Stochastic Fine-grained Labeling of Multi-state Sign Glosses for Continuous Sign Language Recognition

Zhe Niu[0000−0001−5833−1142] and Brian Mak

Department of Computer Science & Engineering
The Hong Kong University of Science & Technology
{zniu,mak}@cse.ust.hk

Abstract. In this paper, we propose novel stochastic modeling of various components of a continuous sign language recognition (CSLR) system that is based on the transformer encoder and connectionist temporal classification (CTC). Most importantly, We model each sign gloss with multiple states, and the number of states is a categorical random variable that follows a learned probability distribution, providing stochastic fine-grained labels for training the CTC decoder. We further propose a stochastic frame dropping mechanism and a gradient stopping method to deal with the severe overfitting problem in training the transformer model with CTC loss. These two methods also help reduce the training computation, both in terms of time and space, significantly. We evaluated our model on popular CSLR datasets, and show its effectiveness compared to the state-of-the-art methods.

1 Introduction

Sign language is the primary communication medium among the deaf. It conveys meaning using gestures, facial expressions and upper body posture, etc., and has linguistic rules that are different from those of spoken languages. Sign language recognition (SLR) is the task of converting a sign language video to the corresponding sequence of (sign) glosses (i.e., "words" in a sign language), which are the basic units of the sign language semantics. Both isolated sign language recognition (ISLR) [15] and continuous sign language recognition (CSLR) have been attempted. ISLR classifies a gloss-wise segmented video into its corresponding gloss, whereas CSLR classifies a sentence-level sign video into its corresponding sequence of glosses. The latter task is more difficult and is the focus of this paper.

Most of the modern CSLR architectures contain three components: visual model, contextual model and alignment model. The visual model first extracts the visual features from the input video frames, based on which the contextual model further mines the correlation between the glosses. Convolutional neural networks (CNNs) and recurrent neural networks (RNNs) are commonly used architectures for the visual and contextual model, respectively. In CSLR, sign glosses occur (time-wise) monotonically with the corresponding events in the

video. Thus, an alignment model is required to find the proper mapping between the video frames and glosses such that the model can be trained. Methods like [12, 14, 11] align the video frames to glosses by applying Viterbi search on the hidden Markov models (HMMs). While others [2, 5, 25, 27, 17] adopt the connectionist temporal classification (CTC) method, where a soft full-sum alignment is calculated as the final training objective. For both HMM-based and CTC-based CSLR models, it is usually necessary to fine-tune the lower-level visual feature extractor during model training, as it has been shown that the visual network cannot learn effective features in end-to-end training [14, 17].

To address this problem, we propose to use the transformer encoder [22] as the contextual model for CSLR, which has been shown effective in tasks such as machine translation [22, 6] and speech recognition [16]. The residual connections between layers in the transformer encoder help backpropagate the errors better to the visual model. Moreover, to improve model robustness and to alleviate the overfitting problem, we propose dropping video frames stochastically and randomly stopping the gradients of some frames during training. We call these two procedures *stochastic frame dropping* (SFD) and *stochastic gradient stopping* (SGS), respectively. More importantly, we perform detailed modeling and allow each gloss model to have multiple states, but the number of states for each gloss model is variable and is modeled by a probability distribution that is trained jointly with the rest of the system. We named this method *stochastic fine-grained labeling* (SFL). SFL provides stochastic finer-grained labels for the CTC loss, thus provides more supervision in the temporal domain.

Overall, the main contributions of our work are:

1. We propose stochastic frame dropping (SFD) and stochastic gradient stopping (SGS) to reduce video memory footprint, improve model robustness and alleviate the overfitting problem during model training.
2. We introduce stochastic fine-grained labeling (SFL) to model glosses with multiple states. The number of states of any gloss is variable and follows a probability distribution. As a result, the performance of our SLR model is further improved.

The rest of this paper is organized as follows: In Section 2, we review related works. Section 3 introduces the use of stochastic modeling in three components of our model. Section 4 presents the experimental evaluation of our proposed methods and discussions on the findings. Finally we conclude in Section 5.

