Real-Time Sign Language Detection using Human Pose Estimation

Amit Moryossef[1,2], Ioannis Tsochantaridis[1], Roee Aharoni[1],
Sarah Ebling[3], and Srini Narayanan[1]

[1] Google {ioannis,roeeaharoni,srinin}@google.com
[2] Bar-Ilan University amitmoryossef@gmail.com
[3] University of Zurich ebling@cl.uzh.ch

Abstract. We propose a lightweight real-time sign language detection model, as we identify the need for such a case in videoconferencing. We extract optical flow features based on human pose estimation and, using a linear classifier, show these features are meaningful with an accuracy of 80%, evaluated on the Public DGS Corpus. Using a recurrent model directly on the input, we see improvements of up to 91% accuracy, while still working under 4ms. We describe a demo application to sign language detection in the browser in order to demonstrate its usage possibility in videoconferencing applications.

Keywords: Sign Language Detection, Sign Language Processing

1 Introduction

Sign language detection [3] is defined as the binary-classification task for any given frame of a video if a person is using sign-language or not. Unlike sign language recognition [8, 9], where the task is to recognize the form and meaning of signs in a video, or sign language identification, where the task is to identify *which* sign language is used, the task of sign language detection is to detect *when* something is being signed.

With the recent rise of videoconferencing platforms, we identify the problem of signers not "getting the floor" when communicating, which either leads to them being ignored or to a cognitive load on other participants, always checking to see if someone starts signing. Hence, we focus on the real-time sign language detection task with uni-directional context to allow for videoconferencing sign language prominence.

We propose a simple human optical-flow representation for videos based on pose estimation (§3.1), which is fed to a temporally sensitive neural network (§3.2) to perform a binary classification per frame — is the person signing or not. We compare various possible inputs, such as full-body pose estimation, partial pose estimation, and bounding boxes (§4), and contrast their acquisition time in light of our targeted real-time application.

We demonstrate our approach on the Public DGS Corpus (German Sign Language) [11], using full-body pose estimation [27] collected through OpenPose

[5, 28]. We show results of 87%-91% prediction accuracy depending on the input, with per-frame inference time of $350 - 3500\mu$s (§5), and release our training code and models[1].

2 Background

The computational sign language processing (SLP) literature rarely addresses detection [3] and mainly focuses on sign language recognition [8, 9, 17] and identification [10, 19].

2.1 Sign Language Detection

Previous work [3] introduces the classification of frames taken from YouTube videos as either signing or not. They take a spatial and temporal approach based on VGG-16 CNN [29] to encode each frame and use a GRU [7] to encode the sequence of frames, in a window of 20 frames at 5fps. In addition to the raw frame, they also either encode optical flow history, aggregated motion history, or frame difference. However, for our use case, 5fps might not be enough, as it introduces an artificial 200ms delay from when a person starts signing to when they could be detected. Furthermore, this network takes upwards of 3 seconds to run on CPU per inference.

Most recently, Apple [2] announced sign language detection for group FaceTime calls in iOS 14, iPadOS 14, and macOS Big Sur. They did not share any implementation details of their detection model, which makes it hard to compare their model to the one we propose in this paper. Nonetheless, as FaceTime group calls are encrypted end-to-end, we assume that the detection happens on-device rather than on the server-side.

2.2 Sign Language Recognition

Sign language recognition has been widely studied across different domains and sign languages. As sign language corpora are usually small [4], previous works take one of two approaches to reduce the network's parameters: (1) using pose estimation on the original videos [16, 32, 17]; or (2) using pre-trained CNNs to get a feature vector per frame [9, 8]. While different, both methods can encode adequate features to be used for recognition. Studies of human signers have shown that detailed information like exact descriptions of the hand shape are not always required for humans to interpret sign language [25, 31].

Looking at examples of sign videos, we hypothesize that the most challenging part of this task is to identify when a person starts signing, because a signer might initiate hand movement for other purposes, for example, to touch their face. Distinguishing this type of ambient motion from actual linguistic sign movement is not always straightforward. Although not explicitly studied on signers, studies

[1] https://github.com/google-research/google-research/tree/master/sign_language_detection

find the average person touches their face between 15.7 and 23 times per hour [20, 18]. Further complicating this issue, people in different cultures exhibit different face-touching patterns, including frequency, area, and hand preference [12].

