Joints2Sign: Sign Language Recognition through Hand Pose Estimation

Implementation available at https://github.com/enricol96/Joints2Sign

Pastore Giuseppe
Politecnico di Torino
s257649@studenti.polito.it

Loparco Enrico
Politecnico di Torino
enrico.loparco@studenti.polito.it

Abstract

J2S is a new approach for static Sign Language recognition, obtained from the combination of a regression stage and a classification one in a sequential way. It is an end-to-end trainable multi-task model with static or dynamic loss weighting. It offers a trade-off between positives and negatives of visual-based and sensor-based state-of-the-art techniques and high future potentialities in terms of hand rotation invariance because of a polar-based representation of joints' relative position with respect to the wrist's one. On the Italian Sign Language dataset, pre-annotated through OpenPose, J2S reaches good performances, equaling a pure CNN-model in terms of accuracy and even outperforming it when used with the dynamic losses weighting variant.

1. Introduction

Sign Language Sign Language is a visual-manual technique born mainly to fill the communication gap between deaf or in general unable to physically speak people and the whole community. In Italy, more than 8% of the population has hearing problems. Sign Language is a proper language, with its own grammar and lexicon, and for this reason it is very difficult that common people understand it.

State of the art During the last few years, a lot of effort was put into the development of automated recognition and translation systems, especially based on deep learning. As clearly explained in [1], two approaches exist for Sign Language classification: the static sign classification, mainly based on images or other static captures of the gestures, and the continuous or stream sign classification, in which more complex models, mostly based on the combination of CNN, LSTM and RNN, are used to take into account the temporal relation among gestures and movement sequences. This paper focuses on the static sign language classification. Even if there exist some non-deep-learning-based techniques, the major part of the state-of-the-art approaches is based on deep-learning because of the capacity to extract more representative features with respect to the handcrafted ones of shallow models. Most solutions are based on full CNN-based models operating on visual appearance or on more complex models receiving as input the representation of the position and the pose of the hand captured through sensors and external technologies like glove-based ones, motion sensors or Kinect. Both the approaches still have some defects: for example, the latter has high accuracy, but it is not so flexible because of the constraints of the external technologies, that is also an additional cost; the former is more cost-friendly and flexible as it does not depend on any external technology, but suffer from high variability according to the visual appearance of the input data, resulting in lower accuracy, more expensive training and huge number of parameters required.

Joints2Sign In this paper a different hybrid approach is proposed, in which the signs classification is based on the hand joints[1], representing the position of the hand and extracted from RGB images through a CNN-based network. The key idea of the work is combining a hand joints regression model, based on DeepPose [2], with a fully connected classifier in an end-to-end learnable model in order to explore and to evaluate its potentialities with respect to the state-of-the-art and to merge the positives of the previously described approaches.

A2J The starting idea of the project was exploiting the A2J model described in the paper of Xiong *et al*. [3] to retrieve the hand joints position. A2J addresses the hand pose estimation task on 3D depth images 'using a novel anchor-based approach': it estimates joint positions by aggregating estimation results of multiple anchor points, proposed by an anchor proposal branch and used as a point of reference for regression of predicted joints. During the first phase of ideation of this work, the code and the scientific basis of this model were deeply analyzed. However, this path was abandoned in favor of the DeepPose model because of the lack of an annotated depth image Sign Language dataset and the

[1]The terms 'joint' and 'keypoint' are used interchangeably in this paper to refer to the articulations of the hand skeleton

technological impossibility to generate synthetic data compatible with the pre-trained model.

Overview The architecture of the proposed model is based on DeepPose for the joint extraction and on a FCMLNN[2]-based network for the sign label prediction, all integrated into a single model trained at once. A polar conversion from the image scale-dependent Cartesian coordinates of joints is applied before inputting the joint coordinates to the classifier, according to [4]: what is expected from this is a higher robustness of the model to the possible different orientation of the hand, *i.e*. flip/rotation of images, at test time. A more detailed description is reported in the related section.

The accuracy of the hand pose estimation is evaluated according to the metrics described in [5]: average endpoint error (EPE) in pixels and area under the curve (AUC) on the percentage of correct keypoints (PCK) for different error thresholds. The accuracy of the sign prediction is estimated using confusion matrices, widely used for classification tasks and also adopted in the field of hand gesture recognition, as pointed out in [6]. Also, the classification performance of the model is compared with a full CNN-based model.

