SIGN LANGUAGE DETECTION "IN THE WILD" WITH RECURRENT NEURAL NETWORKS

Mark Borg, Kenneth P. Camilleri

Systems and Control Engineering, University of Malta, Msida, Malta

ABSTRACT

We propose a multi-layer RNN for sign language detection. The system uses features extracted automatically from a 2-stream convolutional neural network (CNN) that takes video image data and motion data as input. We also created a dataset of videos containing signing "in the wild" to be used for training and evaluation purposes. We compare our system against the state-of-the-art, and attain an improvement of around 18%, indicating that our network is able to leverage dynamic information of hand motion during detection.

***Index Terms*—** sign language detection, RNN, CNN

1. INTRODUCTION

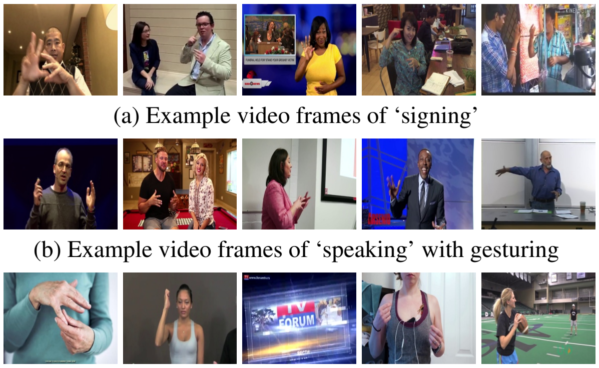
Sign languages are the main, and oftentimes the only, method of communication used by deaf communities around the world. These visual languages exhibit rich linguistic structure, and primarily convey semantic meaning via the location, shape, and dynamic motion of the hands.

Automatic sign language recognition (ASLR) provides quite a number of challenges to the fields of computer vision and image processing. And although great strides have been made in recent years, sign language technologies in general still lag far behind speech technologies. For example, ASLR is typically attempted within well-defined settings and under pre-determined constraints, like having a single signer with known orientation and position. Further progress is needed until sign language technologies become robust enough to be able to assist the Deaf community in overcoming communication barriers that exist both in the physical world and in the digital world.

One such area that merits further attention is *sign language detection*, the process of identifying whether a video (or video segment) from a generic and unconstrained video collection contains signing or not. Potential applications include automatic tagging and categorisation of videos, as a first step towards auto-captioning of sign videos, as well as for the automatic initialisation of ASLR rather than relying on pre-determined assumptions about the input videos.

2. RELATED WORK

Much of the current work on sign language detection utilise face detection algorithms to identify region of interest (ROI) in an image, where the system could look for telltale patterns of hand motions associated with signing. For example, Monteiro et al. [1] perform background subtraction in the ROIs, and then compute simple visual features that describe the foreground pixels and their evolution in time. Shipman et al. [2, 3] also apply background subtraction, but then extract polar motion profiles from the ROIs. These describe foreground pixels in terms of their distributions of distances and polar orientations from the centre of the face. A limitation of these methods is that the detection of signing activity depends on the success rate of face detection. Gebre et al. [4] try to mitigate this by using skin detection in addition to face detection, while an ensemble of face detectors is used in [3]. As regards to the classifier, support vector machines (SVMs) are the most popular [1, 5, 2], followed by random forests [4], and k-nearest neighbour (kNN) [6]. In contrast to the field of generic video action recognition, where deep learning techniques are widely used, in the area of sign language detection, limited use has been made, probably due to the lack of large datasets needed for training such systems. Gebre at al. [7] use a sparse autoencoder and a 3D CNN for the identification of a number of sign languages. 3D CNNs perform convolution operations across 3 dimensions, by operating on a batch of video frames at one go. One limitation is the increase in complexity (parameters) due to the extra kernel dimension, and hence harder to train.


(a) Example video frames of 'signing'

(b) Example video frames of 'speaking' with gesturing

(c) Example video frames of class 'other'

Fig. 1. Sample video frames from our "signing in the wild" dataset. The two classes shown in (b) & (c) were chosen because of their potential for confusion with (a) signing videos.