2.3 Sign Language Identification

A study [10] finds that a random-forest classifier can distinguish between British Sign Language (BSL) and Greek Sign Language (ENN) with a 95% F1 score. This finding is further supported by more recent work [19] which manages to differentiate between British Sign Language and French Sign Language (Langue des Signes Française, LSF) with 98% F1 score in videos with static backgrounds, and between American Sign Language and British Sign Language with 70% F1 score for videos mined from popular video sharing sites. The authors attribute their success mainly to the different fingerspelling systems, which is two-handed in the case of BSL and one-handed in the case of ASL and LSF.

3 Model

For a video, for every frame given, we would like to predict whether the person in the video is signing or not.

3.1 Input Representation

As evident by previous work [3], using the raw frames as input is computationally expensive, and noisy. Alternatively, in computer vision, optical flow is one way to calculate the movement of every object in a scene. However, because signing is inherently a human action, we do not care about the flow of every object, but rather only the flow of the human. Optimally, we would like to track the movement of every pixel on the human body from one frame to another, to gauge its movement vector. As a proxy to such data, we opt for full-body human pose estimation, defining a set of points detected in every video frame that marks informative landmarks, like joints and other moving parts (mouth, eyes, eyebrows, and others).

Getting the optical flow F for these predefined points P at time t is then well defined as the L2 norm of the vector resulting from subtracting every two consecutive frames. We normalize the flow by the frame-rate in which the video was captured for the representation to be frame-rate invariant (Equation 1).

$$F(P)_t = ||P_t - P_{t-1}||_2 * fps \tag{1}$$

We note that if a point p was not identified in a given frame t, the value of $F(p)_t$ and $F(p)_{t+1}$ automatically equals to 0. This is done to avoid introducing fake movements from a poor pose estimation system or unknown movement from landmarks going out-of-frame.

An additional benefit of using full-body pose estimation is that we can normalize the size of all people, regardless of whether they use a high-/low-resolution camera and the distance at which they are from the camera.

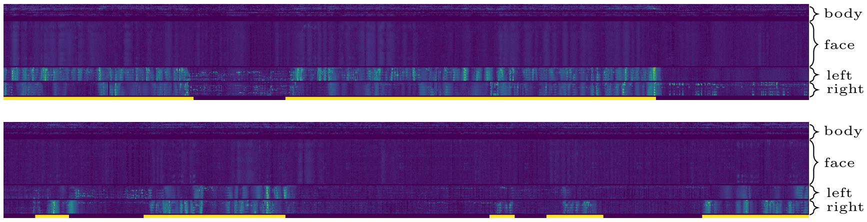

Fig. 1: Optical-flow norm representation of a conversation between two signers. The x-axis is the progression of time, 1,500 frames over 30 seconds in total. The yellow marks are the gold labels for spans when a signer is signing.

3.2 Temporal Model

Figure 1 demonstrates our input representation for an example video. It shows, to the naked eye, that this representation is meaningful. The movement, indicated by the bright colors, is well aligned with the gold spans annotation. Thus, we opt to use a shallow sequence tagging model on top of it.

We use a uni-directional LSTM [14] with one layer and 64 hidden units directly on this input, normalized for frame rate, and project the output to a 2-dimensional vector. For training, we use the negative-log-likelihood loss on the predicted classes for every frame. For inference, we take the arg-max of the output vector (Equation 2).

$$signing(P) = \arg\max LSTM(F(P)) * W \tag{2}$$

Note that this model allows us to process each frame as we get it, in real-time, by performing a single step of the LSTM and project its output. Unlike autoregressive models, we do not feed the last-frame classification as input for the next frame, as just classifying the new frame with the same tag would almost get 100% accuracy on this task, depending on gold labels to be available. Instead, we rely on the hidden state of the LSTM to hold such information as a probability.

4 Experiments

The Public DGS Corpus [11] includes 301 videos with an average duration of 9 minutes, of two signers in conversation[2], at 50fps. Each video includes gloss annotations and spoken language translations (German and English). Using this information, we mark each frame as either “signing” (50.9% of the data) or “not-signing” (49.1% of the data) depending on whether it belongs to a gloss segment. Furthermore, this corpus is enriched with OpenPose [5] full-body pose estimations [27] including 137 points per frame (70 for the face, 25 for the body, and 21 for each hand). In order to disregard video resolution and distance from

[2] There are also monologue story-telling, but both signers are always shown.

