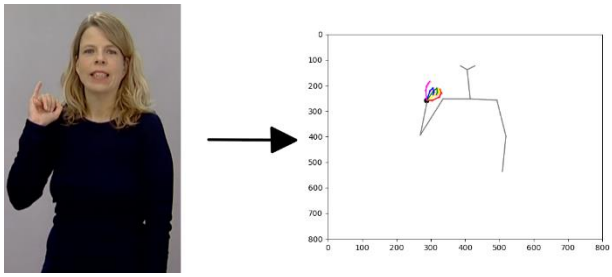

Figure 1: Example of OpenPose

program that the specific keypoint is correctly located in the frame. The novelty behind OpenPose relies on the fact that it works for more than one person per image but more importantly the keypoint analysis is not affected when part of the individual's body is out of frame. This last feature is crucial for applications on sign language videos where the signer appears above the waist level (Figure 1).

4. Our Method

The first step in our method is transforming each video frame into keypoints using the OpenPose software. With this step we keep all the necessary information of the signer's posture and hand articulation, while we reduce the data dimensionality from 1280 x 720 pixels to 137 keypoints. Moreover, in our case the keypoints of the legs are redundant since none of the videos shows the bottom half of the signer. In addition, for each of the systems we will analyze below, we used a different number of keypoints related to the feature we are trying to classify in each case.

In general, the complete feature vector produced by OpenPose has the form:

$$X_\tau = \begin{bmatrix} x_{0,\tau}\ y_{0,\tau}\ \sigma_{0,\tau} \\ x_{1,\tau}\ y_{1,\tau}\ \sigma_{1,\tau} \\ \vdots \\ x_{14,\tau}\ y_{14,\tau}\ \sigma_{14,\tau} \\ x^H_{0,\tau}\ y^H_{0,\tau}\ \sigma^H_{0,\tau} \\ x^H_{1,\tau}\ y^H_{1,\tau}\ \sigma^H_{1,\tau} \\ \vdots \\ x^H_{20,\tau}\ y^H_{20,\tau}\ \sigma^H_{20,\tau} \\ x^h_{0,\tau}\ y^h_{0,\tau}\ \sigma^h_{0,\tau} \\ x^h_{1,\tau}\ y^h_{1,\tau}\ \sigma^h_{1,\tau} \\ \vdots \\ x^h_{20,\tau}\ y^h_{20,\tau}\ \sigma^h_{20,\tau} \end{bmatrix}$$

Where $[x_{i,\tau}, y_{i,\tau}, \sigma_{i,\tau}]$ is the i^{th} keypoint of the τ^{th} frame of the video. With the superscripts H, h we denote the keypoints of the dominant and non-dominant hand, respectively.

4.1. Segmentation

The problem of training a model on our data is a problem of semi-supervised learning. The reason is that in every video the annotation provides us with information on which are the static phonological features appearing in the video and the order in which they appear, but we lack a matching of the static features with individual frames. Moreover, we need a filtering of transitional frames that represent none of the annotated features. Those frames appear when a signer starts or stops signing moving his/her hands from or to resting pose, or during transition from sign to sign or from handshape to handshape into one sign. In all those cases the frames have no use in our training algorithm. In (Koller et al., 2016) this problem is solved by considering a "junk" state for those frames and using an Expectation Maximization algorithm for finding the most probable alignment between the frames and the annotation. On the other hand, we will use an alternative method to what was proposed by (Ko et al., 2018). This method is relying on the work of (Choudhury et al., 2017) that categorizes the movement during signing in "Movement Epenthesis" and "Signing" based on the velocity of the centroid of the contour produced during the hand tracking stage. This method sets a velocity threshold and rejects every sequence of frames with greater velocity than the threshold.

In our method we are transforming each video frame into keypoints using the OpenPose software.
This help us skip the hand recognition and tracking process while maintaining the maximal accuracy provided by OpenPose. In addition, we have the opportunity to calculate the velocity of the hand based on more than one point of interest. We calculate the total hand velocity as the sum of the velocities of each keypoint between two frames on the basis of the following equation:

$$v_\tau = \sum_{i=0}^{20} \sqrt{\left(x^H_{i,\tau} - x^H_{i,\tau+1}\right)^2 + \left(y^H_{i,\tau} - y^H_{i,\tau+1}\right)^2}$$

We present our methodology based on the handshape recognition. Although, the method is outright extendable to the other two characteristics.
We modify the method for rejecting redundant frames based on velocity threshold and extend it by adding one more rule. Every frame is removed from dataset unless it satisfies the following 3 rules:

- Belong in a sequence of 5 frames with total velocity below a threshold T_v.
- The logarithmic sum of certainty σ_i of all points is over a threshold T_σ.
- Wrist is over the waist level (not a hand resting posture)

Both video datasets were recorded at 25 frames/second and so the 5 frame sequence corresponds to 0.2 seconds. We remove the third column from the feature vector X_τ and we use the provided information in the second rule in order to remove bad quality data from our dataset before training. The third rule was added to remove frames from the start and end of each video where the signer is crossing his/her hands on waist level. These frames do not involve signing but they pass the first two rules due to very low movement and clarity.
Our final step is to match each frame remaining to the matching handshape. The advantage here is that the maximum number of different handshapes appearing in every video is 2 due to the fact that we have single gloss videos. If according to the HamNoSys annotation, only one handshape appears in the video, we know that the frames are representing that specific handshape. Otherwise, when two handshapes appear in the video, we are clustering the frames into two clusters using Gaussian Mixture Model (GMM) (Koller et al., 2016) (Theodorakis et al., 2014) (Pitsikalis et al. 2011). The first handshape is matched to the cluster the elements of which appear earlier by mean in the video and the second is matched to the other one.

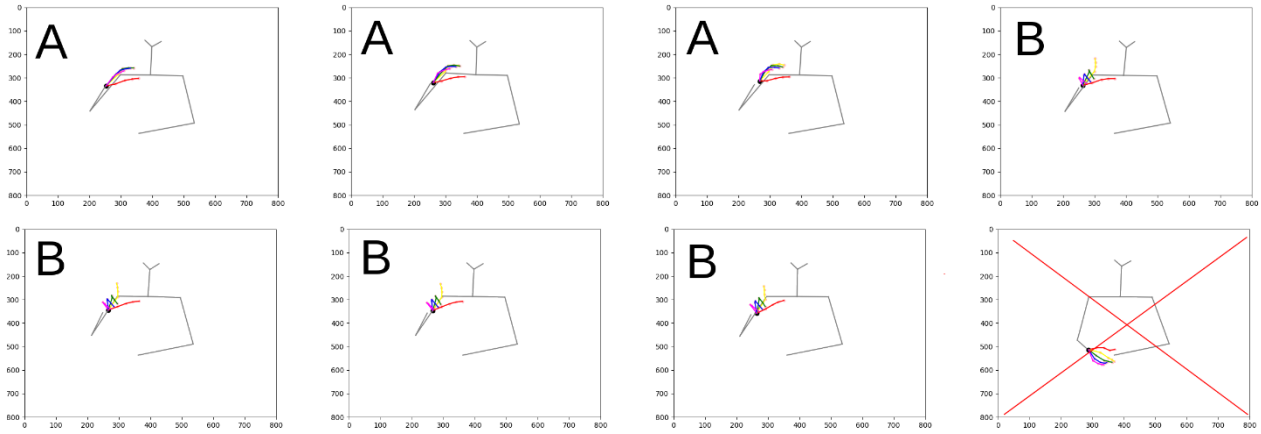

Figure 2: Example of Segmentation process

For example, in Figure 2, there is an example of 8 frames extracted from a video and transformed into keypoints. Beforehand we know that there are two different handshapes. During the segmentation, the last frame is rejected because the signer has crossed his hands and so the right palm is below the elbow level. Otherwise this frame would be labeled as or as adding a false element into the dataset. The GMM algorithm splits the frames into clusters A,B. Cluster-A appears earlier in the sequence so we label the elements of cluster-A, cluster-B as , , respectively.

4.2. Training

At this point we have created a collection of frames representing each possible handshape. In our final dataset, 66 different handshapes, 12 different palm orientations, 33 different locations have found after the segmentations. Approximately 60000 frames ended in the final segmentation for each of the three static features. For the handshape training we used the keypoints extracted only from the dominant hand. We changed the reference system fixing the centroid of the 21 keypoints to (0,0) and the feature vector was normalised using the mean and standard deviation for each dimension, respectively. Moreover, the same process was used for creating the palm-orientation and location feature vectors. For the palm orientation we used all the upper body keypoints plus the dominant hand. Finally, for the location feature vector we included every keypoint including the non-dominant. The reason for this extension of the feature vectors was due to the fact that the orientation of the body of the signer is not the same for all videos and the hand orientation has to be recognized relatively to the body. Obviously, all keypoints are necessary for palm location due to the fact that every articulation is described relatively to a body part including the non-dominant hand and the face. For the two models we use Multi-Layer Perceptrons (MLPs) since we classify each frame independently. 5 hidden layers models with 128 neurons in each layer and softmax activation function are used both for handshape and orientation classifiers. In total we train our models for approximately 400 epochs.