In the rest of this paper we describe our contributions: (1) we propose an RNN based system to improve on the state-of-the-art, (2) we make use of a CNN to automatically extract strong features from image frames, in order to push the boundary on the type of videos that can be used in sign language detection, and (3) we introduce our dataset, "Signing in the Wild", which we make public, to aid the training and evaluation of such systems.

3. DATASET

Publicly-available datasets for generic video action recognition, like AVA [8] and THUMOS [9], do not contain signing as one of their

Thanks to Marc Tanti for the discussions on recurrent neural networks (RNNs). Also thanks to the Biomedical Systems and Control Engineering Lab for making available their GPU server for the experimental runs of this paper, and thanks to Jean Gauci for assisting in the use of the said resources.

978-1-5386-4658-8/18/$31.00 ©2019 IEEE

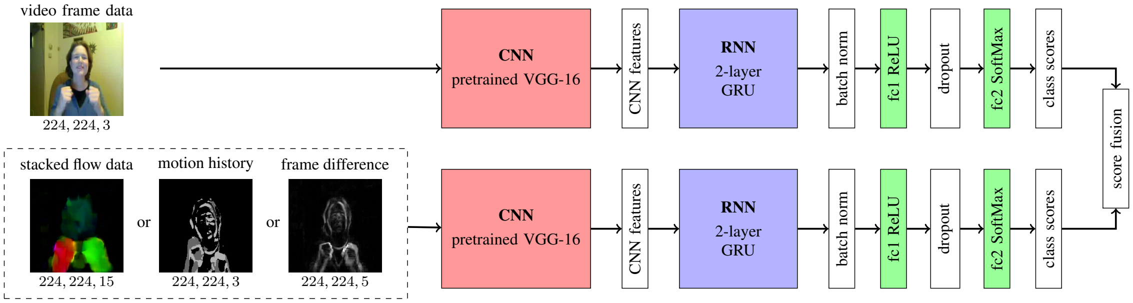


Fig. 2. Our proposed sign language detection framework using features extracted via CNNs from raw video and motion data in parallel. We also employ an RNN to leverage temporal information within video segments, followed by a non-linear classifier.

activity classes. And research works focusing specifically on sign language detection [1, 5, 2, 10], have used datasets that, to the best of our knowledge, have not been made public. We checked ASLR datasets, such as SIGNUM [11] and RWTH-Phoenix [12]. While such datasets could be used for sign language detection, most are acquired under constrained conditions, to ensure an optimal view of the signer. Such datasets do not meet our requirements for signing "in the wild", and we thus opted to create our own dataset.

The videos in our dataset are harvested from YouTube. We used a number of keywords to search for as wide a variety of sign videos as possible, resulting in a rich collection of videos of different sign languages, single and multiple signers, natural signing, complex camera and signer motion, etc.

For the negative set, we created two classes of videos, labelled 'speaking' and 'other'. Our motivation for the 'speaking' class is that speech is often accompanied by hand gestures (gesticulation), which can be easily confused with signing. Signing can be discriminated by its linguistic nature, i.e., its distinct phonological, morphological and categorical (discrete) structures, while gesticulation tends to be more spontaneous, idiosyncratic and analogue in nature. But recent research [13, 14] is showing that the distinction between the two might be more blurred: signers gesture just as speakers do, and the gesticulation by signers is imprinted upon the more structured signing. Another similarity is that signers make use of mouthings (mouth shapes and movements), which can exhibit visual similarities to the mouth movements in speech.

For the 'other' class, we looked for distractors to both 'signing' and 'speaking', i.e., videos containing hand movements that are quite similar to signing/gesticulation and thus might confuse a classifier. Examples include: miming, hand exercises, various manual activities like playing instruments, painting, writing, yoga and martial arts, sports like table tennis, etc. Also included are activities similar to speech, like people laughing, clapping, nodding, listening to other speakers, etc. Figure 1 shows examples from our dataset.

A total of 1120 videos are included in our dataset, each video contributing the first 6.6 minutes, resulting in 2000 frames per video when sampled at 5Hz. We have 1.45 million video frames in total.

Our videos are *untrimmed*, i.e., a video can contain multiple activities, background scenes, scene cuts, and other actions done by the same or different actors. Thus the videos are unconstrained both spatially and also temporally. This is in line with recent trends in video action recognition [8], and unlike ASLR datasets where trimmed videos are the norm. In particular, several videos in our dataset contain all 3 classes (occasionally with temporal overlap), and sometimes the same person alternating between signing and speaking.

