Fingerspelling recognition in the wild with iterative visual attention

Anonymous CVPR submission

Paper ID ****

Abstract

Sign language recognition is a challenging gesture sequence recognition problem, characterized by quick and highly coarticulated motion. In this paper we focus on recognition of fingerspelling sequences in American Sign Language (ASL) videos collected in the wild, mainly from YouTube and Deaf social media. Most previous work on sign language recognition has focused on controlled settings where the data is recorded in a studio environment and the number of signers is limited. A common recognition pipeline consists of hand detection, sometimes hand segmentation, followed by recognition of handshape sequences. This work aims to address the challenges of real-life data, while avoiding the need for supervised detection or segmentation modules. We propose an end-to-end model based on a neural attention mechanism, without hand detection or segmentation. We develop a new approach for obtaining high-resolution regions of interest, which outperforms prior work by a large margin. In addition, we show that performance can be improved by collecting crowd-sourced annotations of fingerspelling videos.

1. Introduction

did a bunch of editing –KL Automatic recognition of sign language has the potential to overcome communication barriers for deaf and hearing-impaired people. With the increased use of online media, sign language video-based websites (e.g., deafvideo.tv) are increasingly used as a platform for communication and media creation. Sign language recognition could also enable web services like content search and retrieval in such media.

From a computer vision perspective, sign language recognition is a complex gesture recognition problem, involving quick and fine-grained motion, especially in realistic visual conditions. It is also relatively understudied, with little existing data in natural day-to-day conditions.

In this paper, we study the problem of American Sign Language (ASL) fingerspelling recognition from naturally occurring sign language videos collected from web sites. Fingerspelling is a component of ASL where each letter has a canonical sign and words are signed out letter by letter (see Figure 1). Words are fingerspelled when they do not have their own ASL signs, for example technical items or proper nouns. Overall fingerspelling accounts for 12 to 35% [21] of ASL and is used frequently for content words in technical conversations and conversations involving current events. Specifically in deaf online media, fingerspelling recognition is crucial as it often contains high a proportion of such content words.

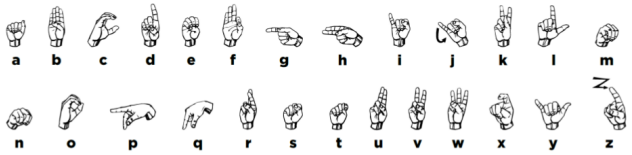

Figure 1. The ASL fingerspelling alphabet, reproduced from [11]

please check next par for repeated language from prev papers –KL Compared to sign language recognition in general, fingerspelling recognition involves a limited set of handshapes and is produced with a single hand (in ASL). On the other hand, fingerspelling recognition presents its own challenges. It involves very quick, small motions that can be highly coarticulated. In lower-quality video, motion blur can be very significant during fingerspelled portions. Furthermore, there exists high ambiguity among fingerspelled handshapes, especially for data "in the wild" (see Figure 2).

add some citations to this par (can grab a few representative ones from the related work section) –KL Automatic sign language recognition is commonly addressed with approaches borrowed from computer vision and speech recognition. The "front end" of the pipeline often consists of hand detection and sometimes segmentation, as well as visual feature extraction. Extracted features are then passed through a sequence model, similar to ones used in speech recognition.

Most prior work on sign language recognition has fo-

Figure 2. Illustrations of ambiguity in fingerspelled handshapes. Upper row: different letters with similar handshapes, all produced by the same signer. Lower row: the same letter (u) signed by different signers. replace images in this fig, maybe add "m". also try to get this fig onto the 1st page –KL

cused on data collected in a controlled environment. Figure 3 shows example images of fingerspelling data collected "in the wild" in comparison to a studio environment. Compared to studio data, naturally occurring fingerspelling images often involve more complex visual context and more motion blur, especially in the signing hand regions. Thus hand detection, an essential pre-processing step in the recognition pipeline, becomes more challenging.

We propose an approach for fingerspelling recognition that does not rely on hand detection. We make several contributions: (1) We propose an attention-based fingerspelling recognition model that can be trained end-to-end from raw image frames qualify this by mentioning face detector for scaling? –KL (2) We propose a new approach, *iterative attention*, for obtaining regions of interest of high resolution with limited computation. Our model trained with iterative attention achieves higher accuracy than the previous best approach [24], which required a custom hand detector. (3) We address the lack of data by collecting a data set of crowdsourced fingerspelling video annotations, in contrast with previous work using smaller carefully curated data sets, and showing that performance can be improved using this data set. We make our data set publicly available.[1]

2. Related Work

include related work on action recognition, gesture recognition, hand tracking? probably also need a few more sign language recognition citations for completeness (copy from SLT paper) –KL In general, conversion between signed and spoken languages is a *translation* problem, since sign languages each have their own lexica and syntax that are not necessarily aligned to those of any spoken language, and some recent work has begun to look at this problem [19]. However, the majority of prior work on sign language involves more constrained tasks, such as recognition of individual signs or fingerspelling recognition. In this paper we consider a constrained task (fingerspelling recognition), but loosen the visual and stylistic constraints in most previous work.

[1]Link to be provided upon paper acceptance.

