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

SUPPLEMENTARY MATERIAL

Mark Borg, Kenneth P. Camilleri

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

Contents

1. INTRODUCTION

This document contains supplementary material to the paper with the same name, submitted to ICASSP. Section 2 provides further detail on the dataset we created for signing "in the wild". Section 3 contains an additional comparison between the two different set of features we extracted from the VGG-16 convolutional neural network (CNN). Section 4 contains some additional diagrams of our proposed model combining a 2-stream CNN with a recurrent neural network (RNN). And in Section 5 we provide more extensive examples of detection results.

2. DATASET

2.1. Video Selection

We make use of the YouTube Data API service [1] to search for videos on YouTube using a set of keywords (search terms). Table 1 lists the main keywords used to search for sign language videos, videos with speaking activity, as well as distractor videos. For sign languages, we also searched with commonly-used abbreviations for particular sign languages like American Sign Language (ASL), British Sign Language (BSL), and others. For lesser known sign languages, we ran search queries with their full names, using the list of sign languages available on Wikipedia [2], e.g., "lingua gestual Portuguesa". Using an API, it is easy to automate such query calls programmatically. We use Python as our main development language and YouTube provides a Python library for this purpose. Our code is available at: `https://github.com/mark-borg/sign-language-detection`.

Apart from just searching for individual videos, we also traverse the contents of YouTube channels, that were returned by the YouTube API as a response to our original search query. A YouTube channel is a collection of videos produced by the same author, and typically belonging to the same topic. Table 2 lists some of the YouTube channels with video content related to signing or sign language. The table only lists the channels with the largest content (number of videos), and this is just a fraction of the thousands of channels that are returned when performing searches with all the sign language keywords of Table 1. When traversing the contents of YouTube channels, we only consider the first 100 videos of each channel. This is done in order to protect the richness in variety in both sources and topics, and not have very prolific channels vastly outnumbering smaller channels.

A list of approximately 38,000 video URLs was compiled from the returned search query results. This list formed the candidate list of videos for our dataset. We then randomly selected videos from this candidate list, downloaded the videos themselves, and applied some basic

Table 1. Keywords used to search on YouTube for videos related to signing and speaking

Activity type	Keywords
Signing	sign language, signing, sign, signs, deaf language, deaf, handspeak, deaf culture, ASL, BSL, Auslan, LSF, OGS, DTS, NGT, SVK, DGS, LIS, GSL, NSL, LSE, LSM, ...
Speaking	speaking, speech, talking, talk, discussion, discussing, discourse, lecture, lecturing, lesson, gesticulation, gesticulating, hand gesturing, interview, interviewing, body language, speaker, argue, arguing, meeting, forum, ...
Other	hand, hand movements, hand exercise, playing piano, playing instrument, hand tricks, mime, miming, write, writing, calligraphy, martial arts, judo, karate, table tennis, typing, archery, yoga, hand signs, exercise, laughing, listening, clap, clapping, ...

Table 2. Some YouTube channels that are returned when using one of the sign language keywords as a search term

Channels	TV-DEAF, Sign1news, deafnewspaper, The Daily Moth, Handspeak, thedailysign, DEAF Inc., 57davison, Sign Duo, Bill Vicars, ASLU, Andy Signs, JameSigns, Ready Set Sign, fsdbvideos, ASL THAT, ConnectRockDeaf, KHS ASL, Signing for Christ, ShallWeSign, ConvoRelay, Deafaotearoa, Sarah Signs, WHS ASL Club, Preparedness4Deaf, Beauty of ASL, ASLeslie, Rochelle Barlow, ASL Anissa, Deaf Bible, West Bengal Sign Language, DeafJapan TV, Jon Lenois Savage, Mystic Signs, Sign The Bible, SVRS, ERCODvideos, ASL MacGyver, Elma Production, ASL Stew, Expression Australia, The Deaf Society, SigningProf, deafdirect, DawnSignPress, Seek the World, GallaudetU, ...

manual filtering at video level. Inappropriate videos, videos not matching their topic/keywords, blurred and other poor quality videos were filtered out. Videos that survived this filtering stage were then incorporated into the final dataset. The final count after filtering is of 1120 videos, with a total size of 53.4GB.

2.2. Video Frame Sampling and Groundtruthing

Videos were sampled at 5Hz and rescaled to a frame size of 224×224. Each video contributes up to 2000 frames at most (only the first 6.6 minutes of each video are used). This results in a total of 1,448,153 video frames (total disk size of 22.9GB, JPEG encoded).

Groundtruthing was applied at video frame level, taking into consideration the surrounding context (surrounding 10 video frames). 1,232,558 video frames were labelled as 'signing', 'speaking' or 'other'. The remaining video frames are either ambiguous cases or are as yet unlabelled. Table 3 shows the counts per class.

Table 3. Number of groundtruthed video frames per class

class	total frames	%
signing	535,105	43%
speaking	511,446	42%
other	186,007	15%
	1,232,558	100%

Figure 1 shows sample video frames from the video segments labelled as 'signing'. Note the variability in camera positions, viewing angle, pose, etc. Several videos contain 2 or more signers. In other videos, the signer(s) is/are not fixed but move around, or the camera moves around. Being untrimmed videos, there are also several scene changes in a typical video, with the signer(s) not guaranteed to be always present. Several videos contain a mixture of signing and speaking actions. Two such examples include videos with TV inset signing, and videos for teaching sign language. For the latter case, the same person alternates between signing and speaking.