The dataset used is the Italian Sign Language (LIS) dataset[3] of RGB images, augmented with the joint positions extracted by using OpenPose [7, 8].

2. Related Work

Sign Language Estimation During the last years, with the improvement of technological capabilities and especially of the deep learning techniques, the study of automatic sign language recognition systems has been faced in many scientific works.

Even if there are many shallow learning solutions, the major part of the most promising approaches can be found among the deep-learning-based ones: this is due mainly to the scarce representativeness of the handcrafted features for the assigned task in the case of shallow solutions.

Sign Language can be classified in static and continuous; for each of the different types, some models reveal to be more suitable than others: generally to address Continuous Sign Language recognition some combinations of the models used for the static case together with LSTM or RNN are necessary to capture the temporal sequences of gestures. Our work focuses on Static Sign Language recognition.

During the planning of the work, the approaches analyzed can be divided into two categories: visual-based and sensor-based ones. The former is based on the extraction of significant features for the classification task from images and videos, and so from visual appearance, and it is typically based on a CNN architecture; the latter, instead relies on different techniques (supervised and unsupervised) applied to the input obtained from different sensors capturing the hand pose, like gloves, Kinect or RGB cameras.

We have not taken into account too much the sensor-based solutions because the providing of higher accuracy with respect to the other solutions is constrained by the cost of the external technologies, by the unflexibility brought from the usage of such technologies and from the user unfriendliness in the capturing of the input. A good work in this area is SignSpeak, that uses PCA on input from different sensors to recognize gestures and classify them in real-time.

Related to the visual-based approaches, pure CNN ones are generally expensive to train because of the time and the number of parameters needed, but they can reach good performance under certain visual constraints: therefore they suffer from high variance on light conditions and on visual appearance. Another interesting approach was proposed by Wang *et al*. [9] and it is based on the segmentation of images according to regions with a skin color percentage above some threshold. Again, this solution is extremely bad affected by variance on the external conditions and noise, even if it can be advantageous from a complexity point of view.

In our work, an innovative method is explored, trying to combine the two approaches described above to guarantee a trade-off between flexibility, accuracy and cost.

Hand Pose Estimation As better explained in the related section and already cited in the introduction, the architecture is based on the combination of a hand joint extraction model and a classifier.

Pose estimation and hand pose estimation are hot topics and many works are continuously publicized. Currently, OpenPose seems to be one of the most effective techniques for body and hand pose estimation even if for the latter some constraints on the position of the hand with respect to the elbow must be respected. We used OpenPose as a reference point to generate the ground truth, *i.e*. the annotations, for the dataset used in this work.

DeepPose It is a deep-learning-based approach discussed in [2] and at the base of the regression model in our work. It is a joint regression technique in which the full image is used to learn the joints. The architecture is based on a series of convolutional layers followed by some fully connected layers. Moreover, what is interesting in the original work is the adoption of a cascade of regressors following the initial one using higher resolution sub-images and used as refiners of the joint predictions. However, in our work we limited the implementation to the first stage for performance trade-off, considered also that, according to [2], no particular advantages are brought from more than one stage of refinement because of the limited context used by subse-

[2]Fully connected multi-layer perceptron neural network

[3]https://github.com/maghid/italian_fingerspelling_recognition

quent stages.

A2J It is an innovative technique addressing 3D hand and body pose estimation. It is based on the idea of having a grid of anchor points that act as a regressor for each joint, each one with a different weight according to the assignment of an anchor proposal network. The final position of a joint is obtained from the aggregation of the outputs of all the anchor points. The overall architecture is based on a pipeline composed of a convolutional backbone and three branches: an anchor proposal branch evaluating the response of anchor points towards a joint, *i.e*. assigning the weight, and an in-plain offset and a depth estimation branches, respectively predicting the in-plain and depth offset between joints and anchor points. The model is end-to-end learnable. During the first phase of ideation of our work, A2J has been deeply studied but then abandoned because of dataset issues. However, this approach is interesting because of the high generalization capability given by the presence of the anchor proposal network.