Figure 3. Fingerspelling images in the wild vs. in studio data. Top: four example fingerspelling frames from our data set (see Section 4. Bottom: An example fingerspelling frame from studio data []. the bottom image is disproportionately big. could make this a 2-column fig with all 5 frames in a row. –KL

Early work on sign language recognition from video mainly focused on isolated signs [5, 3].[2] More recent work has focused on continuous sign language recognition and data sets [10, 7, 12]. Specifically for fingerspelling, the ChicagoFSVid data set includes 2400 fingerspelling sequences from 4 native ASL signers. The RWTH-PHOENIX-Weather Corpus [7] I added the citation but not sure whether [10] or [7] is correct –KL is a realistic data set of German Sign Language, consisting of sign language videos from 190 television weather forecasts. However, its visual variability is still fairly controlled (e.g. uniform background, consistent video frame rate) and it contains a small number of signers (9 signers) signing in a fairly formal style appropriate for weather broadcasts. The recently introduced

[2]There has also been significant work on sign language recognition using specialized equipment such as depth sensors (e.g., [22, 9]). In this paper we consider video-only input, as it is more practical and abundant in naturally occurring online data.

Chicago-Fingerspelling-in-the-Wild (ChicagoFSWild) data set [24] consists of 7304 fingerspelling sequences from online videos. This data set includes a large number of signers (168) and a wide variety of challenging visual conditions, and we use it as one of our test beds.

Automatic sign language recognition tasks are commonly addressed with approaches combining ideas from computer vision and speech recognition. A variety of sign language-specific visual features have been proposed in prior work, including ones based on estimated position and movement of the hand combined with appearance descriptors.add citations. also, this is for fingerspelling only? for sign language in general you need more than the hand –KL Recent work has had more success with convolutional neural network (CNN)-based features [15, 16, 18, **?**, 17, 23, 24]. The visual features are then fed into sequence models such as hidden Markov models [13, 15, 16, 17]is [17] hmm-based? –KL, segmental conditional random fields [14, 12], and recurrent neural networks (RNNs) [23].other rnn-based citations? –KL In this paper, we focus on end-to-end sequential models combining convolutional and recurrent neural layers due to their simplicity and recent success for fingerspelling recognition [23].

Sign language recognition is also related to pose estimation, more specifically articulated hand pose estimation for the case of fingerspelling. There has been extensive work on hand pose estimation, and some models (e.g., [25]) have shown real-life applicability. However, directly applying hand pose estimation to our real-life fingerspelling data is very challenging. Fingerspelling consists of quick, fine-grained movements and often has occlusion, and typical web-quality video is particularly visually challenging. Figure 4 shows typical examples of running an off-the-shelf pose estimation model on video from our data set.

Much previous work on sign language recognition, and the vast majority of previous work on fingerspelling recognition, uses some form of hand detection or segmentation to localize the region of interest as an initial step. Kim *et al.* [13, 14, 12] estimate a signer-dependent skin color model based on mixture of Gaussians using manually annotated hand regions in a small number of frames per signer. Huang *et al.* [9] learn a hand detector based on Faster R-CNN [] using manually annotated signing hand bounding boxes, and apply it to general sign language recognition. Shi *et al.* [23] train a custom *signing* hand detector for fingerspelling recognition on the ChicagoFSWild data set, which avoids detecting the non-signing hand during fingerspelling, and find that this vastly improves performance over a model based on the whole image. Some sign language recognition approaches use no hand or pose pre-processing, using the entire image as input (e.g., [19]), and indeed many signs involve large motions that do not require fine-grained gesture understanding. However, for fingerspelling recognition it is particularly important to understand fine-grained distinctions in handshape. edited this par. it seemed to be saying hand detection doesn't work on data in the wild, but we did use it with good effect in the SLT paper –KL

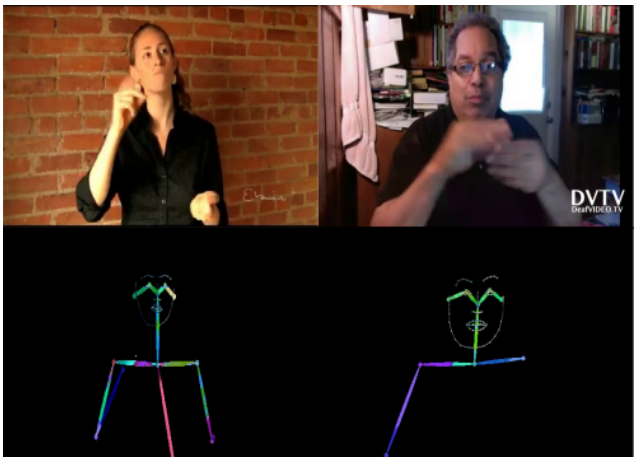

Figure 4. Failure cases of an off-the-shelf hand pose estimator on fingerspelling images in the dataset. Pose estimation is based on Openpose [1]. show sequence of frames from each video with some successes/failures in hand pose estimation. possibly move this fig to supplementary material. –KL

The most closely related work to ours is that of Shi *et al.* [23], which first addressed fingerseplling recognition in the wild. In contrast to this prior work, we propose an end-to-end approach that directly transcribes a sequence of image frames into letter sequences, without a dedicated hand detection step. To our knowledge this is the first attempt to address the continuous fingerspelling recognition problem, or any sign language recognition in similarly challenging visual conditions, without relying on hand detection.not sure we should mention "or any sign language..." as hermann ney et al. might disagree –KL We also contribute the first attempt at crowd-sourced sign language annotation for large-scale data collection.

3. Model

In this section we describe our approach for fingerspelling recognition with two subsections respectively focused on attention model and the iterative zooming-in approach.