We performed manual groundtruthing at video frame level. Since action boundaries can be inherently fuzzy, we consider a short temporal context (10 frames) surrounding the frame to be labelled in order to decide on its class label. We also adopt certain spatial guidelines, e.g. mouth movements must be visible for action 'speaking', thus eliminating distant views and when the speaker turns his/her back to the camera. Ambiguous cases are left unlabelled. We annotate video segments that do not contain signing or speaking as 'other', including opening/closing credits, title screens, scene transitions, animations, background scenes, etc.

One limitation of the groundtruthing of our dataset is that it has been performed by a single person, thus prone to subjectivity. We acknowledge this limitation and we hope to address this issue in a future release of the dataset by incorporating multiple annotators. In total, we annotated 1.23 million out of 1.45 million video frames. We make our dataset publicly available at `https://github.com/mark-borg/Signing-in-the-Wild-dataset`.

4. OUR APPROACH

In contrast to existing sign language detection methods [1, 2, 3, 5, 7, 15, 16, 17], which make use of hand-crafted features as input to their classifier, we employ features extracted from a CNN. We use the VGG-16 network [18] for feature extraction. We select this network because it offers a good compromise on layer depth, and because it has proven itself to be quite successful when applied to a variety of domains and applications including video activity recognition [19].

Since a main discriminating characteristic of signing is its dynamic hand motions, we adopt a two-stream CNN approach [20]. Apart from extracting CNN features from the raw image data on a per-frame basis, we augment this with a second CNN (working in parallel) that extracts features from instantaneous motion data. Optical flow is the typical choice when it comes to selecting the type of motion data used in the second CNN stream [20, 21, 22, 23, 24]. In our work we experiment with other types of motion data, like motion history image (MHI) and multi-frame differencing, so as to find a good compromise between accuracy and computational efficiency. The CNN features extracted from the image data and motion data are then fed to a RNN, in order to leverage the temporal information present in video segments. We choose a gated version as our RNN, called gated recurrent unit (GRU) [25]. A GRU can use its internal state (memory) to persist information across a short time period, allowing it to learn dynamic patterns in the CNN features.

We complete our system with a 2-layer fully-connected classifier, plus layers for batch normalisation and dropout, to help reduce over-fitting. The final class label is obtained via decision-based fusion. Figure 2 illustrates our proposed framework; shaded layers are the trainable ones. In the following sections, we will describe the components of our system. Then we describe our experiments, followed by an evaluation and comparison against the state-of-the-art.

4.1. CNN features

We use the VGG-16 implementation provided in Keras [26] (denoted as Configuration D in the original paper [18]), with pre-trained weights on ImageNet. VGG-16 has 13 convolutional and 3 fully connected layers, with 138 million trainable parameters, and it uses filters with small receptive fields (3×3) in all the layers.

We experimented with extracting CNN features both from the last convolution layer of VGG-16 ('block5_conv3'), as well as from the first fully-connected layer ('fc1'). The former yields a feature map of size $7\times7\times512$ $(=25088)$ after max pooling, while the latter produces a more compact set of 4096 CNN features. No fine-tuning of the CNN is done. Input frame resolution is set to 224×224 pixels.

4.2. Two-stream CNN

In two-stream CNN approaches, RGB data and motion data (typically optical flow), are each processed in parallel by a CNN with identical architecture [20]. In our case we use the VGG-16 network for both streams. We use Farnebäck's algorithm [27] for computing dense optical flow. Apart from optical flow, we also investigate other types of motion data, mainly MHI [28], and multi-frame differencing, since the last two are more computationally efficient. We use a 5-frame temporal window for MHI and multi-frame differencing, with the frames sampled at the original video frame rate.