Polar Conversion in the Gesture Classifier Before inputting to the classifier, the skeleton information is converted according to a polar coordinate system, as proposed in [4]. In this paper a task similar to the one faced in this work is discussed: cooking activities recognition in egocentric videos by extracting hand shape features through OpenPose. Obviously, the input is not static. Moreover, the model considered in [4] is not end-to-end trainable but it is based on the flow of independent modules: hand region detection, hand shape feature extraction and activity classification. However, it was interesting to observe the usage of polar conversion in the normalization of the hand key points to input to the FCMLNN.

3. Dataset

As already mentioned, we used the Italian Sign Language (LIS) dataset for our project. It contains 622×415 RGB images representing the hand gestures used in the Italian sign language, split into 22 letters. Letters G, S, J and Z are excluded since the corresponding signs requires movement. The pictures in the dataset were taken from different angles by 11 different people and they have a dark background.

In addition, the dataset was augmented with the joint positions extracted by using OpenPose, as described in the papers of Cao *et al*. [7] and Simon *et al*. [8]. These keypoints are used as ground truth during the training. This solution was chosen because for the time being there is no dataset publicly available provided with both joints and labels.

4. Model

As opposed to typical multi-task learning scenarios, in which multiple independent branches try to learn a task according to a shared feature space obtained propagating their loss along a common convolutional backbone, J2S is an end-to-end model that includes a regression and a classification stage plugged in sequence, *i.e*. the second stage and its loss depend directly on the first stage. The overall architecture is shown in Figure 2.

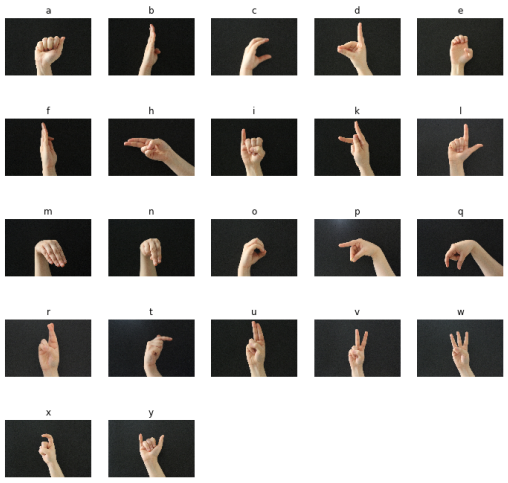


Figure 1: LIS dataset

4.1. Regression stage

This first part of the network aims at extracting the keypoints, *i.e*. solve a regression problem on the hand joints. An architecture similar to the initial stage of [2] is used, but without the dropout layers, that demonstrated to degrade the performance in our specific implementation. As shown in Figure 2, there are 5 CONV layers followed by 3 FC layers. The images, normalized and resized to $256 \times 256 \times 3$ in the pre-processing phase, are received by the first CONV layer, composed of 96 11×11 filters with stride=4 and padding=1. All the CONV layers are activated by a ReLU function and use a stride and a padding of 1, except for the second layer that uses a padding equals to 2; each of the first two layers is followed by a max pooling layer and a normalization layer. The FC6's input is $256 \times 6 \times 6$. At the end of the regression step, an output of 21×2 is returned for each image, representing the 21 Cartesian coordinates retrieved.

4.2. Polar coordinates conversion

As already mentioned, the output of the regression stage is not directly passed to the classification stage, but is subjected to a conversion according to a polar coordinate system. The conversion is obtained as described in [4], using the wrist coordinates p_1 as origin of the new reference sys-

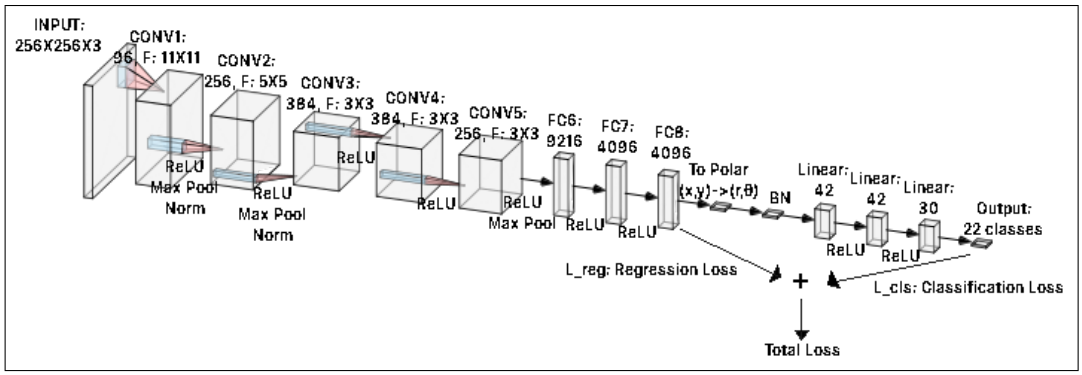


Figure 2: J2S architecture

tem:

$$p_i^{(r)} = \frac{\sqrt{(p_i^{(x)} - p_1^{(x)})^2 + (p_i^{(y)} - p_1^{(y)})^2}}{\sqrt{\text{width}^2 + \text{height}^2}} \tag{1}$$

$$p_i^{(\theta)} = \arctan \frac{p_i^{(y)} - p_1^{(y)}}{p_i^{(x)} - p_1^{(x)}} \times \frac{1}{2\pi} \tag{2}$$

where $p_i^{(r)}$ and $p_i^{(\theta)}$ are the new coordinates for the joint i. Instead, *width* and *height* are the dimensions of the input image. The output is still 21×2 and it represents the relative positions of the joints with respect to the wrist position in the new polar system. The possibility of not detected wrist joint is addressed by pre-calculating the average of the positions of the wrist joint over the training set and substituting it in the computation when necessary. Obviously, this is not the best solution as it strongly depends on the visual appearance and characteristics of the dataset, better solutions can be found, *e.g.* using some heuristics at test time to infer the position of the missing wrist starting from the other keypoints.

4.3. Classification stage

The classification stage takes as input the polar coordinates and performs the sign prediction. It is composed of 3 fully connected layers interleaved with dropout layers and activated by ReLU, in a configuration similar to AlexNet [10]. It is important to notice that a batch normalization layer is added after the polar conversion and before the classification layers to guarantee a good shape of the input data distribution, since the original data distribution of the polar coordinate output demonstrated to degrade the performance of the classifier.

4.4. Loss

Regression loss As proposed in [2] a custom loss function is defined for the regression problem:

$$\mathcal{L}_{reg} = \sum_{(x,y) \in D_N} \sum_{i=1}^{k} \| y_i - \psi_i(x; \theta) \|_2^2 \tag{3}$$

where k indicates the number of joints, x the input image, y the ground truth joints, ψ_i the predicted joint, D_N the set of data with normalized joints. If a key point is not present in the ground truth (because not detected by OpenPose) it is not considered in the summation.

Classification loss The loss used for the classification task is the cross entropy loss.

Tasks losses weighting To obtain the total loss used for the backpropagation as combination of the regression and classification losses, two different approaches are tested and the results of both are reported in this work. The first technique refers to [11] and it is based on the weighted sum of the losses according to:

$$\mathcal{L}_{tot} = \alpha \cdot \mathcal{L}_{reg} + (1 - \alpha) \cdot \mathcal{L}_{cls} \tag{4}$$

where α is a hyper-parameter to tune. The second technique uses dynamically updated weights as explained in [12], according to:

$$\mathcal{L}_{tot} = w_{reg} \cdot \mathcal{L}_{reg} + w_{cls} \cdot \mathcal{L}_{cls} \tag{5}$$

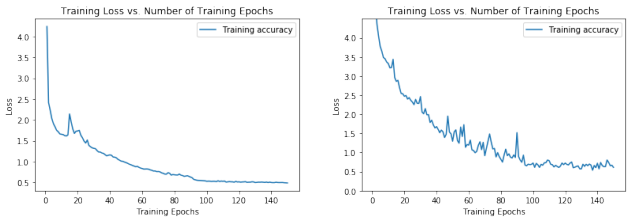
with

$$w_{reg/cls} = \left(\frac{\mathcal{L}_{reg/cls,B}}{\mathcal{L}_{reg/cls,0}} \right)^{\alpha} \tag{6}$$

where $\mathcal{L}_{reg/cls,B}$ is the regression or classification loss calculated over the current batch, while $\mathcal{L}_{reg/cls,0}$ is calculated using the first batch of the epoch. α is still a hyper-parameter to tune.