3.1. Attention-based recurrent neural network

We now describe how to transcribe a signing tube, represented by a sequence of image patches $I_1, I_2, ..., I_T$ into the fingerspelled word w. As we deal with the lexicon-free scenario, in which the lexicon size is unlimited, the word w is represented as a sequence of letters $w_1, w_2, ..., w_s$. To extract the features of image sequence of image sequence, one possible way is to apply a 2D-CNN on individual frames

and use a recurrent neural network such as LSTM on the top to incorporate temporal structure or directly apply a 3D-CNN to get spatial-temporal representation of the frame sequence. One potential problem with the above approach is that the model is not equipped with the mechanism to focus on the informative part of an image. In our case, most information is conveyed by the hand constituting only one part of the whole image. The capacity to distinguish hand and background has to be learned implicitly which can be hard in a setting where only sequence-level labels are available. This is made worse as we often need to enlarge the hand bounding boxes to incorporate whole hand. Such issue can be mitigated by an attention mechanism.

Our attention model is based on convolutional recurrent architecture (see figure 5). At timestep t, a fully convolutional neural network is applied on the image frame I^t to extract a feature map f^t. Suppose hidden state of recurrent unit at timestep $t-1$ is e^{t-1}, we compute the attention map β^t based on f^t and e^{t-1}:

$$v_{ij}^t = v_f^T tanh(W^d e^{t-1} + W^f f_{ij}^t)$$
$$\beta_{ij}^t = \frac{exp(v_{ij}^t)}{\sum_{i,j} exp(v_{ij}^t)} \tag{1}$$

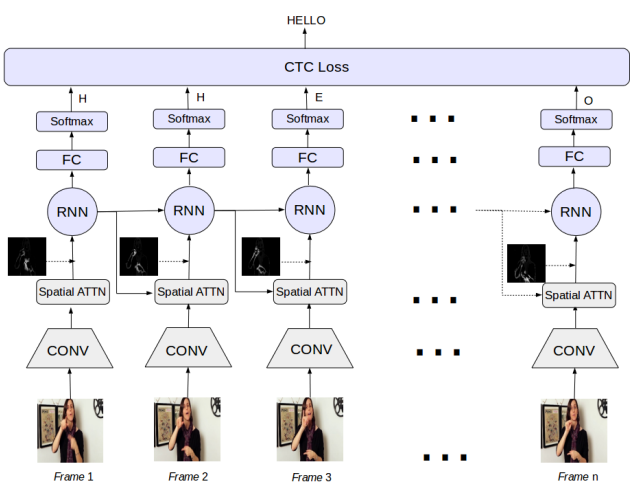

Figure 5. Attention-based recurrent convolutional neural network

Attention map $\boldsymbol{\beta}_t$ reflects the knowledge learned by the model on importance of feature at different spatial locations to the letter sequence. Here we also introduce a prior term $\mathbf{M}$, which represents prior knowledge we have on the importance of spatial locations. For instance we get obtain $\mathbf{M}$ by using optical flow as regions in motion are more likely to be of signing hands compared to the static regions, most of which are background objects. The visual feature at timestep t is a weighted average of f_{ij}^t, $1 \leq i \leq h, 1 \leq j \leq w$, where w and h are width and height of the feature map respectively. α controls the relative weight of prior and attention weights learned by the model.

$$h^t = \sum_{i=1}^{h} \sum_{j=1}^{w} \frac{\beta_{ij}^t (M_{ij}^t)^\alpha f_{ij}^t}{\beta_{ij}^t (M_{ij}^t)^\alpha} \tag{2}$$

The state of recurrent unit at timestep t is updated as equation 3.

$$e^t = LSTM(e^{t-1}, h^t) \tag{3}$$

Once we get the spatial-temporal features for the image frames, the next step is to decode that sequence into words: $(e_1, e_2, ..., e_T) \rightarrow (\mathbf{l}_1, \mathbf{l}_2, .., \mathbf{l}_l)$. Here we employ connectionist temporal classification (CTC) for decoding, which does not rely on the frame-letter alignment. More formally, for an input sequence of visual features $\mathbf{e}$ of length T, we define a continuous map $\mathcal{N}_w : (\mathcal{R}^m)^T \mapsto (L')^T$ representating the transformation from visual feature to frame-level label and a many-to-one map $\mathcal{B} : {L'}^T \mapsto L^{\leq T}$ where $L^{\leq T}$ is the set of all possible labelings. Let $L' = L \cup \{blank\}$, y_k^t the probability of observing label k at time t, the first step is to compute the probability of any possible labeling $\pi \in {L'}^T$:

$$p(\pi|\mathbf{e}_{1:T}) = \prod_{t=1}^{T} y_{\pi_t}^t = \prod_{t=1}^{T} \underset{\pi_t}{\text{softmax}}(\mathbf{A}^e \mathbf{e}_{\pi_t}^t + \mathbf{b}^e) \tag{4}$$

The next step is to compute the probability of a given labeling $\mathbf{l}$ by summing over all the possible labeling π (equation 5), which can be computed by CTC forward-backward algorithm.

$$p(\mathbf{l}|\mathbf{e}_{1:T}) = \sum_{\pi \in \mathcal{B}^{-1}(\mathbf{l})} p(\pi|\mathbf{e}_{1:T}) \tag{5}$$

In order to efficiently compute $p(\mathbf{l}|\mathbf{e}_{1:T})$, it is assumed in CTC that network outputs at different timesteps are conditionally independent given the hidden state at every timestep and alignment between frame and label is monotone. The first assumption is reasonble in practice given the enough representation capacity of encoder, which is convolutional recurrent neural network. The monotonicity generally holds for the case of fingerspelling as the order of frame and letters are consistent with each other.