Figure 2 shows sample videos from the video segments labelled as 'speaking'. Notice how in particular the samples of the first 5 rows of the figure show strong gesticulation and hand gestures (e.g. pointing) accompanying speech.

Figure 3 shows sample videos from the video segments labelled as 'other'. The first 4 rows show various hand movements (close-ups, upper body, full body) that can bear similarity to both signing and gesticulation. Also included are persons and people in various poses. The last rows are video segments that are neither signing nor speaking.

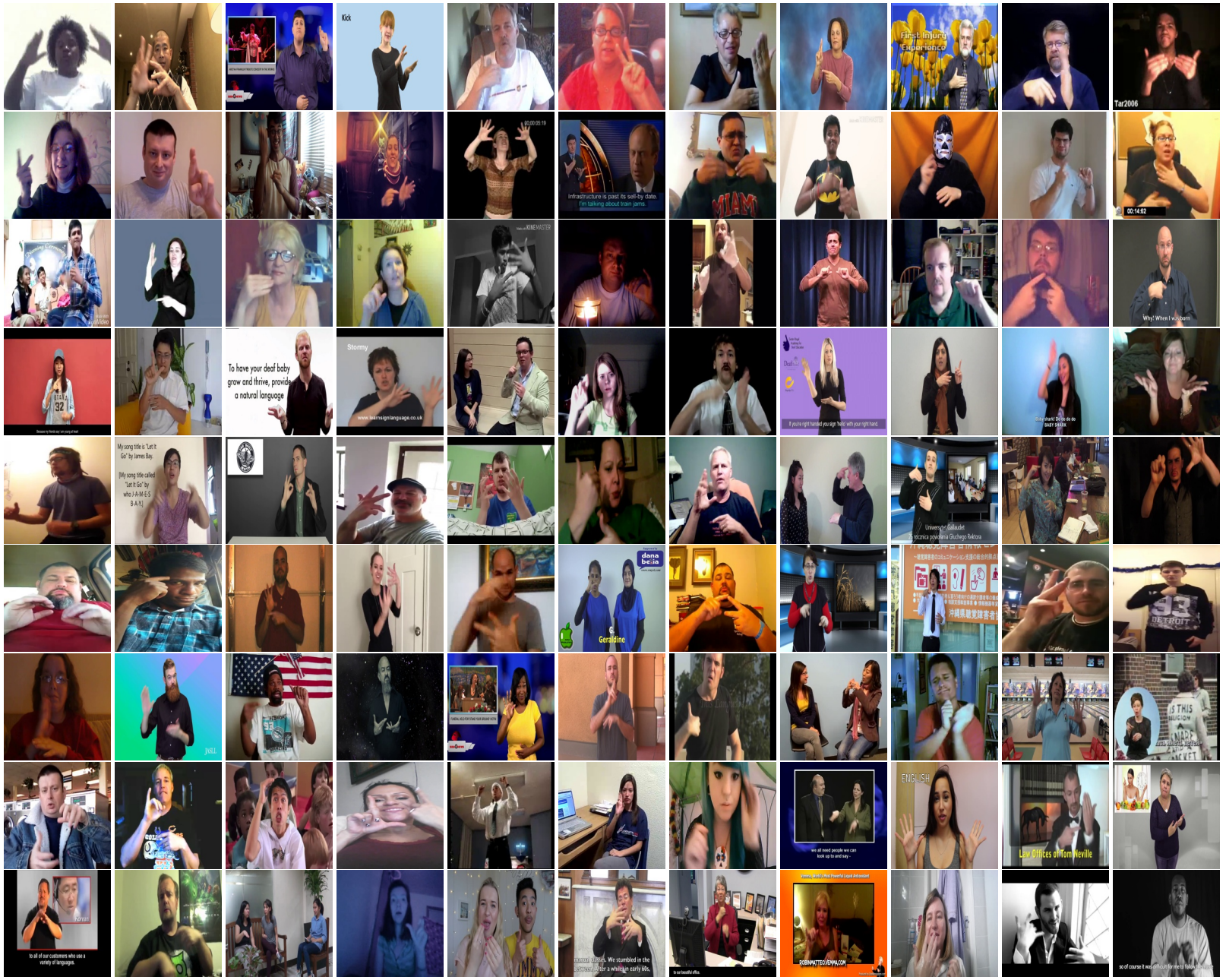


Fig. 1. Sample frames from our dataset labelled as 'signing'.