For the training we isolated the 25% of all the videos as a test set. In this way the models are tested in classifying features from signs they have not come up with or even signers for DSL dataset that the systems are not trained on.

4.3. Results

	Train Set	Val. Set	Test Set
Handshape	99.8%	95.7%	95.6%
Palm Orientation	99.7%	96.2%	96.1%
Location	99.8%	97.1%	96.8%

Table 1: Model Accuracies

In Table 1 we can see that the final accuracies of every model is over 95%. According to Figures 3,4,5 the models can almost perfectly classify the training set. We should, also, point out that many of the errors in our classification method could be related to errors during the segmentation.

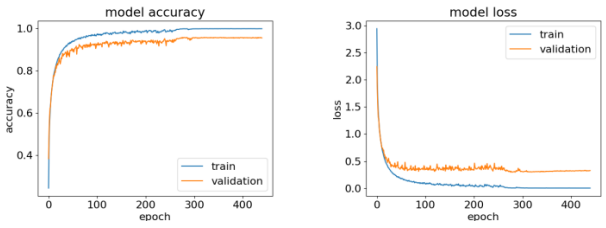

Figure 3: Handshape recognition model

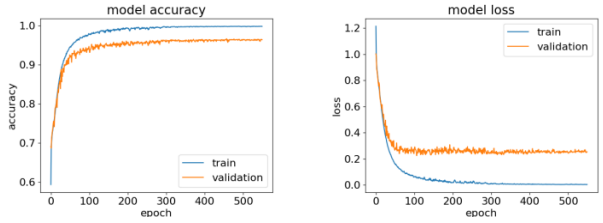

Figure 4: Palm orientation recognition model

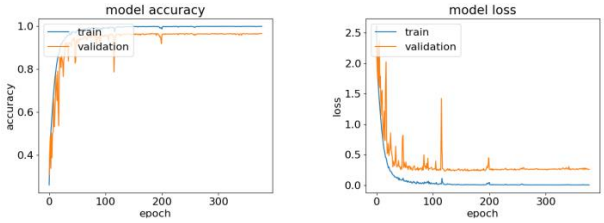

Figure 5: Hand Location recognition model

5. Conclusion

Research efforts relating to recognition of static features of sign formation including the handshape, the palm orientation and the location of signs by means of sequential models, have provided encouraging results as shown in 4.3 above. Such results may prove especially helpful towards (semi-)automatic annotation of SL videos. Furthermore, embedding of the three models handling handshape, palm orientation and location of sign in recurrent neural networks is expected to pave the way towards continuous SL recognition.

6. Acknowledgements

The authors acknowledge support of this work by the project “Computational Science and Technologies: Data, Content and Interaction” (MIS 5002437), which is implemented under the Action “Reinforcement of the Research and Innovation Infrastructure”, funded by the Operational Programme "Competitiveness, Entrepreneurship and Innovation" (NSRF 2014-2020) and co-financed by Greece and the European Union (European Regional Development Fund). Moreover, they wish to thank Center for Tegnsprog (2008–2018). Ordbog over Dansk Tegnsprog. http://www.tegnsprog.dk for providing the Danish Sign Language dataset used in a number of experiments in the framework of the here presented research.

7. References

Ananya Choudhury, Anjan Kumar Talukdar, Manas Kamal Bhuyan, and Kandarpa Kumar Sarma (2017). Movement Epenthesis Detection for Continuous Sign Language Recognition. Journal of Intelligent Systems.

Oscar Koller, Hermann Ney and Richard Bowden, (2016). Automatic Alignment of HamNoSys Subunits for Continuous Sign Language Recognition.

Sang-Ki Ko, Chang Kim, Hyedong Jung, and Choongsang Cho (2019). Neural sign language translation based on human keypoint estimation. Applied Sciences.

Stavros Theodorakis, Vassilis Pitsikalis, and Petros Maragos (2014). Dynamic-static unsupervised sequentiality, statistical subunits and lexicon for sign language recognition. Image and Vision Computing.

Thomas Hanke (2004). HamNoSys - Representing Sign Language Data in Language Resources and Language Processing Contexts. Lrec 2004.

Vassilis Pitsikalis, Stavros Theodorakis, Christian Vogler, and Petros Maragos (2011). Advances in phnetics-based sub-unit modeling for transcrition alignment and sign language recognition.

Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-EnWei, and Yaser Sheikh (2018). OpenPose: realtime multi-person 2D pose estimation usng Part Affinity Fields.