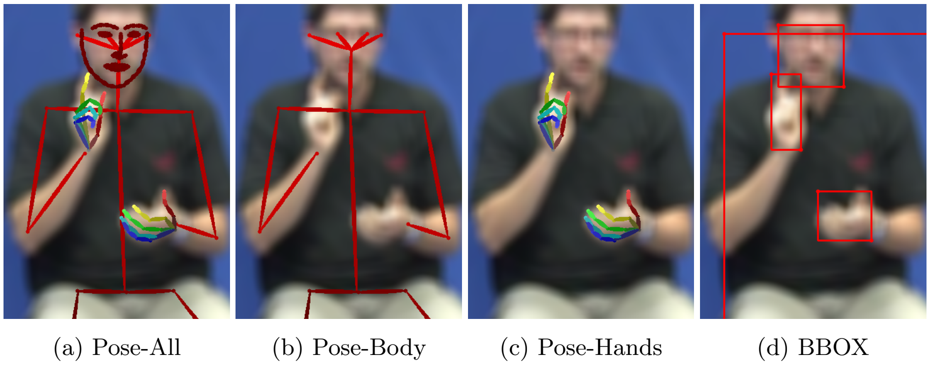
(a) Pose-All (b) Pose-Body (c) Pose-Hands (d) BBOX

Fig. 2: Visualization of our different experiments inputs.

the camera, we normalize each of these poses such that the mean distance between the shoulders of each person equals 1. We split this dataset into 50:25:25 for training, validation, and test, respectively. For every "part" (face, body, left and right hands), we also calculate its bounding box based on the minimum and maximum value of all of the landmarks.

We experiment with three linear baselines with a fixed context (Linear-1, Linear-25, Linear-50) and four experimental recurrent models with different counts of input features:

1. **Pose-All**—including all of the landmarks from the poses. (f. 2a)
2. **Pose-Body**—including only the body landmarks. (f. 2b)
3. **Pose-Hands**—including only the left- and right-hand landmarks. (f. 2c)
4. **BBOX**—including the bounding boxes of the face, body, and hands. (f. 2d)

Finally, we measure the execution time of each model on CPU, using an Intel(R) Xeon(R) CPU E5-2650 v4 @ 2.20GHz. We measure the execution time per frame given a single frame at a time, using multiple frameworks: Scikit-Learn (sk) [24], TensorFlow (tf) [1] and PyTorch (pt) [23].

5 Results

Table 1 includes the accuracy and inference times for each of our scenarios. Our baseline systems show that using a linear classifier with a fixed number of context frames achieves between 79.9% to 84.3% accuracy on the test set. However, all of the baselines perform worse than our recurrent models, for which we achieve between 87.7% to 91.5% accuracy on the test set. Generally, we see that using more diverse sets of landmarks performs better. Although the hand landmarks are very indicative, using just the hand BBOX almost matches in accuracy, and using the entire body pose, with a single point per hand, performs much better. Furthermore, we see that regardless of the number of landmarks used, our models generally perform faster the fewer landmarks are used. We note that the prediction time varies between the different frameworks, but does not vary much within a particular framework. It is clear, however, that the speed of these models' is sufficient, as even the slowest model, using the slowest framework, runs at 285 frames-per-second on CPU.

We note from manually observing the gold data that sometimes a gloss segment starts before the person actually begins signing, or moving at all. This means that our accuracy ceiling is not 100%. We did not perform a rigorous re-annotation of the dataset to quantify how extensive this problem is.

Model	**Points**	**Params**	**Dev Acc**	**Test Acc**	∂t **(sk)**	∂t **(tf)**	∂t **(pt)**
Linear-1	25	25	79.99%	79.93%	6.49μs	823μs	2.75μs
Linear-25	25	625	84.13%	83.79%	6.78μs	824μs	5.10μs
Linear-50	25	1,250	85.06%	83.39%	6.90μs	821μs	7.41μs
BBOX	8	18,818	87.49%	87.78%	—	3519μs	367μs
Pose-Hands	42	27,522	87.65%	88.05%	—	3427μs	486μs
Pose-Body	25	23,170	92.17%	90.35%	—	3437μs	443μs
Pose-All	137	51,842	92.31%	91.53%	—	3537μs	588μs

Table 1: Accuracy and inference-time (∂t) results for the various experiments.