2 Related Works

In the past, many works [12] tackle the CSLR problem using the Gaussian mixture model-hidden Markov model (GMM-HMM) with hand-crafted visual features. Since the features are hand-crafted, they are usually not robust and not optimal as they are not optimized jointly with the rest of their CSLR systems.

To leverage the power of deep learning for automatic feature extraction, hybrid models [13, 14, 11] are proposed, which combine deep neural network models

with HMM. As HMM requires the computation of priors, these methods are usually trained with epoch-wise re-alignment. To avoid prior estimation, some other methods [27, 25, 17] try to replace HMMs with the connectionist temporal classification (CTC) method, which provides a soft full-sum alignment and re-aligns after each mini-batch. However, as the visual model fails to learn representative features, epoch-wise iterative fine-tuning of the visual model is required.

Sequence-to-sequence (Seq2Seq) architecture [1] has also been attempted for CSLR, in which, the alignments are calculated by a global weighted summation of the input sequence. As there is no monotonicity constraint, the model tends to misalign. Moreover, since the decoder takes the ground truth labels as inputs during training, the model suffers from the exposure bias problem, where errors accumulate with each decoding step during testing. [26] utilizes a transformer-based Seq2Seq model and applies reinforcement learning to alleviate exposure bias. On the other hand, [17] uses CTC in additional to a Seq2Seq decoder to make up for the performance degradation. Work [3] proposes to jointly train the SLR and SLT tasks in a CNN-Transformer framework by combining the CTC loss on sign glosses and cross-entropy loss on spoken words, which relies on pretraining the visual model via a CNN+LSTM+HMM setup [11].

In this work, we choose CTC as the alignment model so that we do not need to do re-alignment frequently for the prior estimation in HMM-based models and for alleviating the exposure bias problem in the Seq2Seq architecture. Compared with other CTC works, our method avoid the necessity of having to fine-tune the visual model iteratively after each epoch.

3 Methodology

In this section, we introduce our stochastic multi-states (SMS) framework. We will first give an overview of the framework, then describe stochastic frame dropping (SFD), stochastic gradient stopping (SGS) and stochastic fine-grained labeling (SFL), respectively in detail.

3.1 Framework Overview

The overall framework of our model is presented in Figure 1. The design of the network follows the visual-contextual-alignment model scheme. For the visual model, we choose 2D convolutional neural network (2D-CNN) to extract visual features from individual frames. For the contextual model, we select the transformer encoder with relative positional encoding. The connectionist temporal classification (CTC) is adopted as the alignment model.

Given an RGB video with T frames $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_T)$, the visual model (CNN) first extracts visual features $\mathbf{z} = (\mathbf{z}_1, \ldots, \mathbf{z}_T)$ from individual frames. After this, the contextual model extracts the temporal correlation between the visual vectors. Then, the posterior probabilities of sub-gloss states are calculated based on the features output by the contextual model. At the same time, the SFL module takes the corresponding target gloss sequence $\mathbf{y} = (y_1, \ldots, y_L)$ with length L as