4.5. Training

Using the usual Mini-batch Stochastic Gradient Descent approach, small quantities of training data at a time are forwarded into the network and the model parameters are updated according to the average loss of the mini-batches. The batch size used is 128 and the momentum is set to 0.9. The training is executed for 150 epochs and the learning rate is decayed by a factor 10 every 90 epochs. Two models were trained, one with static weighting of losses and one with dynamic weighting, and their training was characterized by different hyper-parameters: the former is trained with learning rate equal to 0.01 and $\alpha = 0.5$, while the latter uses learning rate $= 0.005$ and $\alpha = 0.8$. In both cases, the regularization value adopted is low ($5e^{-5}$), since generalization and overfitting did not seem an issue. The trend of the total training losses for the two models is shown in Figure 3.



(a) J2S with static weights (b) J2S with dynamic weights

Figure 3: Training losses

4.6. Cross validation

In order to choose the hyper-parameters that minimize the loss function and better optimize the performance of the model, a cross validation approach is used: the dataset is split in training, validation and test set according to 66-17-17 ratio. A grid search layout is adopted on values obtained through initial empirical observation of parameters. For the model with static loss weighting, it was noticed that values of alpha below 0.6 and of learning rate in the range $[10e^{-2}, 10e^{-4}]$ lead to better performance. Therefore the first grid-search was executed with few epochs on learning rate values in the set $[10e^{-2}, 10e^{-3}, 10e^{-4}]$, α in the set $[0.3, 0.4, 0.5]$ and with a low regularization value ($5e^{-5}$). Once the two best performing hyper-parameters sets are obtained, a further search step with a higher number of epochs has been performed, to have a better estimation of the regression and classification errors on the validation set and to have a look at the loss trends, in order to choose the step size for the learning rate decay. The final parameters chosen are the ones on which the model was trained on the union of training and validation set: lr$= 0.01$, $\alpha = 0.5$, step size $= 90$ and regularization $= 5e^{-5}$. For the model with dynamic loss weighting, basically the same process was followed. The values of α empirically chosen were in the range $[0.5, 1.0]$ and the values of learning rate in $[0.001, 0.005]$. After the execution of the grid search for these values the best performing hyper-parameters were: learning rate $= 0.005$ and $\alpha = 0.8$, keeping the regularization with a low value. A further execution with more epochs was executed to set the step size to 90 epochs.

5. Evaluation

5.1. Hand pose estimation

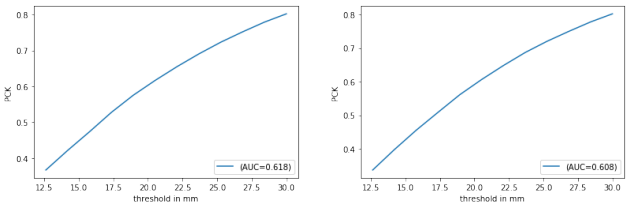
For this task, the most popular evaluation metrics [5] are:

- **Average endpoint error (EPE):** for each point, the endpoint error is computed as the Euclidean distance between the prediction and the ground truth and it is measured in pixels; the mean and median of the EPE are calculated for each keypoint and then averaged between all the joints.
- **Percentage of correct keypoints (PCK):** it consists in the average percentage of predicted keypoints which are within a certain euclidean distance from the ground truth joint location; the threshold is expressed in mm. The PCK curve shows how the percentage changes with varying thresholds. The area under the curve (AUC) summarizes this measure, the higher the better;

The results obtained using these metrics are shown in Table 1 and Figure 4.

Method	EPE median	EPE mean	AUC
$\text{J2S}_{\text{static}}$	17.815	21.913	0.406
$\text{J2S}_{\text{dynamic}}$	17.149	21.817	0.419

