Supplementary Material: Fingerspelling Detection in American Sign Language

Bowen Shi[1], Diane Brentari[2], Greg Shakhnarovich[1], Karen Livescu[1]
[1]Toyota Technological Institute at Chicago, USA [2]University of Chicago, USA
{bshi,greg,klivescu}@ttic.edu dbrentari@uchicago.edu

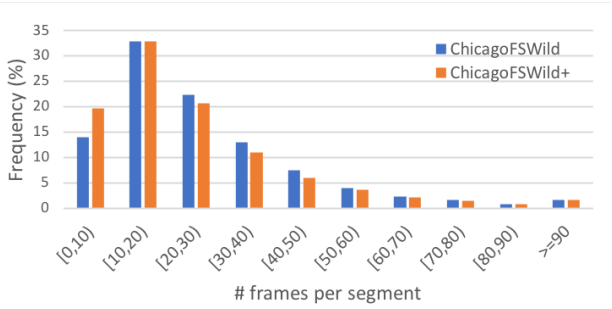
1. Statistics of the datasets

Table 1: Numbers of 300-frame raw ASL clips in the ChicagoFSWild and ChicagoFSWild+ data subsets. The number of fingerspelling segments in each subset is given in parentheses.

	train	dev	test
ChicagoFSWild	3539 (6927)	691 (1246)	613 (1102)
ChicagoFSWild+	13011 (44861)	867 (2790)	885 (1531)

Table 1 provides the numbers of clips and of fingerspelling segments in the datasets used in our work. Note that the number of fingerspelling segments is not exactly same as in [7, 8] due to the 75-frame overlap when we split raw video into 300-frame clips. On average there are 1.9/1.8 fingerspelling segments per clip for ChicagoFSWild/ChicagoFSWild+. The distributions of durations are shown in Figure 1.

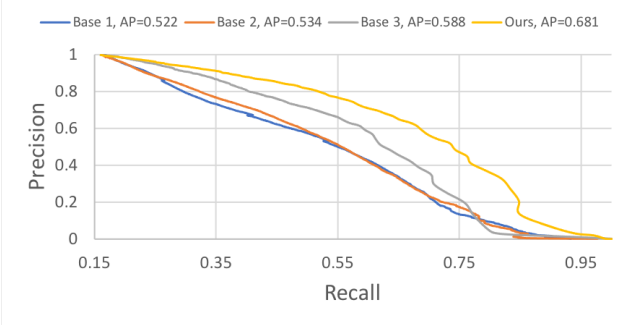
Figure 1: Distribution of length of fingerspelling segments.



2. Precision-recall curves for frame classification

Figure 2 shows the precision-recall curves for frame classification of the three baseline models and our proposed model. On the one hand, our approach dominates the others in terms of these frame-level metrics as well. In addition, the differences in frame-level performance among the three baselines are much smaller than the differences in sequence-level performance reported in the main text.

Figure 2: Precision-recall curves for frame classification of three baselines and our approach.



3. Comparison with state-of-the-art models for related tasks

In addition to the baseline models, we compare against two additional approaches—boundary matching network (BMN) [5] and multi-stage temporal convolutional network (MS-TCN) [3]—on our task of fingerspelling detection. Those two methods are state-of-the-art on temporal action proposal generation in ActivityNet1.3 [4] and sign language segmentation [6]. The implementations are based on [1, 2]. For fair comparison, we use the same backbone network as in the other methods. We use the same network architecture for the individual submodules of the two models and tune hyperparameters on our datasets. As MS-TCN does frame classification in principle, we follow the same post-processing steps as in baseline 1 and 2 to convert frame probabilities into sequence predictions for evaluation.

As is shown in Table 2, these two approaches do not outperform our approach. Comparing BMN and our baseline 3, we notice that the size of the training set has a large impact. The more complex modeling choices in BMN, which searches over a wider range of proposals, leads to better performance mostly when using the larger training set of ChicagoFSWild+. The discrepancy in performance

of these two models as measured by different metrics (e.g., AP@IoU vs. AP@Acc) also shows that a model with lower localization error does not always enable more accurate downstream recognition. The MS-TCN model is generally better than other frame-based approaches (baseline 1, 2) but remains inferior to region-based approaches including baseline 3 and ours. Our post-processing steps lead to inconsistency between the training objective and evaluation. Similarly in [6], it is noted that the model sometimes oversegments fingerspelled words.

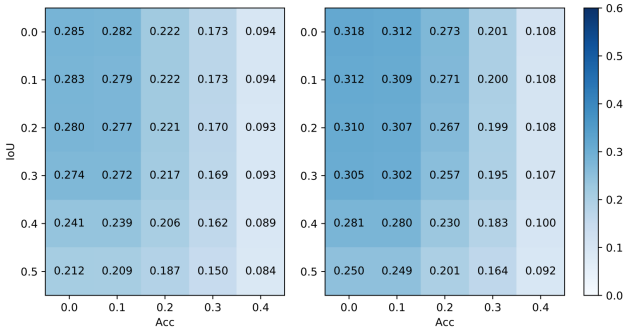
Table 2: Performance of BMN and MS-TCN on fingerspelling detection using our evaluation metrics on the (a) ChicagoFSWild and (b) ChicagoFSWild+ test sets.