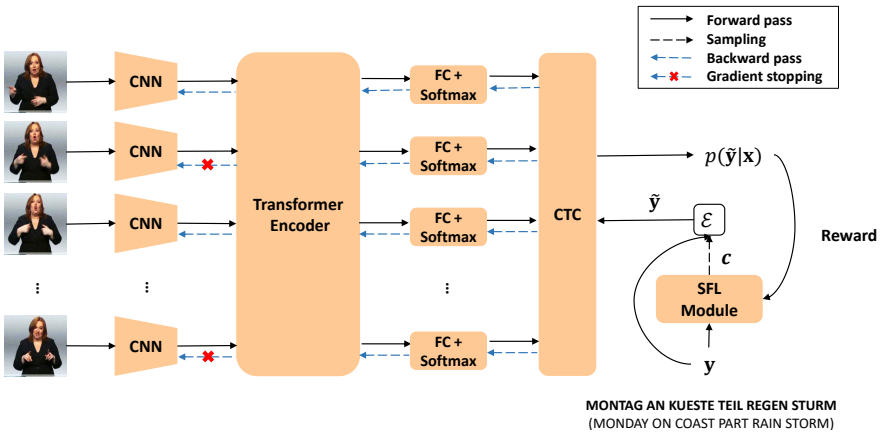

Fig. 1. Overview of our framework. Video frames are first processed by CNN to produce a visual feature sequence which is then fed into a transformer encoder. The output of the transformer encoder is further sent to a fully connected (FC) + softmax layer to produce the probability of sub-gloss states at each time step conditioned on the input video. In the meantime, the target sequence $\mathbf{y}$ is fed into the stochastic fine-grained labeling (SFL) module, and a sequence of sub-gloss state numbers $\mathbf{c}$ is sampled. The extension function $\mathcal{E}(\mathbf{y}, \mathbf{c})$ extends the input gloss sequence $\mathbf{y}$ according to the state number sequence $\mathbf{c}$ to produce a sequence of sub-gloss states $\tilde{\mathbf{y}}$. Based on the probability the sequence produces and the fine-grained label $\tilde{\mathbf{y}}$, CTC calculates the probability of the fine-grained labels $\tilde{\mathbf{y}}$ given the video $p(\tilde{\mathbf{y}}|\mathbf{x})$, which is further used to reward the SFL module for producing state number sequence that leads to higher $p(\tilde{\mathbf{y}}|\mathbf{x})$.

input and generates a probability distribution of the number of states for each gloss. Based on this distribution, a sequence of state numbers $\mathbf{c} = (c_1, \ldots, c_L)$ is sampled and the original gloss sequence is extended to a sub-gloss state sequence $\tilde{\mathbf{y}} = (\tilde{y}_1, \ldots, \tilde{y}_S)$, where $S = \sum_{l=1}^{L} c_l$. Finally, CTC is applied to calculate the posterior probability of the sub-gloss state sequence $p(\tilde{\mathbf{y}}|\mathbf{x})$.

3.2 Visual Model

We use ResNet [8] as the visual model as they are powerful enough but relatively lightweight compared to models like GoogLeNet [21] or VGG [20]. The ResNet is pre-trained on ImageNet [18]. The final fully connected layer is replaced by a linear layer that suits the dimension of our contextual model.

Stochastic Frame Dropping

Due to the limited amount of sign language data, overfitting is a main issue during training. To avoid the network overfitting salient frames and ignoring the less representative ones, we introduce a stochastic frame dropping (SFD)

technique that stochastically drops out some frames during training. The frame dropping also changes the rate of signs and introduces signing speed variations into the training data.

During training, we randomly discard a fixed proportion of frames in a video by uniform sampling without replacement. Suppose there are T' frames in a video originally, and the proportion hyper-parameter is p_{drop}, then $\lceil T' \times p_{\text{drop}} \rceil$ frames will be discarded. During testing, to match the training condition, we evenly select every $\frac{1}{p_{\text{drop}}}$-th frame from the testing video to drop. If $\frac{n}{p_{\text{drop}}}$ is not a whole integer, it will be rounded to the nearest integer.

The SFD mechanism not only augments the data but also improves time efficiency and reduces memory footprint as fewer frames are processed during training and testing.

Stochastic Gradient Stopping

To further prevent overfitting, reduce memory consumption and speed up training, we propose the stochastic gradient stopping (SGS) training technique. This method avoids to compute back-propagation for a part of the input frames during visual feature extraction. Similar to SFD, a hyper-parameter p_{stop} is set for the proportion of frames whose gradient will be stopped. Again, the SGS frames are sampled stochastically and uniformly without replacement. Denote the output of CNN, i.e., the sequence of visual features as $\mathbf{z} = (\mathbf{z}_1, \ldots, \mathbf{z}_T)$, the gradient of the CTC objective with respect to any CNN parameter ϕ can be written as follows:

$$\begin{aligned} \nabla_\phi \log p(\mathbf{y}|\mathbf{x}) &= \sum_{i=t}^{T} \langle \nabla_{\mathbf{z}_t} \log p(\mathbf{y}|\mathbf{z}), \nabla_\phi \mathbf{z}_t \rangle \\ &\approx T \cdot \mathbb{E}_t \left[\langle \nabla_{\mathbf{z}_t} \log p(\mathbf{y}|\mathbf{z}), \nabla_\phi \mathbf{z}_t \rangle \right] \\ &\approx T \cdot \frac{1}{K} \sum_{k=1}^{K} \langle \nabla_{\mathbf{z}_{t_k}} \log p(\mathbf{y}|\mathbf{z}), \nabla_\phi \mathbf{z}_{t_k} \rangle \end{aligned} \tag{1}$$

where $t_k \sim \mathcal{U}\{1, \ldots, T\}$ and $K = \lfloor T \times p_{stop} \rfloor$.

In SGS, the mean gradient of CNN parameter ϕ is approximated by its sample mean, which introduces noise to the gradient to prevent CNN from overfitting. Since a part of the frames are detached from the computation graph and the intermediate outputs of those frames are no longer required to be held for back-propagation, SGS reduces memory footprint during training. Moreover, the training procedure is sped up as less computation of back-propagation is needed.

3.3 Contextual Model

The transformer encoder [22] is adopted as the contextual model to further extract the temporal information between frames. Relative positional encoding [19] is used instead of the absolute position encoding [22]. For each transformer

encoder layer, the relative positional encoding holds two sets of learnable vectors $\{\mathbf{a}_m \in \mathbb{R}^d\}_{m=-M}^{M}$ and $\{\mathbf{b}_m \in \mathbb{R}^d\}_{m=-M}^{M}$. For each attention head, denote the query at the i-th time step as $\mathbf{q}_i \in \mathbb{R}^d$ and the key, value at the j-th time step as $\mathbf{k}_j, \mathbf{v}_j \in \mathbb{R}^d$. Instead of calculating the score before the softmax function as

$$s_{i,j} = \frac{\mathbf{q}_i^\top \mathbf{k}_j}{\sqrt{d}} \ , \tag{2}$$

the relative positional encoding calculates this score by injecting the relative positional information $\mathbf{a}_{\text{clip}(i-j)}$ to give

$$\begin{aligned} s_{i,j} &= \frac{\mathbf{q}_i^\top \mathbf{k}_j + \mathbf{q}_i^\top \mathbf{a}_{\text{clip}(i-j)}}{\sqrt{d}} \\ \text{clip}(m) &= \max(-M, \min(M, m)) \ . \end{aligned} \tag{3}$$

And for the context vector $\mathbf{c}_i$, instead of calculating it as the weighted average of the value vectors:

$$\mathbf{c}_i = \sum_j \alpha_{i,j} \mathbf{v}_j \tag{4}$$

where $\boldsymbol{\alpha}_i = \text{softmax}(\mathbf{s}_i)$, the relative positional encoding injects the positional vector $\mathbf{b}_{\text{clip}(i-j)}$ into the calculation of $\mathbf{c}_i$ to give

$$\mathbf{c}_i = \sum_j \alpha_{i,j} (\mathbf{v}_j + \mathbf{b}_{\text{clip}(i-j)}) \ . \tag{5}$$

The relative positional encoding is more suitable for video tasks as the input sequence is continuous in time and consecutive frames are more correlated.

3.4 Alignment Model

The contextual model produces a spatio-temporal feature vector sequence with T time steps. To align the feature vector sequence to the target label sequence, we propose the stochastic fine-grained labeling (SFL) mechanism to enhance the supervision along the temporal domain based on the CTC method.