6 Analysis

As we know that different pose landmarks have varying importance to the classification, we use the *Linear-1* model's coefficients magnitude to visualize how the different landmarks contribute. Figure 3 visualizes the average human pose in the dataset, with the opacity of every landmark being the absolute value of the coefficient.

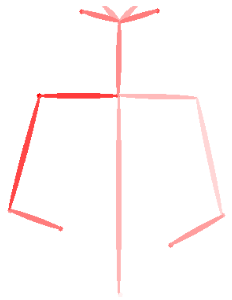

Fig. 3: The average pose in the dataset. The opacity of every landmark is determined by its coeffient in the *Linear-1* model.

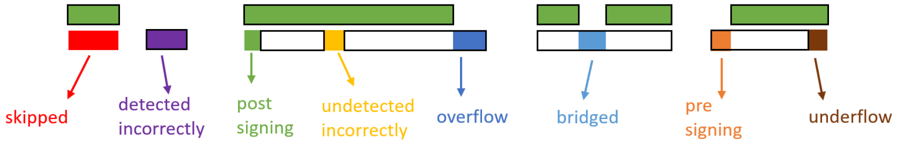

Fig. 4: Visualization of the different types of errors. The first row contains the gold annotations, and the second row contains a model's prediction.

First, we note that the model attributes no importance to any landmark below the waist. This makes sense as they both do not appear in all videos, and bare no meaning in sign language. The eyes and nose seem to carry little weight, while the ears carry more. We do not attribute this to any signing phenomenon.

Additionally, we note hands asymmetry. While both wrists have a high weight, the elbow and shoulder for the right hand carry more weights than their corresponding left counterparts. This could be attributed to the fact that most people are right handed, and that in some sign languages the signer must decide which hand is dominant in a consistent manner. We see this asymmetry as a feature of our model, and note that apps using our models could also include a "dominant hand" selection.

To further understand what situations our models capture, we check multiple properties of them on the test set. We start by generally noting that our data is conversational. 84.87% of the time, only one participant is signing, while 8.5% of the time both participants are signing, and in the remaining 6.63% of the time no one is signing, primarily when the participants are being instructed on the task.

Our test set includes 4,138 *signing* sequences with an average length of 11.35 seconds, and a standard deviation of 29.82 seconds. It also includes 4,091 *not-signing* sequences with an average length of 9.95 seconds, and a standard deviation of 24.18 seconds.

For each of our models, we compare the following error types (Figure 4):

- **Bridged**—Cases where the model bridged between two signing sections, still predicting the person to be *signing* while the annotation says they are not.
- **Signing Detected Incorrectly**—Cases where the model predicted a *signing* span fully contained within a *not-signing* annotation.
- **Signing Overflow**—Cases where signing was still predicted after a *signing* section ended.
- **Started Pre-Signing**—Cases where *signing* was predicted before a *signing* section started.
- **Skipped**—Cases where the model did not detect entire *signing* sections.
- **Signing Undetected Incorrectly**—Cases where the model predicted a *not-signing* span fully contained within a *signing* annotation.
- **Started Post-Signing**—Cases where the *signing* section started before it was predicted to start.

- **Signing Underflow**—Cases where the *signing* section was predicted to end prematurely.

	linear-1	**linear-25**	**linear-50**
Bridged	107 (0.10 ± 0.15)	308 (0.34 ± 0.40)	426 (0.45 ± 0.46)
Signing Detected Incorrectly	132151 (0.04 ± 0.07)	8773 (0.30 ± 0.81)	6594 (0.34 ± 1.06)
Signing Overflow	4094 (0.09 ± 0.15)	3893 (0.32 ± 0.43)	3775 (0.46 ± 1.17)
Started Pre-Signing	873 (0.09 ± 0.13)	345 (0.45 ± 0.68)	257 (0.88 ± 4.27)
Skipped	50 (1.41 ± 1.95)	298 (1.38 ± 1.43)	446 (1.49 ± 1.60)
Signing undetected incorrectly	219531 (0.05 ± 0.10)	26185 (0.27 ± 0.50)	18037 (0.32 ± 0.66)
Started Post-Signing	4199 (0.17 ± 0.23)	3951 (0.48 ± 0.57)	3803 (0.60 ± 0.77)
Signing Underflow	1677 (0.15 ± 0.26)	1092 (0.58 ± 0.91)	827 (0.71 ± 0.96)