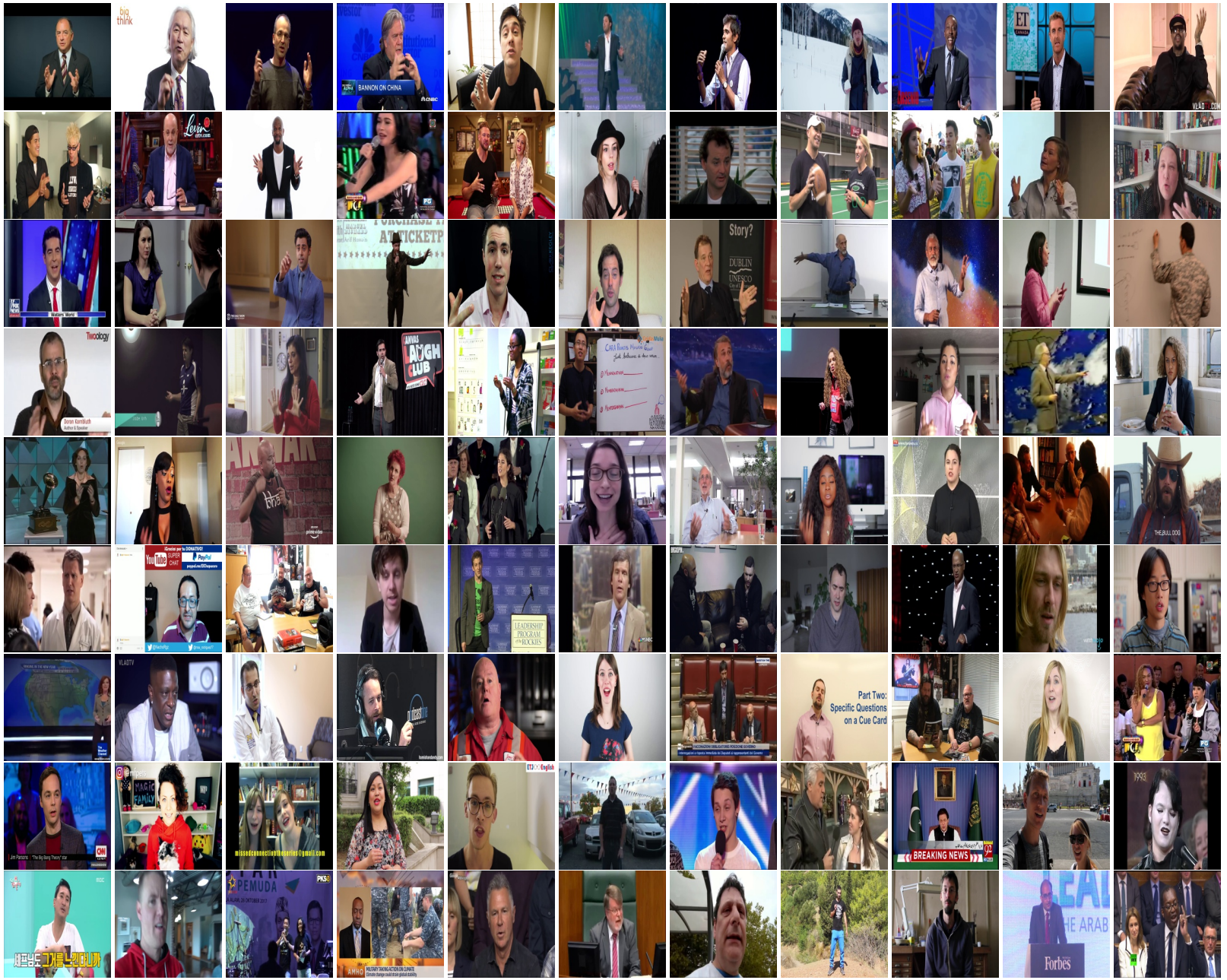


Fig. 2. Sample frames from our dataset labelled as ‘speech’.