3.2. Iterative zooming

As we are under a recognition setting where the input is a sequence of coarse-grained image frames, signing hand(s) only constitute a small portion of the whole frame. Attention mechanism equips the model with capacity to focus on informative regions, the issue of low resolution in signing

regions persists. This problem is in our recognition setting as fingerspelling often involves fine-grained motions and minor difference in handshape. One straightforward way to mitigate this issue is to enlarge the size of inputting images. However, as the convolutional recurrent encoder covers the full image sequence increasing size of whole image can lead to prohibitively large memory footprints especially for the training.

To address the problem of low resolution, we propose to iteratively refine the input image frames based on the attention map. Given a trained attention model $\mathcal{M}$, we run inference step with $\mathcal{M}$ on target image sequence $\mathbf{I}_1, \mathbf{I}_2, ..., \mathbf{I}_T$ to generate the associated sequence of attention maps: $M_1, M_2, ..., M_T$. The sequence of attention maps are utilized to get new sequence of images $I'_1, I'_2, ..., I'_T$. At training time, $I'_1, I'_2, ..., I'_T$ are used to train a new model $\mathcal{H}'$. This iterative process runs S steps until image of sufficiently high resolution is obtained. Given a series of zooming ratios $r_1, r_2 ..., r_s$, the zooming process consists of finding a series of bounding box sequences $\{b_t^1\}_{1 \le t \le T}, ..., \{b_t^S\}_{1 \le t \le T}$. The zooming ratio is defined as ratio between size of bounding box and full frame. How to select the series of ratios will be detailed in experimental section. S models are trained in the above iteraive process. At test time, input image sequence $I_1, I_2, ..., I_T$, $\mathcal{H}_{1:S-1}$ is run subsequently to get sub-region sequence $I_1^{S-1}, I_2^{S-1}, ..., I_T^{S-1}$ on which $\mathcal{M}_S$ are applied for word decoding. The above process is illustrated in algorithm 1.

In each iteration, the objective is to find a sequence of bounding boxes $\{b_1, b_2, ..., b_T\}$ based on attention map sequence $\{M_1, M_2, ..., M_T\}$. We assign a score s_t^i to each box b_t^i defined as the its center value in attention map $\mathbf{M}_i$. We also define a linking score between two bounding boxes b_t^i in two consecutive frames as equation 6

$$e(b_t^i, b_{t+1}^j) = s_t^i + s_{t+1}^j + \lambda * IoU(b_t^i, b_{t+1}^j) \qquad (6)$$

, where $IoU(b_t^i, b_{t+1}^j)$ is the intersection over union of b_t^i and b_{t+1}^j and λ is a hyperparameter measuring the relative weight between the box score and smoothness. Using intersection over union has a smoothing effect and ensures the framewise bounding box does not shift between different hands. Such formulation is analogous to finding "action tube" in action recognition [8]. Finding the sequence of best bounding boxes can thus be turned into an optimization problem as equation 7, which can be efficiently solved by a Viterbi-like dynamic programming. Once the zooming boxes are found, we take average of all boxes for further smoothing.

$$E(l) = \frac{1}{T} \sum_{t=1}^{T-1} e(b_t^{l_t}, b_{t+1}^{l_{t+1}}) \qquad (7)$$

Algorithm 1 Iterative zooming

Training, Input: $\{(\mathbf{I}_{1:T_n}^{n,0}, \mathbf{w}^n)\}_{1 \le n \le N}$

for $s \in \{1, 2, ..., S\}$ **do**
  Train model $\mathcal{H}_s$ with $(\mathbf{I}_{1:T_n}^{n,s-1}, \mathbf{w}^n)_{1 \le n \le N}$
  for $n = 1, ...N$ **do**
    Run inference on $\mathbf{I}_{1:T_n}^n$ with $\mathcal{H}_s$ to obtain attention map $\mathbf{M}_{1:T_n}^n$
    Solve equation 7 to obtain sequence of bounding boxes $b_{1:T_n}^n$
    Crop and resize $\mathbf{I}_{1:T_n}^{n,0}$ with $b_{1:T_n}^n$ to get $\mathbf{I}_{1:T_n}^{n,s}$
  end for
end for
Return $\mathcal{H}_s$, $1 \le s \le S$

Test, Input: $\mathbf{I}_{1:T}^0$

for $s \in \{1, 2, ..., S\}$ **do**
  Run inference on $\mathbf{I}_{1:T}^{s-1}$ with $\mathcal{H}_s$ to obtain attention map $\mathbf{M}_{1:T}$ and predicted words $\mathbf{w}^s$
  Solve equation 7 to obtain sequence of bounding boxes $b_{1:T}$
  Crop and resize $\mathbf{I}_{1:T}^0$ with $b_{1:T}$ to get $\mathbf{I}_{1:T}^s$
end for
if Ensemble **then**
  Return $\mathbf{w}^S$
else
  Return $Ensemble(\mathbf{w}^{S-k}, ..., \mathbf{w}^S)$
end if

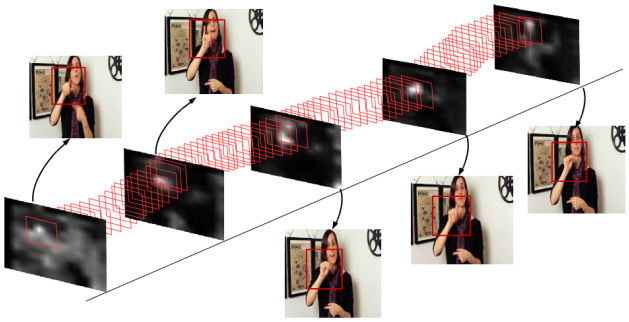

Figure 6. Illustration of finding zoomed-in ROI sequence based on attention maps in one iteration. remove the diagonal line connecting image corners –KL

Throughout this paper, we pre-set ratio between size of bounding box and the input image frame in each iteration. Tuning the "zooming ratio" and number of iterations will be detailed in experimental section. Overall in the iterative approach, a total of S models are trained and $S-1$ models are used for the purpose of generating images of high resolution and model trained in last iteration is for word decoding. One can also ensemble models in several iterations for testing.