Since motion data exhibits strong differences in its distribution from RGB data, one would naturally expect that a CNN model like VGG-16 needs to be trained from scratch for motion data. Conspiring against this is the limited training data available, compared to the original ImageNet dataset. As a result we decided to use a pretrained VGG-16 for the motion data stream. Backing our decision are the findings of Yosinski et al. [29], demonstrating that applying transfer learning even from a distant task (unrelated data) is still better than training from scratch. We adopt an approach similar to Wang et al's cross modality pre-training method [30] for preparing optical flow data to be fed into the VGG-16 network. The optical flow field is represented as a 3-channel RGB image with the angle of the flow vector determining the chrominance value and the flow vector magnitude defining the luminance. Ten consecutive optical flow fields are stacked together, creating an input of size $224 \times 224 \times 30$. The filter weights of the first convolution layer of VGG-16 ('block1_conv1') are then replicated to handle the extra channels in the input data.

A similar approach is used for feeding a stack of 5 multi-frame difference images to VGG-16, with the addition that the filter weights are first averaged across the 3 channels before replication, due to the difference images being single channel. Because of time constraints at the time of paper submission, we did not attempt any fine-tuning of the VGG-16 CNN, for any of the two streams.

4.3. RNN and GRU

RNNs are the networks of choice when it comes to handling sequences of data where the temporal dynamics connecting them is highly important. But a traditional RNN suffers from a number of problems during training, including that of vanishing gradients [31]. Gated versions of RNNs, such as long short-term memory (LSTM) [32] and GRU [25], address and minimise such training problems.

We evaluate both LSTMs and GRUs for sign language detection, and opt for GRU over LSTM, mainly due to its simpler structure (2 gates per cell instead of 3), the need for less training data, and that a GRU can achieve this without sacrificing accuracy. We also investigate the use of stacked RNNs with 2 or more layers, since these are able to perform hierarchical processing on temporal data. Figure 3 shows an unrolled version of our RNN network.

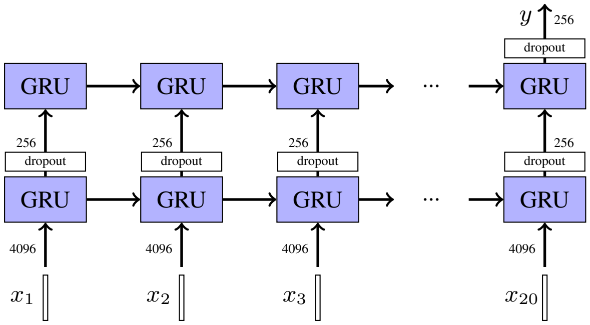


Fig. 3. A 2-layer RNN network, with 256 hidden units and 20 timesteps. x_i are the input CNN features; y is the RNN output.

5. EXPERIMENTS & EVALUATION

5.1. Experiments

In our experiments we use the dataset described in §3, partitioning it into 5 folds, 4 for training and 1 for validation, taking care that all labelled frames of each video appear in the same fold only. We use video segments of 20 frames throughout all of our experiments.

We train the RNN for 500 epochs, using early stopping based on validation cross-entropy loss (if no improvement over 10 epochs). Mini-batch stochastic gradient descent is used, with Adam as the optimisation algorithm [33]. We employ a training schedule inspired by the findings and recommendations of [34, 35]. We start with a fixed mini-batch size of 32 and a learning rate of 0.001, reducing the learning rate each time the validation loss stops improving for that learning rate value. Once validation loss stops decreasing no matter how much we reduce the learning rate, we increase the batch size and use the learning rate value that performed best in the previous step. This process is repeated till no further improvement in loss is observed regardless of batch size increase. We make our code available here: `https://github.com/mark-borg/sign-language-detection`.

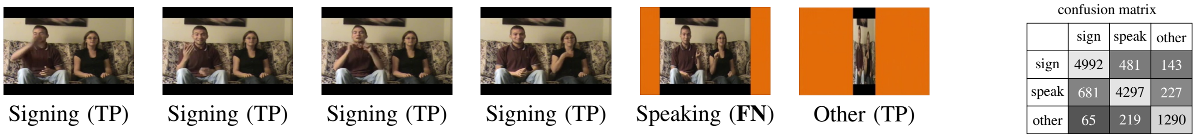
5.2. Evaluation

We start by comparing CNN features extracted from the 'block5_conv3' layer against those taken from 'fc1'. Results of Table 1 show that features extracted from VGG-16 layer 'fc1' perform better, in addition to being more efficient in size.