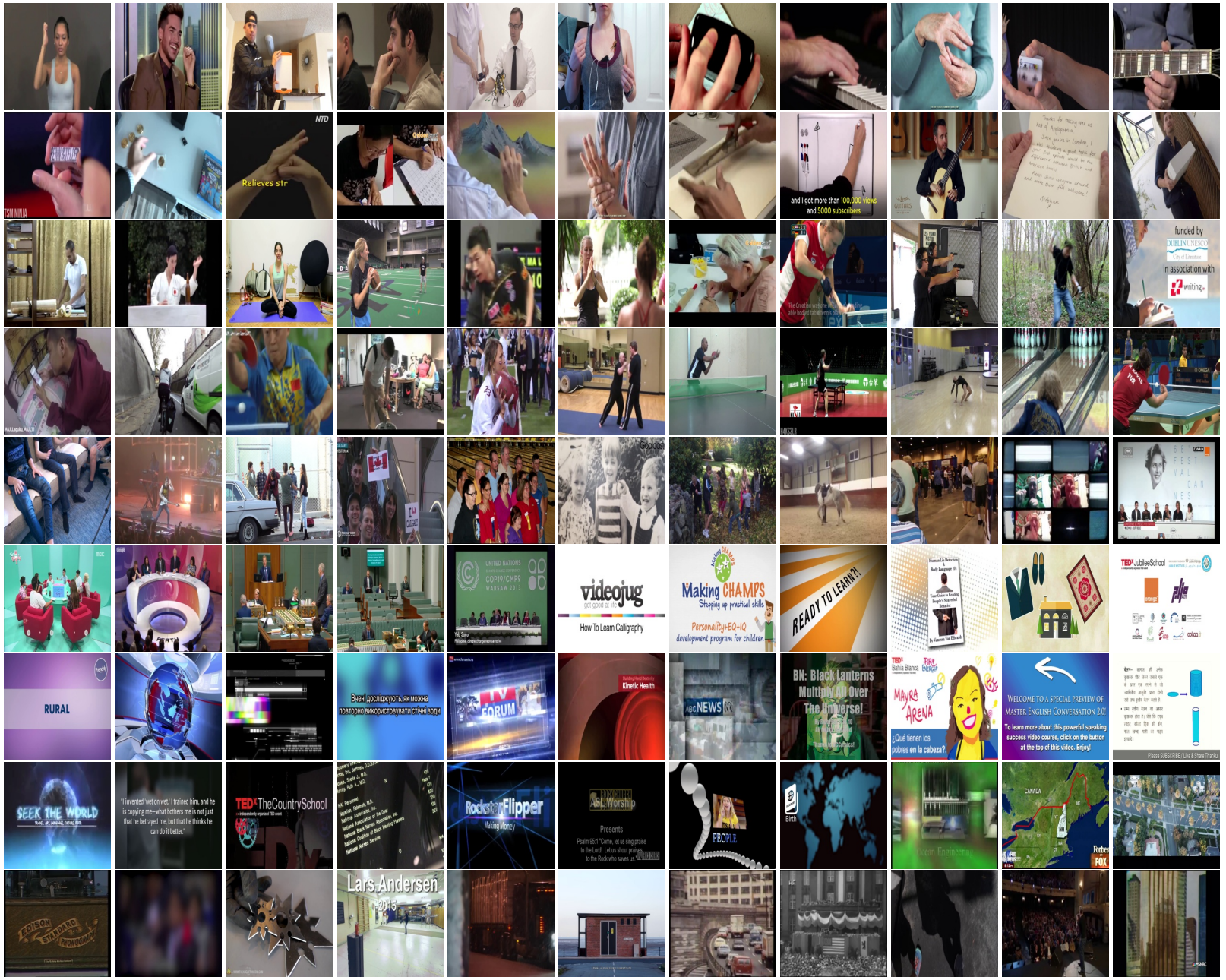


Fig. 3. Sample frames from our dataset labelled as 'other', i.e., non-signing and non-speaking video segments.

2.3. Bias and Side Information

The rationale behind the use of YouTube, and the way we select YouTube videos, is to get as wide a variety in signing as it is possible, so that the dataset can be truly that of "signing in the wild". But for all the good intentions, some bias will inevitably exist in the dataset. For example, ASL is the most prevalent language on YouTube, which naturally creates biases in terms of signing, signing styles, and more generally in terms of race, gender, age, etc. Thus the signing in our dataset might not reflect the "true" distribution of signing. The same might happen for the speaking category.

We also tried to make sure that a machine learning algorithm is not able to exploit any *side information* that might creep in to the dataset. Examples of side information might be: differences in camera viewpoints (signing tends to be captured with a frontal camera position, zoomed on the upper body of the signer), background elements and scenes being more common in videos of one class than for the other (speakers tend to use microphones), video durations (long videos tend to be lectures), and logos or video title screens specific to one class ('TedX' videos contain people speaking, the word 'deaf' in a title screen means signing will follow).

While it is difficult to ascertain that such side information is absent, we tried to minimise these occurrences by limiting the amount of videos coming from the same source (channel), selecting randomly from a much larger list of potential videos, and setting an upper limit on how many video frames are taken from a video.

2.4. Video IDs and Stratified 5-fold Partitioning

We apply *stratified 5-fold partitioning* to the video frames, making sure that all the frames of each single video appear in the same fold only. Because it is a stratified partitioning, the relative occurrences of the classes in each fold remain the same as for the whole dataset (the third column of Table 3)

YouTube assigns each video a unique identifier as its video ID. The video ID consists of 11 alphanumeric characters, with each character taken from the set {0-9, a-z, A-Z, _, -} [3]. Video IDs are assigned randomly by YouTube upon upload, with the random nature of the assignment and the vast combinatorial space of all possible identifiers ensuring no name collision. Prior to partitioning, for ordering videos, we use the natural sort order that arises from the video IDs assigned by YouTube. This is a valid random order in its own right, with the added advantage that it can aid in the reproducibility of our experiments.

3. CNN FEATURES

Our experiments show that CNN features extracted from the first fully-connected layer ('fc1') of the VGG-16 network [4] are better discriminators for our dataset than those obtained from the last convolutional layer ('block5_conv3'). Please refer to Figure 4 for an illustration of the VGG-16 model of Simonyan and Zisserman [4], on which we indicate the layers from where we extract the CNN features.

To examine this further, we perform a 2D embedding of the CNN features using t-distributed stochastic neighbour embedding (t-SNE) [5]. t-SNE is a dimensionality reduction technique that uses the Kullback-Leibler (KL-) divergence as a dissimilarity measure between the distribution of data in the original space and the distribution of the same data in a lower dimensional space (2D in our case). A Gaussian is used to model the conditional probability of the data in the high dimensional state, while a t-Student distribution is used for the conditional probabilities in the lower-dimensional space. A gradient descent method is used by t-SNE to minimise the sum of the KL-divergences over all the data points. Because of the KL-divergence's asymmetric properties, t-SNE is able to preserves local structures that arise in the original space. In our case, we hope that the local structures reflect the fact that the CNN features of signing, speaking, and other video segments belong to separate manifolds.