4. Data

I split off the data section from the expts section since it is a contribution, and also added some more info about the mturk collection –KL We use two data sets: Chicago Fingerspelling in the Wild (ChicagoFSWild) [24], a previously existing data set; and our newly collected data set with crowdsourced annotations, FSWildCrowd. Both data sets contain clips of fingerspelling sequences excised from sign language video "in the wild", collected from online sources such as YouTube and deafvideo.tv. However, ChicagoFSWild was carefully annotated by linguistics students, while FSWildCrowd uses crowdsourced annotations. ChicagoFSWild contains 5455 training sequences from 87 signers, 981 development sequences from 37 signers, and 868 test sequences from 36 signers, with no overlap in signers in the training/development/test sets.[3]

One goal of our work is to enable quicker data collection for sign language research. To this end, we have developed a fingerspelling video annotation interface, based on VATIC [**?**],show vatic screenshot? –KL and have used it to collect our new data set, FSWildCrowd, by crowdsourcing the annotation process via Amazon Mechanical Turk [**?**]. Annotators are presented with one-minute clips from sign language videos, and are asked to mark the start and end frame of fingerspelling sequences within the clips (if any fingerspelling is present). Annotators also provide a transcription (a sequence of English letters) for each fingerspelling sequence, but do not align the transcribed letters to video frames. Two annotators are used for each clip, and both annotations are included in FSWildCrowd. No post-processing or cleanup of the annotated data is done.is the last sentence true? are we using any of the proofread data? –KL Compared to ChicagoFSWild, therefore, less researcher effort is put into collection, annotation, and proofreading in FSWildCrowd.

The videos in FSWildCrowd include sources such as webcam videos and online lectures, and include varied viewpoints and styles. [More description and statistics on the MTurk data] yes please –KL

FSWildCrowd includes 24,086 training sequences from 122 signers, 4025 development sequences from 22 signers, and 1715 test sequences from 22 signers. The split into training, development, and test sets has been done in such a way as to approximately evenly distribute certain attributes (such as signer gender and handedness) between the three sets. In addition, order to enable clean comparisons between results on ChicagoFSWild and FSWildCrowd, we used the signer labels in the two data sets to ensure that there are no overlaps in signers between the ChicagoFSWild training set and the FSWildCrowd test set.please check my edits –KL

[3] According to [24], the signer identities were determined manually, so could potentially include overlaps due to mistaken identity. We follow the same procedure here.

5. Experimental Setup

In this section we describe the data we use, experimental results and analysis on our approach. Note all experiments are done in signer-independent setting.

5.1. Implementation Details

Preprocessing The objective of image pre-processing here is to roughly unify the scale of hands in different input sequences. As our data are from videos with a large variety of viewpoints, the scale of hands of vary in a wide range across different input sequences. For instance proportion of hand in an image from a webcam video can be several times larger than that in an image of third-person view. As a preprocessing step, we first run an off-the-shelf face detector on image frames to obtain the face bounding box and rescale image according to the size of the bounding box to ensure hand scale consistent in every sequence.

Our face detection is based on the implementation [2], which is trained on WIDER dataset [27]. To save computation we run face detector on one in every five frames in each sequence. We then take the average of all bounding box for the whole sequence. In cases when multiple faces are detected, we first find a smooth "face tube" by successively taking the bounding box in next frame which has highest IoU with the face bounding box in current frame. For every tube, a motionness score is defined as the average value of a surrounding region ($3\times$ size of bounding box) in the optical flow, which are calculated based on two-frame motion estimation algorithm by Farneback [6]. Finally the tube with highest score is selected and again the box is averaged for the whole sequence. In cases where face detection fails, we use the mean of all face bounding boxes detected in all images of same size in the training set. We empirically observe the failure case (no face detected) is rare ($\approx 0.5\%$ in training set).

With detected face bounding box for an input sequence, we experiment on two setups with different input: (1) face-based ROI (2) whole frame. In first case, a large region centered on the detected face is cropped and resized to serve as input. Particularly in case of fingerspelling (and general sign language), the signing hand(s) are spatially close to the face. Cropping the signing region given face bounding box can gives a good initialization to the input of our approach. Specifically we crop a region centered on the bounding box which is of size 3 times larger. The ROI is resized with a ratio of $\frac{224}{max(w_{roi},h_{roi})}$ and then padded on short side to make a squared target image of size 224×224. In latter type, we input the whole image frame with consistent scale of face. The objective is to remove any artifact aroused by cropping in preprocessing steps. As the image size and its associated bounding boxes are scattered in very wide range,

we unify the scale in two-step manner. Specifically we set a base size (w_b, h_b) for bounding box and a maximum size (W_{max}, H_{max}) for image. Input image $(W_I \times H_I)$ with a bounding box of size $w_I \times h_I$ are rescaled with a ratio of $\frac{max(w_I,h_I)}{(w_b,h_b)}$. If the size of rescaled image is larger than $W_{max} \times H_{max}$ we further resized the input image by ratio of α so that the area of resulting image is smaller than $W_{max} \times H_{max}$. α is multiplied in the iterative zooming-in process for that input sequence. Note input images can be of different sizes under this setup. [Some explanations on this two-step rescaling].

