Supplementary Materials: Weakly-supervised Fingerspelling Recognition in British Sign Language Videos

K R Prajwal*[1]
http://robots.ox.ac.uk/~prajwal
Hannah Bull*[2]
https://hannahbull.github.io
Liliane Momeni*[1]
https://www.robots.ox.ac.uk/~liliane
Samuel Albanie[3]
https://samuelalbanie.com
Gül Varol[4]
http://imagine.enpc.fr/~varolg
Andrew Zisserman[1]
http://robots.ox.ac.uk/~az

[1] Visual Geometry Group
Department of Engineering Science
University of Oxford
Oxford, UK

[2] LISN, Univ Paris-Saclay,
CNRS, France

[3] Department of Engineering, University of Cambridge, UK

[4] LIGM, École des Ponts, Univ Gustave Eiffel, CNRS, France

We provide further details on the computation of feature similarity to obtain fingerspelling localisations in Sec. B. Details on our modified text edit distance are in Sec. C. Implementation details on the Transpotter model are provided in Sec. D and an experiment demonstrating our choice of features is recorded in Sec. E. Finally, we show qualitative examples of our results on the test set as well as our pseudolabels on the training set in Sec. F.

A Supplementary video

We highly recommend checking out the supplementary video at:
`https://www.robots.ox.ac.uk/~vgg/research/transpeller/`,
which contains several video results and explanations on how crucial parts of our paper, such as pseudolabeling, impact the final results.

B Feature similarity

As mentioned in Sec. 4.1, we use feature similarity to search for fingerspelling instances, given a small number of manual annotations (115 examples of frames containing fingerspelling). We compute the cosine similarity between these exemplars E and features of the videos in our corpus.

Formally, given a video V with N frames and d dimensional features corresponding to each frame, we can express V as a $N \times d$ matrix, which we call $\mathcal{V}$. Similarly, we can express the E exemplar frames as a $E \times d$ matrix $\mathcal{W}$. The cosine similarity between rows of $\mathcal{V}$ and rows of $\mathcal{W}$ is a measure of similarity between frames of V and fingerspelling. We can thus normalise $\mathcal{V}$ and $\mathcal{W}$ along the feature dimension, denoted $\mathcal{V}_*$ and $\mathcal{W}_*$, and compute a similarity matrix $\mathcal{S} = \mathcal{V}_* \times \mathcal{W}_*^T$ of dimension

© 2022. The copyright of this document resides with its authors.
It may be distributed unchanged freely in print or electronic forms.

* Equal Contribution.

$N \times E$. We consider the percentage of columns c of $\mathcal{S}$ with a similarity value about a threshold S. If c is greater than a threshold C, then we consider the frame to be fingerspelling. In our case, we let $S = 0.4$ and $C = 0.2$. We apply some smoothing to the edges of the fingerspelling detections and remove very short segments of either fingerspelling or gaps between fingerspelling localisations.

C Text proximity score

In order to pseudo-label the training dataset using the outputs of the Transpeller model from Stage 1, we find potential fingerspelled words in the subtitles with a high proximity score to the decoded outputs, as described in Sec. 4.2. This proximity score is a modified version of the Levenshtein edit distance, as the classic Levenshtein edit distance between two strings is not necessarily well adapted to our task of fingerspelling recognition. Although the first letter of a word is almost always fingerspelled, other letters may be omitted, e.g. 'Scapa Flow' may be fingerspelled 'SCAPAFW' and 'HARRY' may be fingerspelled 'HRY'. Letting w_1 be the subtitle word and w_2 be the Transpeller outputs, we compute our adapted edit distance dist(w_1, w_2) as follows:

1. All repeated letters in w_1, w_2 are removed before computation, e.g. (HARRY becomes HARY).
2. Letters in w_2 that are not in w_1 are penalised and removed, e.g. (HARY, HERY) becomes (HARY, HRY) with a malus of +1.
3. Correct prediction of the first letter reduces the edit distance by 1, e.g. (HARY, HRY) has a bonus of -1, but (HARY, ARY) does not.
4. Insertions and subtitions each carry a malus of +1, e.g. (HARY, HYR) has a malus of +1.
5. Deletions are not penalised.
6. The proportion of letters of w_1 not in w_2 is added as a fractional malus, e.g. (HARY, HRY) has a malus of +1/4. This is to ensure that dist(HARY, HRY)<dist(HUMPHREY, HRY), given that deletions are not penalised.

D Implementation details

Data sampling. We form the training samples $\mathcal{D}$ by randomly (50% chance) sampling positive (with fingerspelling) or negative (without fingerspelling) clips. Note that we have a lot more detections of fingerspelling (see Tab. 1), but only a fraction of them are associated with a word label. The duration of the sampled clips ranges from 1.2s to 4.8s. For training the Stage 1 model, we found that a simple augmentation strategy such as randomly dropping characters (except the first character) helps combat overfitting in the low data regime. This was, however, not helpful for our Stage 2 model training, when we obtain a larger training set.