For computational reasons, we perform this experiment on a subset of the dataset, selecting 30,000 video segments split equally between the three classes. The CNN features extracted from the 'block5_conv3' layer have a size of $25,088$ ($7 \times 7 \times 512$ prior to flattening), while those of the 'fc1' layer have a size of 4096. We apply principal component analysis (PCA) prior to running t-SNE, keeping the principal 1000 components, accounting for $\approx 78\%$ of the variation in the feature data. PCA increases the efficiency with which an embedding is generated by t-SNE [5].

As recommended by van der Maaten [6], we run t-SNE five times for each set of CNN features and select the embedding with the lowest KL-divergence value. We use an implementation of t-SNE that employs the Barnes-Hut optimisation for gradient calculation. t-SNE is configured with perplexity set to 40, early exaggeration of 12.0, learning rate of 200, and maximum number of iterations limited to 1000.

Details of the t-SNE runs are listed in Table 4. And Figure 5 shows the resulting 2D embeddings of the 'block5_conv3' and 'fc1' features corresponding to the runs with lowest KL-divergence values (bold entries in Table 4). The figures show that there is a slightly better separation (less overlap) between the signing, speaking and other clusters of the 'fc1' features when compared to the 'block5_conv3' features. This is in agreement with the improvement in classification results in favour of the 'fc1' features. Another advantage of the 'fc1' CNN features over 'block5_conv3' is the smaller size of the feature vector (4096 versus 25,088), increasing the computational efficiency of our neural network model.