Connectionist Temporal Classification (CTC)

CTC introduces a sequence of hidden variables $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_T), \pi_t \in \mathcal{V} \cup \{blank\}$, where $\mathcal{V}$ is the vocabulary and *blank* is a special token for representing silent time steps and separating consecutive repeating glosses. The hidden state π_t indicates the alignment between the input time step t and the corresponding gloss in the target sentence. Given the input sequence $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_T)$ and the target sequence $\mathbf{y} = (y_1, \ldots, y_L)$, consider the conditional probability:

$$
\begin{aligned}
p(\mathbf{y}|\mathbf{x}) &= \sum_{\boldsymbol{\pi}} p(\mathbf{y}|\boldsymbol{\pi}, \mathbf{x}) p(\boldsymbol{\pi}|\mathbf{x}) \\
&= \sum_{\pi \in \mathcal{B}^{-1}(\mathbf{y})} p(\boldsymbol{\pi}|\mathbf{x}) \\
&\approx \sum_{\pi \in \mathcal{B}^{-1}(\mathbf{y})} \prod_{t=1}^{T} p(\pi_t|\mathbf{x}) \ ,
\end{aligned}
\tag{6}
$$

where $\mathcal{B} : (\mathcal{V} \cup \{blank\})^T \rightarrow \mathcal{V}^L$ is a function that maps a hidden sequence to its corresponding sequence of glosses. More specifically, $\mathcal{B}$ converts the hidden sequence to the gloss sequence by first removing the consecutive repeating words and then the *blank* symbol in the hidden sequence. In CTC, the term $p(\boldsymbol{\pi}|\mathbf{x})$ is approximated by $\prod_{t=1}^{T} p(\pi_t|\mathbf{x})$ as the hidden variables are assumed to be independent given the input $\mathbf{x}$. The CTC loss is defined as

$$
\mathcal{L}_{\mathrm{CTC}}(\mathbf{x}, \mathbf{y}) = -\log p(\mathbf{y}|\mathbf{x}) \ . \tag{7}
$$

During training, the CTC loss is minimized so that $p(\mathbf{y}|\mathbf{x})$ is maximized. In testing, the prefix beam search algorithm [7] is used to decode the conditional probability sequence. It retains only the k most probable prefixes at each decoding time step so as to reduces the search space and speed up decoding.

Stochastic Fine-grained Labeling

HMM-based CSLR methods have exploited multiple hidden states to represent each gloss to increase the label granularity and improve recognition performance [14, 11]. As one sign gloss usually consists of multiple motion primitives, using multiple states instead of one single state for one gloss helps the network learn more discriminative features at different time steps of one gloss.

Given an input video $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_T)$, the corresponding gloss sequence: $\mathbf{y} = (y_1, \ldots, y_L)$, a sub-gloss state number sequence $\mathbf{c} = (c_1, \ldots, c_L)$ and the maximal state number $c_{\max}$, the fine-grained label $\tilde{\mathbf{y}}$ is defined as an extension of the original gloss sequence:

$$
\tilde{\mathbf{y}} = \mathcal{E}(\mathbf{y}, \mathbf{c}) = (y_1^1, \ldots, y_1^{c_1}, y_2^1 \ldots, y_2^{c_2}, \ldots, y_L^1, \ldots, y_L^{c_L}) \tag{8}
$$

where $y_i^j \in \mathcal{V} \times \{1, \ldots, c_{\max}\}$ and $j \in \{1, \ldots, c_i\}$. The extension function $\mathcal{E}(\mathbf{y}, \mathbf{c})$ takes the gloss sequence $\mathbf{y}$ and the sub-gloss state number sequence $\mathbf{c}$ as inputs, and extends the gloss sequence to a sub-gloss state sequence $\tilde{\mathbf{y}}$. In HMM-based methods, the number of states is usually fixed for each gloss as there is no prior knowledge about the optimal number of states. Skip transitions are used to allow exiting a gloss earlier. To avoid extra effort of manually fine-tuning the skip/exit penalty, we propose the stochastic fine-grained labeling (SFL) method that allows the model to automatically learn the number of states for each gloss. SFL utilizes the REINFORCE algorithm [23] to estimate the distribution of the

number of states during training. By sampling different state number sequence $\mathbf{c}$ based on the target sequence, we reinforce the probability of the state number sequences that produce lower CTC loss.