In another experiment we examine the accuracy and computational time of the different types of motion data given in §4.2, both in conjunction with the video data (i.e., as a 2-stream CNN, one video stream plus one motion stream), but we also evaluate each type of motion stream on its own, in order to assess its individual contribution. Results are given in Table 2. Surprisingly, just using motion

Table 1. Results obtained when using different CNN features

CNN layer	Feature size	Loss (validation set) ↓
VGG-16 block5_conv3	25088	0.6681
VGG-16 fc1	4096	**0.5037**


Signing (TP) Signing (TP) Signing (TP) Signing (TP) Speaking (**FN**) Other (TP)

confusion matrix

	sign	speak	other
sign	4992	481	143
speak	681	4297	227
other	65	219	1290

Fig. 4. Qualitative results and confusion matrix for our classifier. In this video, two persons are signing (frames 1 to 4 above), followed by a video transition effect (horizontal shrink, last 2 frames). Note that the fifth frame is classified incorrectly: 'speaking' instead of 'signing'.

Table 2. Motion stream & fusion results for 2-stream CNN

Modality	Loss ↓	Accuracy ↑	Time (ms) ↓
RGB stream only	0.5128	85.01%	–
optical flow only	**0.5387**	83.12%	57.8
MHI only	0.5445	83.67%	17.1
multi-frame diff. only	0.5738	84.08%	**9.7**
RGB stream + optical flow stream	–	**87.67%**	–
RGB stream + MHI stream	–	87.60%	–
RGB stream + frame diff. stream	–	85.61%	–

Table 3. Results of various RNN architectures

RNN	layers	trainable parameters	Loss (valid. set) ↓
LSTM	1	4,474,627	0.5144
LSTM	2	4,999,939	0.6413
LSTM	3	5,525,251	0.5714
GRU	1	3,360,259	0.5267
GRU	2	3,754,243	**0.5037**
GRU	3	4,148,227	0.6028

data alone can still yield a high detection rate. Also noteworthy is that MHI performs nearly as well as optical flow, while being more computationally efficient. Finally, Table 3 gives the results obtained when evaluating different types and stacked-layer depths of RNNs. In these experiments we fix the number of hidden units in the RNNs to 256 for all layers, and the number of timesteps is 20, the video segment length. We can observe that a 2-layer stacked GRU performs the best, with the added advantage that the model has less parameters; this is the model of Figure 3.

5.2.1. *Ablation studies*

Once we determined the best architecture from several model designs, we performed ablation studies on the model: do all the layers contribute to the final outcome of the model? Can the model be simplified further without decreasing accuracy? Due to time constraints, for these ablation studies we only used 2 folds from the dataset – one for training, another for validation. Table 4 lists the main results. Dropout appears to be a critical element in our proposed model, both as part of the fully-connected classifier as well as within the stacked GRU layers. Table 4 shows loss for different dropout rates.

5.2.2. *Comparison with the state-of-the-art*

Next we compare our system against the work of Shipman et al. [2, 3], currently the state-of-the-art in sign language detection. They report an F1 score of 78% (precision of 83%) against a dataset of 227 YouTube videos (111 sign and 116 non-sign). Since their dataset is not publicly available and in order to perform as fair a comparison as possible, we implemented their system as a baseline against which to evaluate our proposed solution, both running against our dataset.

Table 4. Ablation studies on the proposed RNN network

Model settings		Cross-entropy loss on validation set ↓					
proposed model		**0.504**					
no batch normalisation		0.609		(≈ 20% increase in loss)			
no dropout layer		0.715		(≈ 42% increase in loss)			
no GRU dropout		0.693		(≈ 38% increase in loss)			
no classifier fc1 layer		0.649		(≈ 29% increase in loss)			
with dropout layer	rate:	0.1	0.2	0.3	**0.4**	0.5	0.6
	loss:	0.605	0.577	0.575	**0.504**	0.511	0.602
with GRU dropout	rate:	0.1	0.2	**0.3**	0.4	0.5	0.6
	loss:	0.628	0.601	**0.548**	0.649	0.554	0.552