Table 1: Regression metrics

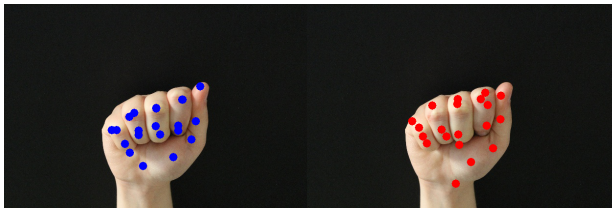


(a) J2S with static weights (b) J2S with dynamic weights

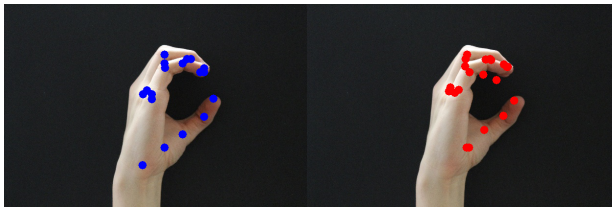
Figure 4: AUC

5.2. Sign classification

The typical measures used in classification are also adopted for hand gesture recognition [6]: **precision**, **recall**, $\boldsymbol{F_1}$ **score** and **confusion matrix**. In particular, in confusion matrices, each row represents the real class while the



(a) OpenPose, letter A (b) J2S, letter A



(c) OpenPose, letter C (d) J2S, letter C

Figure 5: Qualitative measures

columns represent the predictions. It is useful in our case to spot the most challenging signs to be detected: in fact, *R* and *U* letters are quite similar, fooling our model trained with static weights as shown in Figure 6.

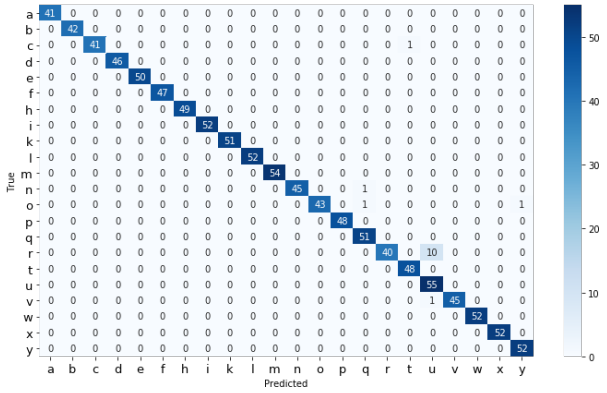
Moreover, the classification performance of our model is compared (Table 2) with a full CNN-based model, whose architecture is explained in the ISL dataset documentation and that has been trained using the procedure and parameters described in it. It consists in 3 groups of CONV and POOL layers and 3 groups of FC layers.

Method	Acc. (%)	Precision	Recall	F_1 Score
ISL	99.07	0.9906	0.9908	0.9906
$J2S_{static}$	98.60	0.9889	0.9858	0.9865
$J2S_{dynamic}$	99.62	0.9962	0.9961	0.9961

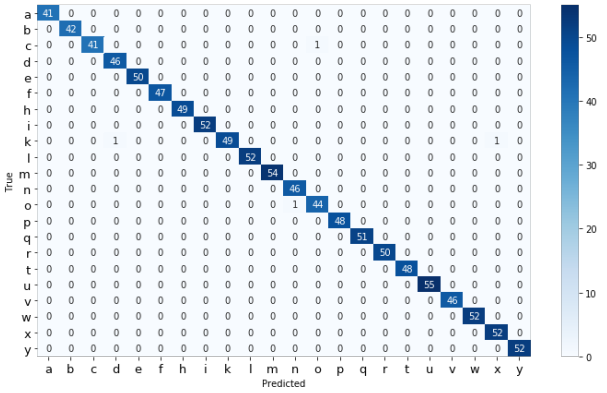
Table 2: Classification metrics

5.3. Qualitative results

In figure 5 the results of the regression stage of the model are shown for two signs of the data set, *A* and *C*, and compared with the OpenPose annotation, *i.e.* the assumed ground truth. It is possible to see that the model performs pretty well and that the degree of precision of the estimate of the joints' position depends on the accuracy of the pre-annotation: in the case of *C* letter the joints plotted on the image does not reflect the actual position of the finger and it is partially due to the intrinsic imprecision of the related ground truth, besides the error induced by the model.



(a) J2S with static weights



(b) J2S with dynamic weights

Figure 6: Confusion Matrices

5.4. Dynamic weighting effectiveness analysis

The dynamic weighting approach revealed to be interesting, increasing the performance from the point of view of both regression and classification metrics. In addition, it helped to generate a model able to distinguish when making predictions between the most similar signs *R* and *U* (Figure 6). The loss trend in Figure 3 shows a greater instability during training compared to the smoother plot with static weights, with peaks justified by the batch-based adaptive updates.