Model training We base convolutional layers of our model on AlexNet[20] pre-trained on ImageNet[4]. The last max-pooling layer of AlexNet is also removed so that we have a sufficiently large feature map. When input of size 224×224, the extracted feature map is of size 13×13. A deeper network like VGG [26] cannot be used due to the memory requirements introduced by its small stride. To prevent over-fitting, we added 2D-dropout layers between last three convolutional layers with drop rate being 0.2 in each layer. For recurrent neural network, we used one-layer LSTM with 512 hidden units. The model is trained with SGD at initial learning rate of 0.01 for 20 epochs and a decayed learning rate of 0.001 for additional 10 epochs. We use dev set for early stopping.

For iterative zooming, we select the zooming ratio based on beam search. Zooming ratio are selected from $\{0.9, 0.81, 0.729, 0.6561\}$ (the power of 0.9) for each iteration and beam size is 2. To prevent over-tuning, we use the dev set of MTurk dataset for choosing the series of zooming ratio. In viterbi decoding (equation 7), λ is tuned to be 0.1. We select among 2 candidate bounding boxes at each timestep, mainly for sake of preventing switchingbetween two hands in neighboring frames.

6. Results and Analysis

6.1. Main Results

	inhouse test	MTurk test
[24], unscaled whole	12.7	
[24], scaled whole		
[24], hand	41.9	41.2
Ours, face	**44.2**	46.8
Ours, scaled whole	41.5	

Table 1. Comparison of letter accuracies (%) between our approach and previous work. Unscaled whole: whole frame as input. Scaled whole: whole frame scaled with face detection as input. Hand: signing hand as input. Face: an enlarged region surrounding face bounding box as input

Table 1 shows the main results of our approach on two different inputs as well as the results of prior work [24]. Compared to the prior work, we achieve better and comparable performance under face ROI and whole frame setting without using hand detection based on manually annotated hands on the dataset. Though using high-resolution ROI of signing hand can be directly obtained with a special-purpose signing hand detector, the error in detector itself can have negative impact on the recognition accuracy. Such impact is prominent in the "wild" case where images are much more noisier and high variance exists in signing hands. Using a large region can avoid the loss of information in the preprocessing step. Though an off-the-shelf face detector is leveraged in both setups, the detector is not trained with any manually face annotations in our dataset. Compared to signing hand with higher variability, detecting face is more robust and large training dataset is more accessible.

Though loss of information can be caused under specific scenarios in Face-ROI setup, the high performance shows the assumption that face and signing hand are spatially close to each other is quite reasonble in the specific domain of fingerspelling. The whole frame setup consists of even less supervision compared to Face ROI. Only the size of detected face bounding box is used for training and no cropping is involved in pre-processing step. The relative worse performance compared to Face ROI is due to the complexity of our image data. In cases of multiple moving objects in one same image, the zooming-in process can fail.

6.2. Analysis

Our approach comprises of attention model, iterative zooming, model ensemble and language model. To see how those factors contribute to the performance we did an ablatiion study and results are show in table 2.

	Face-ROI	Whole
Attn only	33.4	14.2
+ Iter zoom	44.8	42.3
+ Ensemble	45.0	43.0
+ lm	46.2	44.1

Table 2. How different factors contribute to final performance (on dev)

Effect of iterative zooming The process of iterative zooming is important to ensure the high performance, as can also be seen from table 2. In both setups, raw inputs are only a coarse-grained image where hand only consists of a small portion. Figure 8 shows how accuracy and input image changes in different zooming iterations. Though no supervision regarding hand is used for training, the location of signing hand is implicitly learned by the model through the attention mechanism. The accuracy is proportional to the scale of hands in the input image.

As the input image is zoomed in, the resolution of hand increases. In addition, the gradual shrinkage of input im-

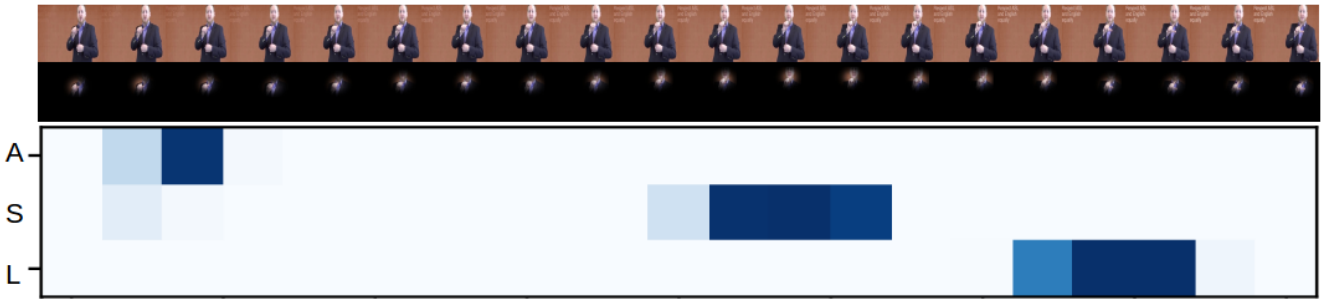

Figure 7. Alignment between letter labels and image frames as well as the attended region in each frame. revise as discussed in greg's group meeting. –KL