Data splits. All our models are trained on the automatic annotations for the videos in the *train* split of the BOBSL dataset. Similarly, for validation, we use the automatic annotations of the videos in the *val* split. We choose the checkpoint with the lowest validation loss and evaluate it on the manually verified test benchmark

Automatic annotations. We use Flair [1] parts of speech tagging to identify nouns and proper nouns in the subtitle text. We take mouthing annotations with confidence scores of above 0.1, using the model in [3].

Evaluation. At evaluation, we give the model as input the context window seen by the annotators, i.e. 2.1s before and 4s after the midpoint of the automatically detected fingerspelling instance.

Lookup-based correction at inference-time. We first curate a list of nouns present in the subtitles of the BOBSL train set and use edit distance to match the model's predictions to the closest noun in our list. If the edit distance is below a set threshold, i.e., a very close match, we replace the predicted character sequence with the matched word from our list.

E Choice of features

To extract the visual features, we use a Video-Swin-S model pretrained for the sign classification task. Previous works [2, 3, 4] used an I3D model in a similar fashion. In Tab. A.1, we compare these two feature extractors and show that our Video-Swin-S features are far superior to the I3D features.

Annotations	**# Recognition ex.**	**CER_{fspell}**	**CER_{full}**
I3D (Exemplars + Mouthings: proper nouns)	39K	64.6	66.5
Swin (Exemplars + Mouthings: proper nouns)	39K	**58.5**	**62.1**

Table A.1: **I3D vs. Swin features.** Using the Video-Swin-S model rather than an I3D model trained for sign classification provides markedly better features. We thus train the Transpotter model on Swin features in all of our experiments.

F Qualitative examples

Fig. A.1 shows examples of success and failure cases of the Transpotter model on our test set. Fig. A.2 demonstrates how letter labels from mouthings in Stage 1 are improved by Transpeller pseudolabels in Stage 2 for better supervision.