6. Conclusion

The model presented in this work addresses Sign Language recognition through an innovative approach based on multi-task learning. It offers a trade-off between two possible strategies used in literature to learn hand gestures: visual-based models and sensors-based ones, trying to provide the same accuracy of the latter, classifying gestures starting from joints, and the cost-friendliness and flexibility of the former, in capturing input directly from images.

Visual invariance, particularly rotation one, is another is-

sue that the model could potentially address: in the actual implementation the usage of a polar coordinate relative system makes the representation of key points' position related only to the position of the wrist joint and so independent of the absolute position in the image; this could be good to guarantee invariance of sign classification with respect to the hand's rotation if it was not for the limits imposed by the convolutional backbone, that is rotation variant.

A first improvement to solve the issue could be the adoption of some other visual-based keypoints extraction techniques or also of convolutional layers trained on bigger dataset and with technique that include hand rotation (*e.g.* data augmentation), in such a way that the polar conversion could effectively make the classifier correctly performing on all the images, despite of rotation.

However, on the current status and with the shown dataset, the overall classification performance of the model is very good and there is no much room for improvements, as the accuracy on the test set is 98,60% for the model with static loss weighting and 99.62% using dynamic loss weighting. The latter even outperforms the pure CNN model used for comparison.

The joints extraction produces good results too, but is influenced by the limitation of having the ground truth coordinates not being human-annotated or sensor-based but obtained using another model (even if good for the state-of-the-art), *i.e.* Open Pose, and so characterized by intrinsic imprecision.

In future work, it would be interesting to try our model with a bigger and sensor-annotated data set, to better test the effectiveness of the regression stage and try to build a model more robust to visual variance and explore improvements related to possible joints occlusion in the input or absence of specific keypoints in the ground truth.

References

[1] August C. Thio-ac Maria Abigail B. Pamahoy Joni Rose R. Forteza Lean Karlo S. Tolentino, Ronnie O. Serfa Juan and Xavier Jet O. Garcia. Static sign language recognition using deep learning.

[2] Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. *2014 IEEE Conference on Computer Vision and Pattern Recognition*, Jun 2014.

[3] Fu Xiong, Boshen Zhang, Yang Xiao, Zhiguo Cao, Taidong Yu, Joey Zhou Tianyi, and Junsong Yuan. A2j: Anchor-to-joint regression network for 3d articulated pose estimation from a single depth image. In *Proceedings of the IEEE Conference on International Conference on Computer Vision (ICCV)*, 2019.

[4] Tsukasa Okumura, Shuichi Urabe, Katsufumi Inoue, and Michifumi Yoshioka. Cooking activities recognition in egocentric videos using hand shape feature with openpose. In *CEA/MADiMa '18*, 2018.

[5] Christian Zimmermann and Thomas Brox. Learning to estimate 3d hand pose from single rgb images. In *IEEE International Conference on Computer Vision (ICCV)*, 2017. https://arxiv.org/abs/1705.01389.

[6] Guillaume Devineau, Fabien Moutarde, Wang Xi, and Jie Yang. Deep learning for hand gesture recognition on skeletal data. In *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*. IEEE, May 2018.

[7] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields. In *arXiv preprint arXiv:1812.08008*, 2018.

[8] Tomas Simon, Hanbyul Joo, Iain Matthews, and Yaser Sheikh. Hand keypoint detection in single images using multiview bootstrapping. In *CVPR*, 2017.

[9] Huafeng Wang, Y. Wang, and Y. Cao. Video-based face recognition: A survey. *World Academy of Science, Engineering and Technology*, 60:293–302, 12 2011.

[10] Alex Krizhevsky. One weird trick for parallelizing convolutional neural networks. *CoRR*, abs/1404.5997, 2014.

[11] Yue Wu, Yinpeng Chen, Lu Yuan, Zicheng Liu, Lijuan Wang, Hongzhi Li, and Yun Fu. Rethinking classification and localization in R-CNN. *CoRR*, abs/1904.06493, 2019.

[12] Shengchao Liu, Yingyu Liang, and Anthony Gitter. Loss-balanced task weighting to reduce negative transfer in multi-task learning. In *AAAI*, 2019.