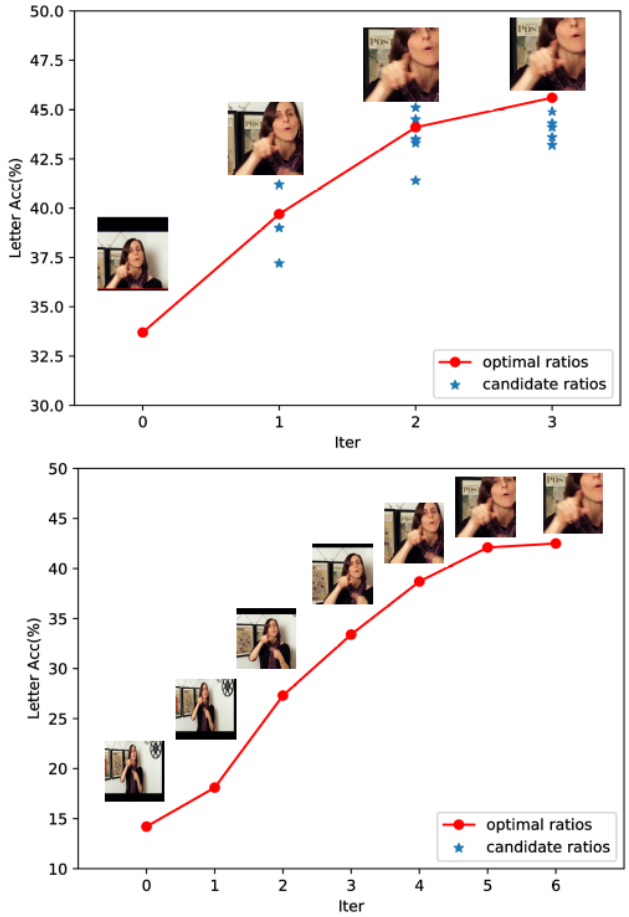

Figure 8. How accuracy and input image change in each iteration. The red curve corresponds to the sequence of zooming ratios obtained by beam search. the bottom panel doesn't have results with candidate ratios. also, label each point on the red curve with the zooming ratio. –KL

age also removes the background generally being noise and irrelevant to the recognition. To see how this removal factor contributes to the performance, we compared the accuracy of iterative zooming with zooming ratio series

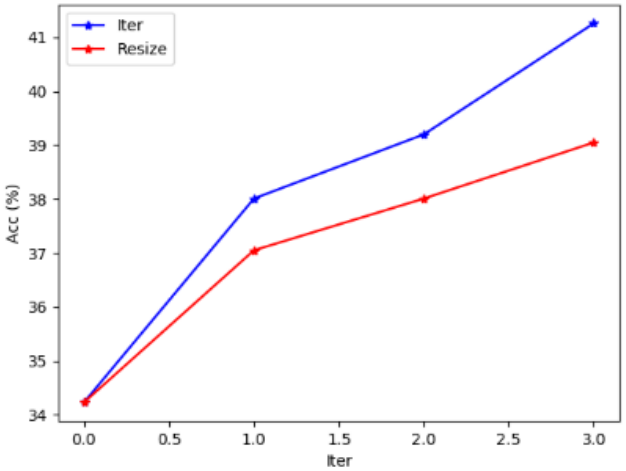

Figure 9. Comparison of accuracy between zooming-in and image resizing. Size of hands are same in same iteration.

$(0.9, 0.8, 0.7)$ and our attention model with input image enlarged with those ratios respectively (see figure **??**). Under same ratio, the zooming approach outperforms model trained with enlarged input though the resolution of hands in two inputs are same. This shows the benefits of noise removal brought by the zooming process.

Effects of model ensemble, language model Model ensemble and rescoring with a language model bring additional improvements according to table 2. Model ensembling can mitigate the problem of over-zooming. Hand scale is not always consistent across different input sequences, which can be caused by the error in face detection or the irregular ratio between face and hand, As we use one fixed zooming ratio for every sequence in one iteration, parts of signing hand can be removed for certain images. Compared to the last iteration, input images of preceding iterations are more complete despite its lower resolution. Besides, in each iteration model is trained with images of a certain scale. Ensembling can make the model less sensitive to the scale change at test time.

In terms of language model, we train an LSTM with 200

hidden units with the inhouse training data. The dev set perplexity of our language model is 17.3. Rescoring with language model improves the performance by a small margin ($\approx 1\%$), which mainly because fingerspelling is used for words without sign form and does not follow the common distribution of English words.

6.3. Training with MTurk data

	inhouse test	MTurk test
[24], Hand	53.6	53.4
Ours, face	**58.0**	**58.4**

Table 3. Comparison of letter accuracies (%) with training data augmented by MTurk training set. Hand: signing hand as input. Face: an enlarged region surrounding face bounding box as input

All previous results are trained with inhouse training set. Those fingerspelling annotations have gone through carefully proofreading. Compared to the above training set, the annotations of MTurk data is much more noisy and there exists potential domain discrepency between this dataset and inhouse test set. However, the data size is 4 times larger than the inhouse training set. To validate whether those large amount of noisy data is useful for training fingerspelling recognizer, we merged both training set and test on inhouse test set. Both annotations are kept in MTurk training data. The training data amounts to # with only # from clean inhouse data. We follow the face-ROI setup decribed above as better accuracy is achieved under this setup. As can be seen from results in table 3, using the MTurk data significantly improves the performance. The gain is partly attributed to the increasing size of training set. Besides, keeping both annotations performs naturally as data augmentation, which further increases accuracy of the model.