		AP@IoU			AP@Acc			MSA
		AP@0.1	AP@0.3	AP@0.5	AP@0.0	AP@0.2	AP@0.4	
(a)	BMN	.442	.394	.284	.209	.157	.070	.307
	MS-TCN	.282	.177	.095	.141	.093	.036	.319
(b)	BMN	.580	.549	.437	.433	.401	.260	.470
	MS-TCN	.429	.345	.179	.350	.299	.147	.414

4. Analysis of AP@Acc

Figure 3 shows how varying δ_{IoU} and δ_{acc} impacts the value of AP@Acc. The accuracy threshold δ_{acc} has a much larger impact on AP than does δ_{IoU}. This is primarily because a large overlap between predicted and ground-truth segments is often necessary in order to achieve high accuracy. Therefore, we set the default value of δ_{IoU} to 0.

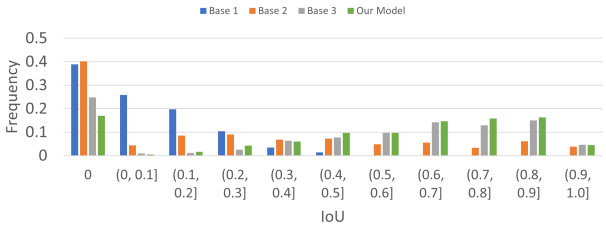
Figure 3: AP@Acc with different IoU thresholds on ChicagoFSWild dev set. Left: baseline 3. Right: our model.



5. Histogram of IoU

Figure 4 shows histograms of IoU of predicted segments with respect to the ground truth at peak thresholds used in the MSA computation. Our model has overall higher IoU than the three baselines. The average IoUs of the three baselines and our model for the optimal (peak) threshold δ_f are 0.096, 0.270, 0.485, and 0.524 respectively. The average IoUs of baseline 3 and our model suggest that for AP@IoU, AP@0.5 is more meaningful to compare in terms of recognition performance for those two models.

Figure 4: Histogram of IoU at peak thresholds.



6. Performance breakdown over durations

We separate raw video clips into three categories based on the duration of the fingerspellng segments: short (<20 frames), medium (20-80 frames), and long (≥80 frames). This division is based on the statistics of the dataset. The performance of our model for the three categories is shown in Table 3, and can be compared to the overall performance in Table 1 of the paper. Shorter fingerspelling segments are harder to spot within regular signing. The typical fingerspelling pattern (relatively static arm and fast finger motion) is less obvious in short segments. In addition, Figure 5 shows the length distribution of false positive and false negative detections from our model. The length distribution of false positives roughly matches that of ground-truth segments in the dataset.

Table 3: Performance on segments of different durations.

	AP@IoU			AP@Acc			MSA
	AP@0.1	AP@0.3	AP@0.5	AP@0.0	AP@0.2	AP@0.4	
Short	.411	.346	.235	.149	.140	.051	.357
Medium	.675	.671	.623	.476	.361	.156	.435
Long	.781	.703	.420	.704	.362	.130	.393

7. Speed test

The inference time per video clip is shown in Table 4. The speed test is conducted on one Titan X GPU. Inference times for all models are under 1 second. Baselines 1 and 2 are faster as the model architecture is simpler. Our model takes roughly twice the time of baseline 3, which is mainly due to the second-stage refinement.

8. Detection examples

Figure 6 shows various detection examples from the ChicagoFSWild dev set.

Figure 5: Distribution of lengths of false positives and false negatives.

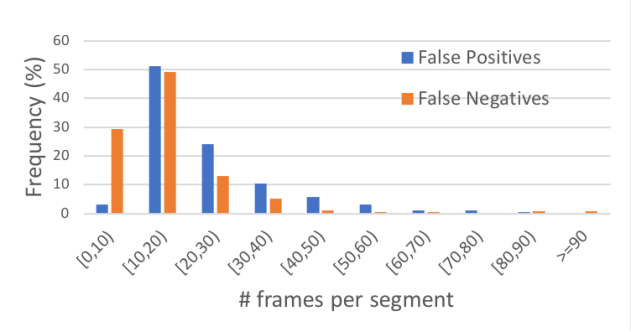


Table 4: Inference time per 300-frame video clip

	Base 1	Base 2	Base 3	Ours
Inference time (ms)	10.9	11.6	284.5	511.1

References

[1] https://github.com/JJBOY/BMN-Boundary-Matching-Network. 1

[2] https://github.com/yabufarha/ms-tcn. 1

[3] Yazan Abu Farha and Juergen Gall. MS-TCN: Multi-stage temporal convolutional network for action segmentation. In *CVPR*, 2019. 1

[4] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *CVPR*, 2015. 1

[5] Tianwei Lin, Xiao Liu, Xin Li, Errui Ding, and Shilei Wen. BMN: Boundary-matching network for temporal action proposal generation. In *ICCV*, 2019. 1

[6] Katrin Renz, Nicolaj Stache, Samuel Albanie, and Gül Varol. Sign language segmentation with temporal convolutional networks. In *ICASSP*, 2021. 1, 2

[7] Bowen Shi, Aurora Martinez Del Rio, Jonathan Keane, Diane Brentari, Greg Shakhnarovich, and Karen Livescu. Fingerspelling recognition in the wild with iterative visual attention. In *ICCV*, 2019. 1

[8] Bowen Shi, Aurora Martinez Del Rio, Jonathan Keane, Jonathan Michaux, Diane Brentari, Greg Shakhnarovich, and Karen Livescu. American Sign Language fingerspelling recognition in the wild. In *SLT*, 2018. 1

Figure 6: Detection examples. Red: ground-truth segment, green: predicted segment. The sequences are downsampled.