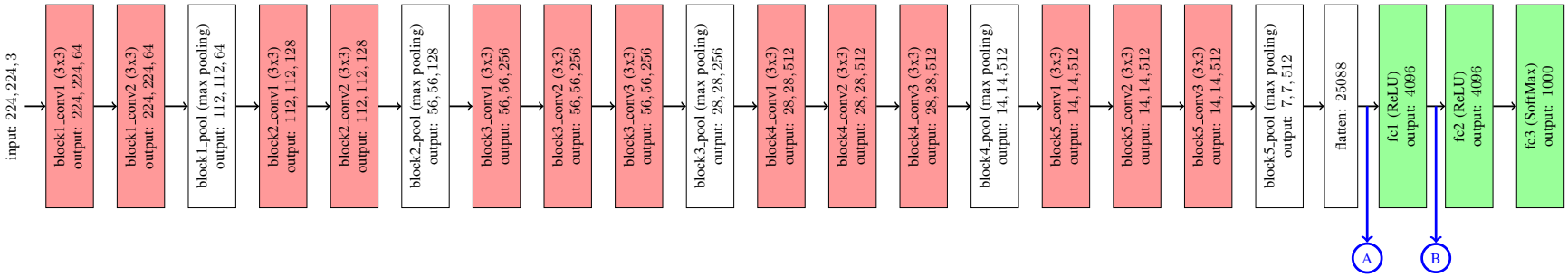
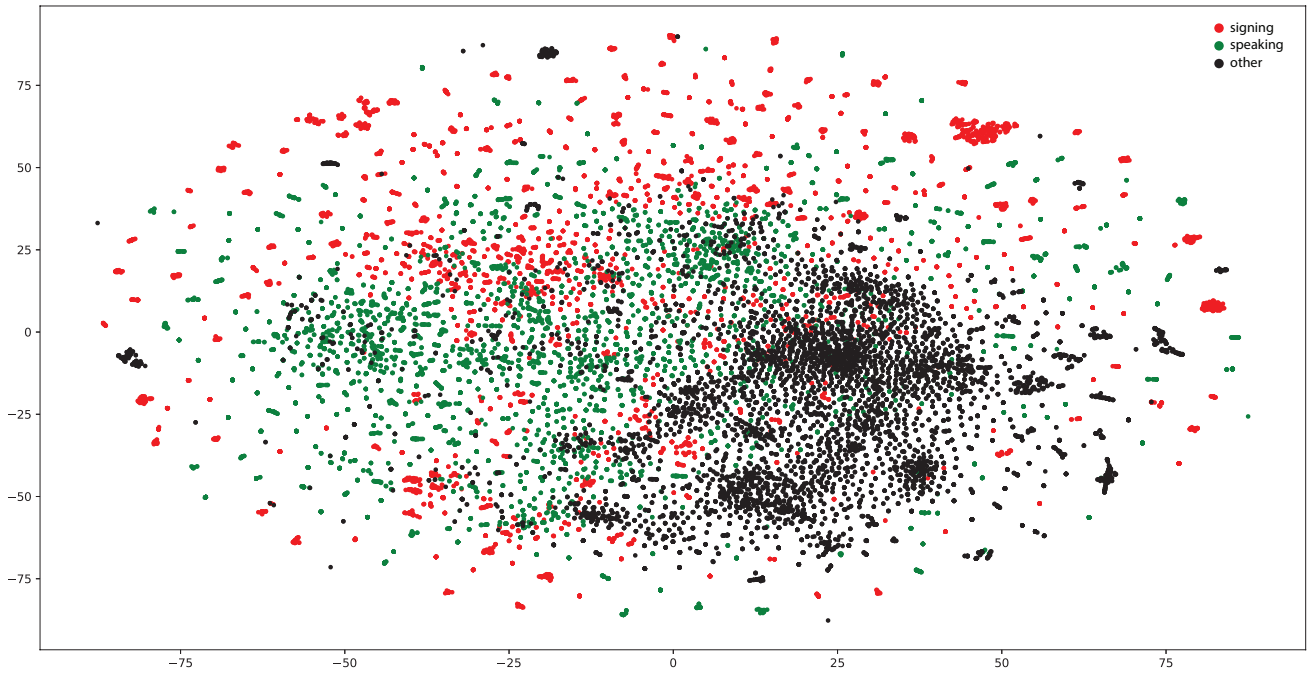
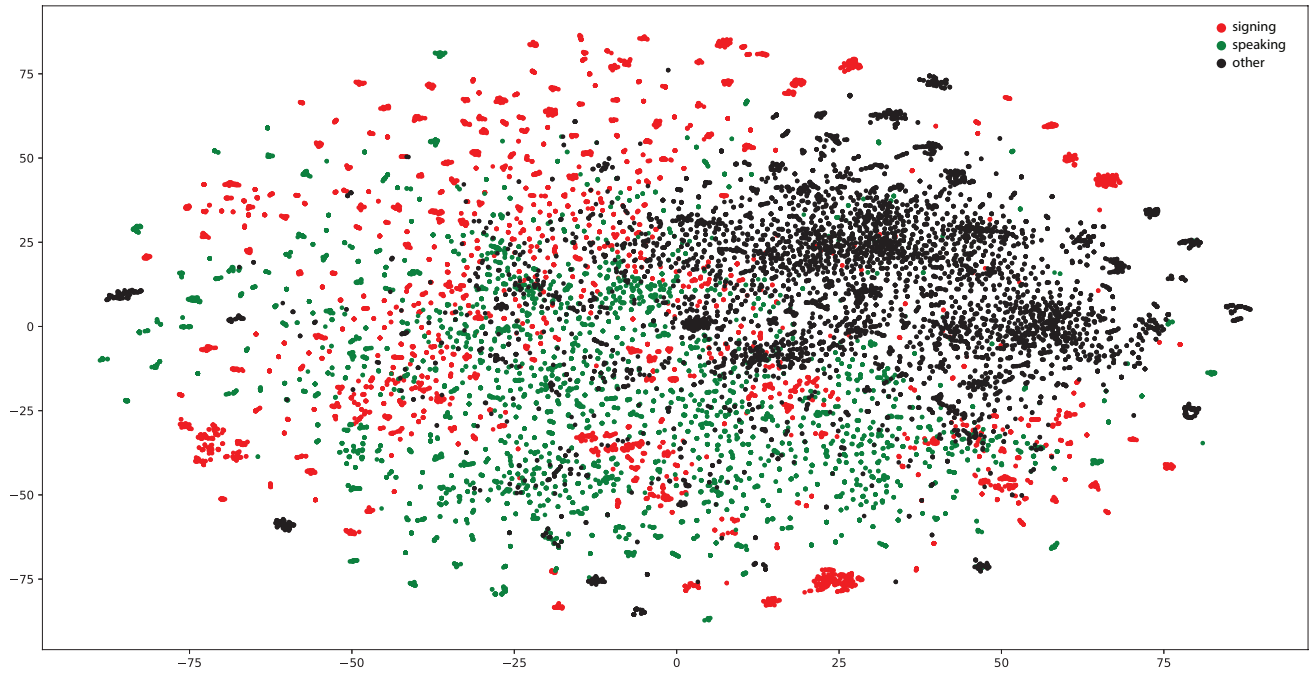


Fig. 4. The full VGG-16 network model [4] consists of 16 trainable layers. The convolutional base is made up of 5 blocks, each block containing 2 or 3 convolutional layers (denoted as red-filled boxes) of 3x3 filters and stride 1, followed by a max pooling layer (white-filled boxes). After the last convolutional layer, VGG-16 adds two fully-connected layers (green-filled boxes), followed by a final softmax classifier with 1000 nodes for the 1000 ImageNet classes. We extract CNN features from 2 different layers in the VGG-16 network. These 2 positions are denoted in blue as Ⓐ and Ⓑ in the figure above, and we refer to them as 'block5_conv3' and 'fc1' features respectively, after the trainable layer that is closest to them (max pooling is a spatial sub-sampling operation while flatten simply re-arranges the output into a vector).



(a) t-SNE embedding of CNN features extracted from layer 'block5_conv3'



(b) t-SNE embedding of CNN features extracted from layer 'fc1'

Fig. 5. 2D t-SNE embeddings of the CNN features extracted from (a) the last convolutional 'block5_conv3' layer and (b) the first fully-connected 'fc1' layer of the VGG-16 network when applied to a subset of our dataset (30k video segments). KL-divergence value for (a) is 1.7108, and for (b) is 1.5806.