7. conclusion

In this paper we present a new model for fingerspelling recognition based on attention mechanism and an iterative approach to zoom into ROI of a larger input image. We show the signing hand is gradually located with attention map and thus input with higher resolution can be used to train the model. Without using any hand annotations our approach outperforms the recognition model based on hand detection. Besides, training with large number of crowdsouring fingerspelling data further improves the recognition accuracy to a large extent.

References

[1] Openpose: Real-time multi-person keypoint detection library for body, face, hands, and foot estimation. https://github.com/CMU-Perceptual-Computing-Lab/openpose. 3

[2] The world's simplest facial recognition api for python and the command line. https://github.com/ageitgey/face_recognition. 6

[3] V. Athitsos, C. Neidle, S. Sclaroff, J. Nash, A. Stefan, A. Thangali, H. Wang, and Q. Yuan. Large lexicon project: American Sign Language video corpus and sign language indexing/retrieval algorithms. In *Workshop on the Representation and Processing of Sign Languages: Corpora and Sign Language Technologies (CSLT)*, 2010. 2

[4] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and F. Li. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009. 7

[5] E. Efthimiou, S. Fotinea, T. Hanke, J. Glauert, R. Bowden, A. Braffort, C. Collet, P. Maragos, and F. Lefebvre-Albaret. Sign language technologies and re- sources of the dicta-sign project. In *LREC Workshop on the Representation and Processing of Sign Languages: Interactions between Corpus and Lexicon*, 2012. 2

[6] G. Farneback. Two-frame motion estimation based on polynomial expansion. In *Proceedings of the 13th Scandinavian Conference on Image Analysis*, 2003. 6

[7] J. Forster, C. Schmidt, O. Koller, M. Bellgardt, and H. Ney. Extensions of the sign language recognition and translation corpus RWTH-PHOENIX-Weather. *Computer Vision and Image Understanding*, 141:108–125, 12 2015. 2

[8] G. Gkioxari and J. Malik. Finding action tubes. In *CVPR*, 2015. 5

[9] J. Huang, W. Zhou, Q. Zhang, H. Li, and W. Li. Video-based sign language recognition without temporal segmentation. In *AAAI*, 2018. 2, 3

[10] F. Jens, S. Christoph, H. Thomas, K. Oscar, Z. Uwe, P. Justus, and N. Hermann. RWTH-PHOENIX-Weather: A large vocabulary sign language recognition and translation corpus. *Language Resources and Evaluation*, pages 3785–3789, 2012. 2

[11] J. Keane. *Towards an articulatory model of handshape: What fingerspelling tells us about the phonetics and phonology of handshape in American Sign Language*. PhD thesis, University of Chicago, 2014. 1

[12] T. Kim, J. Keane, W. Wang, H. Tang, J. Riggle, G. Shakhnarovich, D. Brentari, and K. Livescu. Lexicon-free fingerspelling recognition from video: Data, models, and signer adaptation. *Computer Speech and Language*, pages 209–232, November 2017. 2, 3

[13] T. Kim, G. Shakhnarovich, and K. Livescu. Fingerspelling recognition with semi-Markov conditional random fields. In *ICCV*, 2013. 3

[14] T. Kim, W. Wang, H. Tang, and K. Livescu. Signer-independent fingerspelling recognition with deep neural network adaptation. In *ICASSP*, 2016. 3

[15] O. Koller, H. Ney, and R. Bowden. Deep hand: How to train a CNN on 1 million hand images when your data is continuous and weakly labelled. In *CVPR*, 2016. 3

[16] O. Koller, S. Zargaran, and H. Ney. Re-sign: Re-aligned end-to-end sequence modelling with deep recurrent CNN-HMMs. In *CVPR*, 2017. 3

[17] O. Koller, S. Zargaran, and H. Ney. Recurrent convolutional neural networks for continuous sign language recognition by staged optimization. In *CVPR*, 2017. 3

[18] O. Koller, S. Zargaran, H. Ney, and R. Bowden. Deep sign: Hybrid cnn-hmm for continuous sign language recognition. In *BMVC*, 2016. 3

[19] O. Koller, S. Zargaran, H. Ney, and R. Bowden. Deep sign: Enabling robust statistical continuous sign language recognition via hybrid cnn-hmms. *International Journal of Computer Vision*, 126(12):1311–1325, 2018. 2, 3

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *NIPS*, 2012. 7

[21] C. Padden and D. Gunsauls. How the alphabet came to be used in a sign language. *Sign Language Studies*, pages 10–33, 4 (1) 2003. 1

[22] N. Pugeault and R. Bowden. Spelling it out: Real-time ASL fingerspelling recognition. In *Proceedings of the 1st IEEE Workshop on Consumer Depth Cameras for Computer Vision, jointly with ICCV*, 2011. 2

[23] B. Shi and K. Livescu. Multitask training with unlabeled data for end-to-end sign language fingerspelling recognition. In *ASRU*, 2017. 3

[24] B. Shi, A. M. D. Rio, J. Keane, J. Michaux, D. Brentari, G. Shakhnarovich, and K. Livescu. American sign language fingerspelling recognition in the wild. In *SLT*, 2018. 2, 3, 6, 7, 9

[25] T. Simon, H. Joo, I. Matthews, and Y. Sheikh. Hand keypoint detection in single images using multiview bootstrapping. In *CVPR*, 2017. 3

[26] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 7

[27] S. Yang, P. Luo, C. C. Loy, and X. Tang. Wider face: A face detection benchmark. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 6