	BBOX	**Pose-Hands**	**Pose-Body**	**Pose-All**
Bridged	754 (0.97 ± 1.94)	861 (1.26 ± 2.63)	747 (1.12 ± 2.35)	573 (0.75 ± 1.08)
Signing Detected Incorrectly	5697 (0.64 ± 1.93)	12919 (0.33 ± 1.33)	6286 (0.38 ± 1.29)	11384 (0.25 ± 1.14)
Signing Overflow	3337 (0.95 ± 2.10)	3230 (1.01 ± 2.46)	3344 (0.67 ± 1.29)	3518 (0.48 ± 0.87)
Started Pre-Signing	402 (1.33 ± 2.73)	558 (1.59 ± 5.15)	298 (1.48 ± 3.87)	408 (0.70 ± 1.97)
Skipped	199 (1.31 ± 1.40)	115 (1.45 ± 1.54)	243 (1.31 ± 1.30)	146 (1.41 ± 1.42)
Signing undetected incorrectly	4089 (0.48 ± 0.76)	3526 (0.26 ± 0.51)	4786 (0.32 ± 0.60)	5526 (0.23 ± 0.44)
Started Post-Signing	3939 (0.34 ± 0.44)	4023 (0.24 ± 0.34)	3895 (0.37 ± 0.49)	3992 (0.29 ± 0.36)
Signing Underflow	370 (0.82 ± 1.08)	297 (0.55 ± 0.68)	506 (0.63 ± 0.97)	666 (0.44 ± 0.66)

Table 2: We evaluate every model on the different error types, and show number of sequences with that error, including average sequence length in seconds and standard deviation.

Table 2 includes the number of sequences, including average length and standard deviation in seconds, for each of the error types. Most notably, we see that the less context the model has, the more sporadic its predictions and thus it will generally completely bridge or skip less sequences. The same locality however introduces many signing detected / undetected incorrectly errors, albeit of short lengths.

In the sequential models, we generally see a lower number of sequences as they can incorporate global features in the classification. As indicated by the accuracy scores, we see fewer errors of most types the more diverse the input points are, with one notable exception for the *Pose-All* model which underperforms *Pose-Body* on all errors except for *Bridged* and *Skipped*.

7 Demo Application

With this publication, we release a demo application working in the browser for computers and mobile devices. Pragmatically, we choose to use the "Pose-Body" model variant, as it performs almost on par with our best model, "Pose-All", and we find it is feasible to acquire the human body poses in real-time with currently available tools.

We use PoseNet [22, 21] running in the browser using TensorFlow.js [30]. PoseNet includes two main image encoding variants: MobileNet [15], which is a lightweight model aimed at mobile devices, and ResNet [13], which is a larger model that requires a dedicated GPU. Each model includes many sub-variants with different image resolution and convolutional strides, to further allow for tailoring the network to the user's needs. In our demo, we first tailor a network to the current computation device to run at least at 25fps. While using a more lightweight network might be faster, it might also introduce pose estimation errors.

The pose estimation we use only returns 17 points compared to the 25 of OpenPose; hence, we map the 17 points to the corresponding indexes for OpenPose. We then normalize the body pose vector by the mean shoulder width the person had in the past 50 frames in order to disregard camera resolution and distance of the signer from the camera.

Onward, there are two options: either send the pose vector to the videoconferencing server where inference could be done or perform the inference locally. As our method is faster than real-time, we chose the latter and perform inference on the device using TensorFlow.js. For every frame, we get a signing probability, which we then show on the screen.

In a production videoconferencing application, this signing probability should be streamed to the call server, where further processing could be done to show the correct people on screen. We suggest using the signing probability as a normalized "volume", such that further processing is comparable to videoconferencing users using speech.