Table 5. Comparison with the state of the art

Method	Feature type & Classifier	Loss ↓	Precision ↑
baseline method [2, 3]	hand-crafted features + SVM	1.114	69.23%
baseline+RNN	hand-crafted features + RNN	0.841	78.02%
CNN+SVM	2-stream CNN features + SVM	–	79.15%
our method	2-stream CNN features + RNN	**0.573**	**87.67%**

Table 5 gives the results. Our method outperforms the baseline by ≈ 18%. This result shows that by combining strong features from a CNN together with the temporal abilities of an RNN, a substantial improvement in recognition can be attained on quite a difficult dataset. To assess the RNN's contribution on its own, we combine the hand-crafted features of the baseline method with our RNN network – termed 'baseline+RNN' in Table 5. We also evaluate the CNN features when combined with an SVM instead of RNN (method 'CNN+SVM' in Table 5). We can see that, for this dataset, an RNN improves recognition by ≈9% when compared to the SVM of the baseline method, brought about by the RNN's ability to leverage the dynamic information contained in signing. Finally Figure 4 shows some qualitative results and confusion matrix.

6. CONCLUSION

We present a sign language detection system comprised of CNN and RNN components. A 2-stream CNN processes image data and motion data (optical flow, MHI, and multi-frame differencing) to extract strong features. These are then fed to an RNN network, made up of stacked GRU layers, in order to leverage the dynamic information contained in signing videos. To evaluate our proposed system, we construct a dataset we call "Signing in the Wild", harvested from YouTube videos, and adding challenging distractors such as speaking and gesticulation videos. We compare our system with the state-of-the-art in sign language detection and obtain a substantial improvement in recognition (≈ 18%). Future work could involve localising the signer(s) in the video frames, identifying specific sign languages, and sign language constructs like fingerspelling.

7. SUPPLEMENTARY MATERIAL

`https://github.com/mark-borg/sld-supp.pdf`

8. REFERENCES

[1] C. Monteiro, R. Gutierrez-Osuna, and F. M. Shipman, "Design and Evaluation of Classifier for Identifying Sign Language Videos in Video Sharing Sites," in *SIGACCESS*, 2012.

[2] F. M. Shipman, R. Gutierrez-Osuna, T. Shipman, C. Monteiro, and V. Karappa, "Towards a distributed digital library for sign language content," in *Proc. .15th ACM/IEEE-CS Joint Conference on Digital Libraries*, 2015, JCDL '15, pp. 187–190.

[3] F. M. Shipman, S. Duggina, C. Monteiro, and R. Gutierrez-Osuna, "Speed-Accuracy Tradeoffs for Detecting Sign Language Content in Video Sharing Sites," in *Proc. ACM SIGACCESS*. 2017, ASSETS '17, pp. 185–189, ACM.

[4] B. G. Gebre, P. Wittenburg, and T. Heskes, "Automatic sign language identification," in *2013 IEEE Int. Conf. on Image Processing*, 2013, pp. 2626–2630.

[5] V. Karappa, C. Monteiro, F. M. Shipman, and R. Gutierrez-Osuna, "Detection of sign-language content in video through polar motion profiles," in *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 1290–1294.

[6] P. Yanovich, C. Neidle, and D. Metaxas, "Detection of Major ASL Sign Types in Continuous Signing For ASL Recognition," in *Proc. LREC 2016*, Paris, France, may 2016, ELRA.

[7] B.G. Gebre, O. Crasborn, P. Wittenburg, S. Drude, and T. Heskes, "Unsupervised Feature Learning for Visual Sign Language Identification," in *Proc. 52nd Annual Meeting of the Association for Computational Linguistics*, 2014, vol. 2.

[8] C. Gu, C. Sun, S. Vijayanarasimhan, et al., "AVA: A Video Dataset of Spatio-temporally Localized Atomic Visual Actions," in *CVPR 2018*, 2018.

[9] H. Idrees, A.R. Zamir, Y.G. Jiang, et al., "The THUMOS challenge on action recognition for videos "in the wild"," *Computer Vision and Image Understanding*, vol. 155, 2017.

[10] C. Zhang and Y. Tian, "Automatic Video Captioning via Multi-channel Sequential Encoding," in *Proc. ECCV 2016*, Cham, 2016, pp. 146–161, Springer.

[11] U. von Agris, J. Zieren, U. Canzler, B. Bauer, and K-F. Kraiss, "Recent developments in visual sign language recognition," *Universal Access in the Information Society*, vol. 6, 2008.