Table 4. Results of t-SNE embeddings of CNN features

VGG-16 layer from which CNN features are extracted		**block5_conv3**	**fc1**
number of video segments		30000	30000
length of CNN feature vector		25088	4096
length of CNN feature vector after running PCA		1000	1000
KL-divergence values at end of run	run 1	1.711726	**1.580592**
	run 2	1.713404	1.587284
	run 3	**1.710764**	1.592318
	run 4	1.713658	1.581992
	run 5	1.712586	1.581291

4. RNN-BASED MODEL FOR SIGN LANGUAGE DETECTION

This section contains some additional diagrams of our proposed system that complement the ones in the main paper. We first reproduce in Figure 6 the diagram given in the main paper of the proposed system. And in Figure 7 we reproduce our RNN model made up of 2 stacked layers of gated recurrent units (GRUs). Then in Figure 8, we combine the two together into a single diagram.

For experiments where we considered only one modality (one stream), e.g. RGB data only, or a specific motion data only, then the network of Figure 8 is reduced to either the left hand side only (for RGB data) or the right hand side only (for motion data). We also do away with the 'class score fusion' node shown at the top of Figure 8.

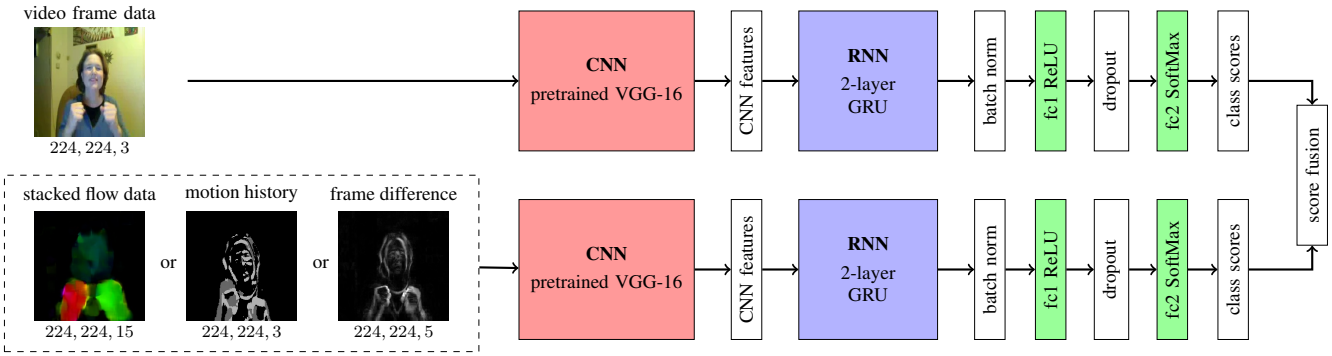


Fig. 6. Our proposed sign language detection framework using features extracted via CNNs from raw video and motion data in parallel. We also employ an RNN to leverage temporal information within video segments, followed by a non-linear classifier.

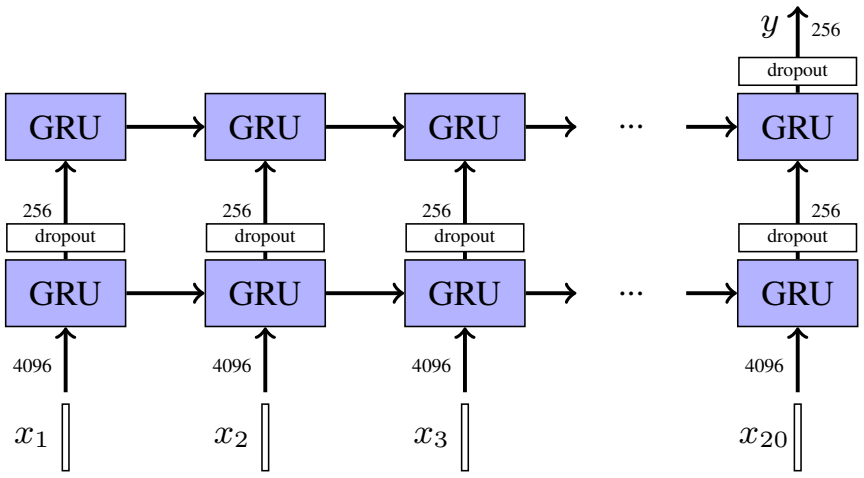


Fig. 7. A 2-layer RNN network, with 256 hidden units and 20 timesteps. x_i are the input CNN features, and y is the RNN output.