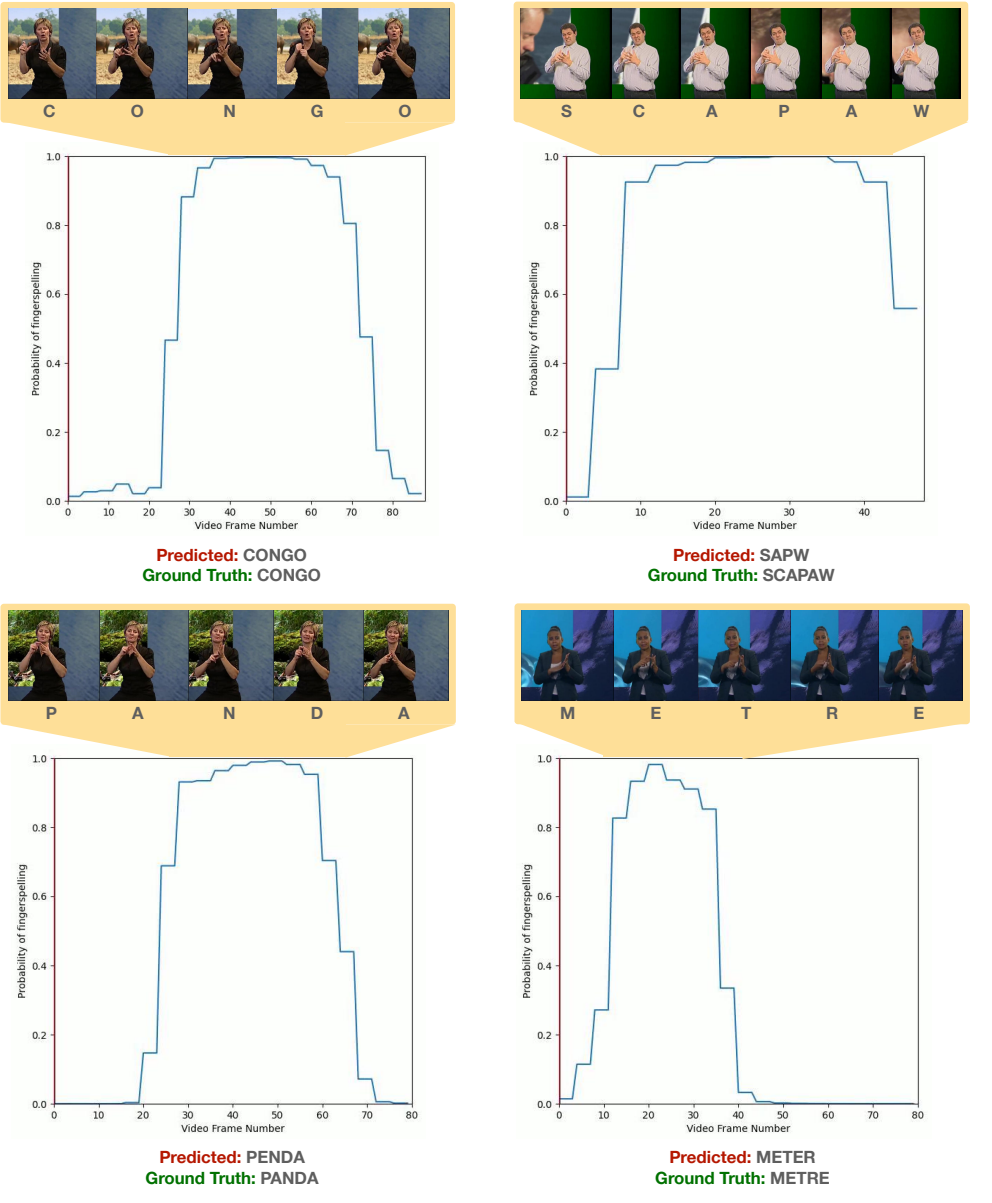

Figure A.1: **Success and failure cases on the test set.** In the top left example, our model correctly predicts the fingerspelling letter sequence. The top right example shows how our model fails to recognise some letters when the fingerspelling is fast or when the hand movements are blurry. In the bottom left example, our model confuses the letter 'E' and 'A', which are very similar in BSL. Finally, the bottom right example shows a case where the model switches the order of two letters. Alternatively, perhaps our model is too reliant on a language model it has learned from the training examples, and predicts an alternative spelling of 'METRE'.

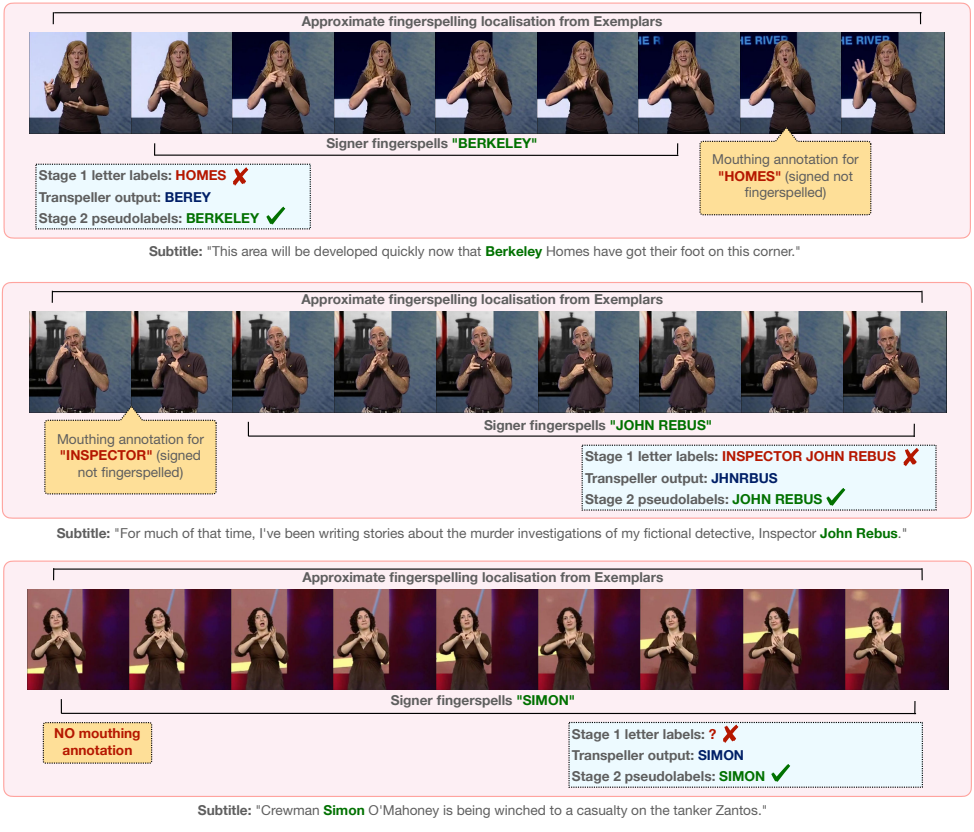

Figure A.2: **Improvements to our pseudolabels in Stage 2.** During Stage 1, we use mouthing cues to annotate fingerspelling. These mouthing cues may not correspond to the word which is being fingerspelled, but rather a sign either directly before or directly after the fingerspelling. The boundaries of our fingerspelling localisations are not necessarily accurate enough to exclude these mouthings. In other cases, the mouthing model fails to accurately spot a mouthing cue in a fingerspelling segment, and so we do not have an annotation. Using the Transpeller outputs in combination with the subtitle text, we are able to improve our training annotations for Stage 2.

References

[1] Alan Akbik, Duncan Blythe, and Roland Vollgraf. Contextual string embeddings for sequence labeling. In *COLING 2018, 27th International Conference on Computational Linguistics*, pages 1638–1649, 2018.

[2] Hannah Bull, Triantafyllos Afouras, Gül Varol, Samuel Albanie, Liliane Momeni, and Andrew Zisserman. Aligning subtitles in sign language videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11552–11561, 2021.

[3] Liliane Momeni, Hannah Bull, K R Prajwal, Samuel Albanie, Gül Varol, and Andrew Zisserman. Automatic dense annotation of large-vocabulary sign language videos. In *ECCV*, 2022.

[4] Gul Varol, Liliane Momeni, Samuel Albanie, Triantafyllos Afouras, and Andrew Zisserman. Read and attend: Temporal localisation in sign language videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16857–16866, 2021.