[12] J. Forster, C. Schmidt, O. Koller, M. Bellgardt, and H. Ney, "Extensions of the Sign Language Recognition and Translation Corpus RWTH-PHOENIX-Weather," in *LREC*, 2014.

[13] C. Müller, "Gesture and Sign: Cataclysmic Break or Dynamic Relations?," *Frontiers in Psychology*, vol. 9, pp. 1651, 2018.

[14] S. Goldin-Meadow and D. Brentari, "Gesture, sign, and language: The coming of age of sign language and gesture studies," *The Behavioral and Brain Sciences*, pp. 1–59, 2017.

[15] F. M. Shipman, R. Gutierrez-Osuna, and C. Monteiro, "Identifying sign language videos in video sharing sites," *ACM Trans. Access. Comput.*, vol. 5, no. 4, pp. 9:1–9:14, 2014.

[16] C. Monteiro, C. M. Mathew, R. Gutierrez-Osuna, and F. Shipman, "Detecting and Identifying Sign Languages through Visual Features," in *2016 IEEE Int. Symposium on Multimedia (ISM)*, 2016, pp. 287–290.

[17] C. Monteiro and F.M. Shipman and R. Gutierrez-Osuna, "Comparing Visual, Textual, and Multimodal Features for Detecting Sign Language in Video Sharing Sites," in *IEEE Multimedia Information Processing and Retrieval (MIPR)*, 2018, pp. 7–12.

[18] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *CoRR*, vol. abs/1409.1556, 2014.

[19] A. Canziani, A. Paszke, and E. Culurciello, "An Analysis of Deep Neural Network Models for Practical Applications," *CoRR*, vol. abs/1605.07678, 2016.

[20] K. Simonyan and A. Zisserman, "Two-Stream Convolutional Networks for Action Recognition in Videos," in *Advances in Neural Information Processing Systems 27*, pp. 568–576. Curran Assoc., Inc., 2014.

[21] J. Donahue, L.A. Hendricks, M. Rohrbach, S. Venugopalan, S. Guadarrama, K. Saenko, and T. Darrell, "Long-Term Recurrent Convolutional Networks for Visual Recognition and Description," *IEEE PAMI*, vol. 39, no. 4, pp. 677–691, 2017.

[22] 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 2017*, 2017, pp. 4724–4733.

[23] H. Ye, Z. Wu, R.-W. Zhao, X. Wang, Y.-G. Jiang, and X. Xue, "Evaluating Two-Stream CNN for Video Classification," in *Proc. 5th ACM on International Conference on Multimedia Retrieval*. 2015, ICMR '15, pp. 435–442, ACM.

[24] O. Köpüklü, N. Kose, and G. Rigoll, "Motion Fused Frames: Data Level Fusion Strategy for Hand Gesture Recognition," in *IEEE Conf on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2018.

[25] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation," in *Proc. Conf. Empirical Methods in Natural Language Processing (EMNLP)*, 2014.

[26] F. Chollet et al., "Keras," `https://keras.io`, 2015.

[27] G. Farnebäck, "Two-frame Motion Estimation Based on Polynomial Expansion," in *Proc. 13th Scandinavian Conference on Image Analysis*, 2003, SCIA'03, pp. 363–370.

[28] J.W. Davis and G.R. Bradski, "Motion segmentation and pose recognition with motion history gradients," in *WACV00*, 2000.

[29] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?," in *Advances in Neural Information Processing Systems 27 (NIPS '14)*, 2014.

[30] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool, "Temporal Segment Networks: Towards Good Practices for Deep Action Recognition," in *Computer Vision – ECCV 2016*, Cham, 2016, pp. 20–36.

[31] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Trans Neural Networks*, pp. 157–166, 1994.

[32] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[33] D.P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *ICLR 2015*, 2015, p. 13.

[34] D. Masters and C. Luschi, "Revisiting Small Batch Training for Deep Neural Networks," *CoRR*, 2018.

[35] S.L. Smith, P.-J. Kindermans, and Q.V. Le, "Don't Decay the Learning Rate, Increase the Batch Size," in *Int. Conf. on Learning Representations*, 2018.