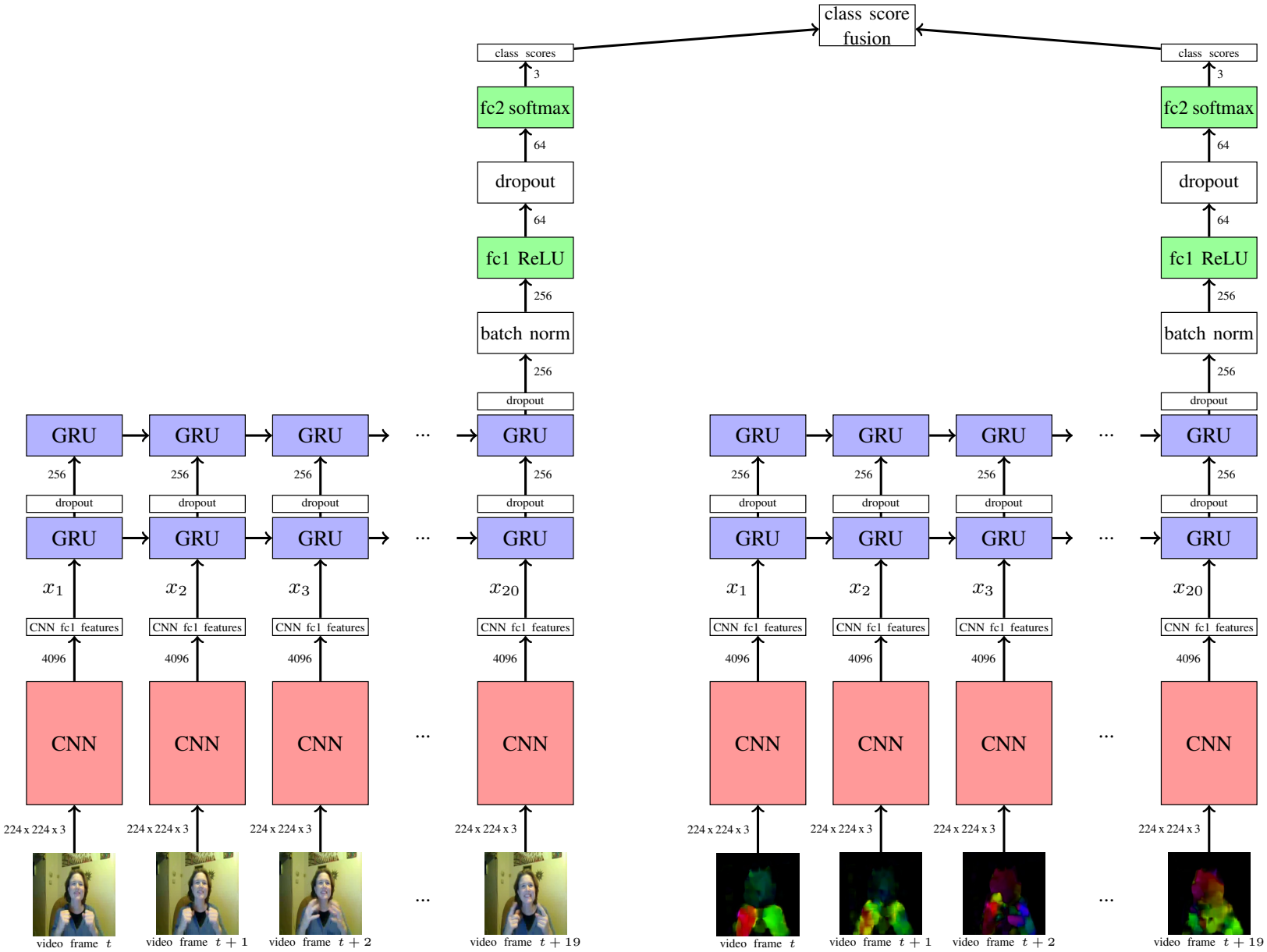


Fig. 8. Our proposed system incorporating a 2-stream CNN network, one stream processing RGB data, the other optical flow data (or MHI or multi-frame differencing). A block of 20 timesteps from each CNN are fed into a 2-layer RNN, which in turn is fed into a non-linear classifier. Finally, class score fusion combines the predictions from both classifiers.

Note that the RNN takes as input the CNN features of 20 timesteps $\{x_1, x_2, ..., x_{20}\}$. Each timestep x_i is processed independently through the CNN, but then the CNN features of the 20 timesteps constitute the sequence data upon which the RNN operates. Since we sampled videos at 5Hz, this sequence equates to a 4-second video segment. In typical fluent signing, the length of an individual sign (equivalent to a word in spoken language) varies from 0.5 to 1 second [7]. Therefore the RNN should be "seeing" the temporal dynamics of between 2 to 4 signs within its input data.

5. DETECTION RESULTS

To include several video examples here: video frames + groundtruth + detection probabilities...

6. REFERENCES

[1] YouTube, "YouTube Data API," `https://developers.google.com/youtube/v3/`, 2016, (version 3).

[2] Wikipedia, "List of sign languages," `https://en.wikipedia.org/wiki/List_of_sign_languages`, 2018.

[3] Haroon Malik and Zifeng Tian, "A Framework for Collecting YouTube Meta-Data," *Procedia Computer Science*, vol. 113, pp. 194 – 201, 2017, The 8th International Conference on Emerging Ubiquitous Systems and Pervasive Networks (EUSPN 2017) / The 7th International Conference on Current and Future Trends of Information and Communication Technologies in Healthcare (ICTH-2017) / Affiliated Workshops.

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

[5] Laurens van der Maaten and Geoffrey Hinton, "Visualizing High-Dimensional Data using t-SNE," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.

[6] Laurens van der Maaten, "t-SNE," https://lvdmaaten.github.io/tsne/, 2018, Accessed: 2018-10-18.

[7] Rosalee Wolfe, John C. McDonald, Larwan Berke, and Marie Stumbo, "Expanding n-gram analytics in elan and a case study for sign synthesis," in *LREC*, 2014.