While this is the recommended way to add sign language detection to a videoconferencing app, as the goal of this work is to empower signers, our demo application can trigger the speaker detection by transmitting audio when the user is signing. Transmitting ultrasonic audio at 20KHz, which is inaudible for humans, manages to fool Google Meet, Zoom and Slack into thinking the user is speaking, while still being inaudible. One limitation of this method is that videoconferencing app developers can crop the audio to be in the audible human range and thus render this application useless. Another limitation is that using high-frequency audio can sound crackly when compressed, depending on the signer's internet connection strength.

Our model and demo, in their current forms, only allow for the detection of a single signer per video stream. However, if we can detect more than a single person, and track which poses belong to which person in every frame, there is no limitation to run our model independently on each signer.

8 Discussion

8.1 Limitations

We note several limitations to our approach. The first is that it relies on the pose estimation system to run in real-time on any user's device. This proves to be challenging, as even performing state-of-the-art pose estimation on a single frame on a GPU with OpenPose [5, 6] can take upwards of 300ms, which introduces two issues: (1) If in order to get the optical-flow, we need to pose two frames, we create a delay from when a person starts signing to when they could be accurately detected as signing, equal to at least two times the pose processing time. (2) Running this on mobile devices or devices without hardware acceleration like a GPU may be too slow.

As we only look at the input's optical flow norm, our model might not be able to pick up on times when a person is just gesturing rather than signing. However, as this approach is targeted directly at sign language users rather than the general non-signing public, erring on the side of caution and detecting any meaningful movements is preferred.

8.2 Demographic Biases

The data we use for training was collected from various regions of Germany, with equal number of males and females, as well as an equal number of participants from different age groups [26]. Although most of the people in the dataset are European white, we do not attribute any significance between the color of their skin to the performance of the system, as long as the pose estimation system is not biased.

Regardless of age, gender, and race, we do not address general ethnic biases such as different communities of signers outside of Germany signing differently - whether it is the size, volume, speed, or other properties.

9 Conclusions

We propose a simple human optical-flow representation for videos based on pose estimation to perform a binary classification per frame — is the person signing or not. We compare various possible inputs, such as full-body pose estimation, partial pose estimation, and bounding boxes and contrast their acquisition time in light of our targeted real-time videoconferencing sign language detection application.

We demonstrate our approach on the Public DGS Corpus (German Sign Language), and show results of 87%-91% prediction accuracy depending on the input, with per-frame inference time of $350 - 3500\mu$s.

References

1. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X.: TensorFlow: Large-scale machine learning on heterogeneous systems (2015), http://tensorflow.org/, software available from tensorflow.org
2. Apple: WWDC (World Wide Developer Conference) (2020), https://developer.apple.com/wwdc20/
3. Borg, M., Camilleri, K.P.: Sign language detection "in the wild" with recurrent neural networks. In: ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 1637–1641. IEEE (2019)
4. Bragg, D., Koller, O., Bellard, M., Berke, L., Boudreault, P., Braffort, A., Caselli, N., Huenerfauth, M., Kacorri, H., Verhoef, T., et al.: Sign language recognition, generation, and translation: An interdisciplinary perspective. In: The 21st International ACM SIGACCESS Conference on Computers and Accessibility. pp. 16–31 (2019)
5. Cao, Z., Hidalgo Martinez, G., Simon, T., Wei, S., Sheikh, Y.A.: Openpose: Realtime multi-person 2d pose estimation using part affinity fields. IEEE Transactions on Pattern Analysis and Machine Intelligence (2019)
6. Cao, Z., Simon, T., Wei, S.E., Sheikh, Y.: Realtime multi-person 2d pose estimation using part affinity fields. In: CVPR (2017)
7. Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using rnn encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078 (2014)
8. Cihan Camgoz, N., Hadfield, S., Koller, O., Ney, H., Bowden, R.: Neural sign language translation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 7784–7793 (2018)
9. Cui, R., Liu, H., Zhang, C.: Recurrent convolutional neural networks for continuous sign language recognition by staged optimization. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 7361–7369 (2017)
10. Gebre, B.G., Wittenburg, P., Heskes, T.: Automatic sign language identification. In: 2013 IEEE International Conference on Image Processing. pp. 2626–2630. IEEE (2013)
11. Hanke, T., Schulder, M., Konrad, R., Jahn, E.: Extending the Public DGS Corpus in size and depth. In: Proceedings of the LREC2020 9th Workshop on the Representation and Processing of Sign Languages: Sign Language Resources in the Service of the Language Community, Technological Challenges and Application Perspectives. pp. 75–82. European Language Resources Association (ELRA), Marseille, France (May 2020), https://www.aclweb.org/anthology/2020.signlang-1.12
12. Hatta, T., Dimond, S.J.: Differences in face touching by japanese and british people. Neuropsychologia **22**(4), 531–534 (1984)
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
14. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997)

15. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017)
16. Isaacs, J., Foo, S.: Hand pose estimation for american sign language recognition. In: Thirty-Sixth Southeastern Symposium on System Theory, 2004. Proceedings of the. pp. 132–136. IEEE (2004)
17. Konstantinidis, D., Dimitropoulos, K., Daras, P.: Sign language recognition based on hand and body skeletal data. In: 2018-3DTV-Conference: The True Vision-Capture, Transmission and Display of 3D Video (3DTV-CON). pp. 1–4. IEEE (2018)
18. Kwok, Y.L.A., Gralton, J., McLaws, M.L.: Face touching: A frequent habit that has implications for hand hygiene. American journal of infection control **43**(2), 112–114 (2015)
19. Monteiro, C.D., Mathew, C.M., Gutierrez-Osuna, R., Shipman, F.: Detecting and identifying sign languages through visual features. In: 2016 IEEE International Symposium on Multimedia (ISM). pp. 287–290. IEEE (2016)
20. Nicas, M., Best, D.: A study quantifying the hand-to-face contact rate and its potential application to predicting respiratory tract infection. Journal of occupational and environmental hygiene **5**(6), 347–352 (2008)
21. Papandreou, G., Zhu, T., Chen, L.C., Gidaris, S., Tompson, J., Murphy, K.: Personlab: Person pose estimation and instance segmentation with a bottom-up, part-based, geometric embedding model. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 269–286 (2018)
22. Papandreou, G., Zhu, T., Kanazawa, N., Toshev, A., Tompson, J., Bregler, C., Murphy, K.: Towards accurate multi-person pose estimation in the wild. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4903–4911 (2017)
23. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: Wallach, H., Larochelle, H., Beygelzimer, A., dÁlché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems 32, pp. 8024–8035. Curran Associates, Inc. (2019), http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf
24. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011)
25. Poizner, H., Bellugi, U., Lutes-Driscoll, V.: Perception of american sign language in dynamic point-light displays. Journal of experimental psychology: Human perception and performance **7**(2), 430 (1981)
26. Schulder, M., Blanck, D., Hanke, T., Hofmann, I., Hong, S.E., Jeziorski, O., König, L., König, S., Konrad, R., Langer, G., Nishio, R., Rathmann, C.: Data statement for the Public DGS Corpus. Project Note AP06-2020-01, DGS-Korpus project, IDGS, Hamburg University, Hamburg, Germany (2020)
27. Schulder, M., Hanke, T.: OpenPose in the Public DGS Corpus. Project Note AP06-2019-01, DGS-Korpus project, IDGS, Hamburg University, Hamburg, Germany (2019). https://doi.org/10.25592/uhhfdm.842
28. Simon, T., Joo, H., Matthews, I., Sheikh, Y.: Hand keypoint detection in single images using multiview bootstrapping. In: CVPR (2017)

29. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
30. Smilkov, D., Thorat, N., Assogba, Y., Yuan, A., Kreeger, N., Yu, P., Zhang, K., Cai, S., Nielsen, E., Soergel, D., et al.: Tensorflow. js: Machine learning for the web and beyond. arXiv preprint arXiv:1901.05350 (2019)
31. Sperling, G., Landy, M., Cohen, Y., Pavel, M.: Intelligible encoding of asl image sequences at extremely low information rates. Computer vision, graphics, and image processing **31**(3), 335–391 (1985)
32. Zafrulla, Z., Brashear, H., Starner, T., Hamilton, H., Presti, P.: American sign language recognition with the kinect. In: Proceedings of the 13th international conference on multimodal interfaces. pp. 279–286 (2011)