The log conditional probability $\log p(\mathbf{y}|\mathbf{x})$ in CTC loss (Equation 7) can be re-written by introducing the state number sequence $\mathbf{c}$ as latent variables as follows:

$$\begin{aligned}\log p(\mathbf{y}|\mathbf{x}) &= \log \sum_{\mathbf{c}} p(\mathbf{c}|\mathbf{x}) \frac{p(\mathbf{y},\mathbf{c}|\mathbf{x})}{p(\mathbf{c}|\mathbf{x})} \\ &\geq \sum_{\mathbf{c}} p(\mathbf{c}|\mathbf{x}) \log \frac{p(\mathbf{y},\mathbf{c}|\mathbf{x})}{p(\mathbf{c}|\mathbf{x})} \quad \text{(Jensen's inequality)} \\ &= \mathbb{E}_{\mathbf{c}\sim p(\mathbf{c}|\mathbf{x})}[\log p(\mathbf{y},\mathbf{c}|\mathbf{x}) - \log p(\mathbf{c}|\mathbf{x})] \ .\end{aligned} \tag{9}$$

Instead of directly maximizing $\log p(\mathbf{y}|\mathbf{x})$, the lower bound given by Equation 9 is maximized, which can be further written as follows by taking the extended sub-gloss sequence $\tilde{\mathbf{y}} = \mathcal{E}(\mathbf{y},\mathbf{c})$ into account:

$$\mathbb{E}_{\mathbf{c}\sim p(\mathbf{c}|\mathbf{x})}[\log p(\mathbf{y},\mathbf{c}|\mathbf{x}) - \log p(\mathbf{c}|\mathbf{x})] = \mathbb{E}_{\mathbf{c}\sim p(\mathbf{c}|\mathbf{x})}[\log p(\tilde{\mathbf{y}}|\mathbf{x})] + H[\mathbf{c}] \ . \tag{10}$$

Here $H[\mathbf{c}]$ is the entropy of the state number distribution whose gradient can be calculated explicitly given the model. The gradient of the term $\mathbb{E}_{\mathbf{c}\sim p(\mathbf{c}|\mathbf{x})}[\log p(\tilde{\mathbf{y}}|\mathbf{x})]$ with respect to the network parameter θ can be written as follows after approximating it by the Monte Carlo method with the log derivative trick:

$$\begin{aligned}&\nabla_\theta \mathbb{E}_{\mathbf{c}\sim p(\mathbf{c}|\mathbf{x})}\left[\log p(\tilde{\mathbf{y}}|\mathbf{x})\right] \\ =&\mathbb{E}_{\mathbf{c}\sim p(\mathbf{c}|\mathbf{x})}\left[\nabla_\theta \log p(\tilde{\mathbf{y}}|\mathbf{x}) + \log p(\tilde{\mathbf{y}}|\mathbf{x}) \nabla_\theta \log p(\mathbf{c}|\mathbf{x})\right] \\ \approx&\frac{1}{N}\sum_{i=1}^{N}[\nabla_\theta \log p(\mathcal{E}(\mathbf{y},\mathbf{c}^{(i)})|\mathbf{x}) + \mathcal{R}(\mathbf{c}^{(i)}) \nabla_\theta \log p(\mathbf{c}^{(i)}|\mathbf{x})]\end{aligned} \tag{11}$$

where $\mathcal{R}(\mathbf{c}^{(i)}) = \log p(\mathcal{E}(\mathbf{y},\mathbf{c}^{(i)})|\mathbf{x})$.

Given $\tilde{\mathbf{y}}$ and $\mathbf{x}$, the term $\log p(\tilde{\mathbf{y}}|\mathbf{x})$ and its gradient can be directly calculated using the existing CTC method mentioned above. To reduce the variance of the Monte Carlo estimator of the gradient, similar to [24], we introduce a reward baseline term $b(\mathbf{x})$ into the gradient estimator as follows:

$$\hat{\mathcal{R}}(\mathbf{c}^{(i)}) = \log p(\mathcal{E}(\mathbf{y},\mathbf{c}^{(i)})|\mathbf{x}) - b(\mathbf{x}) \ . \tag{12}$$

Approximation Trick

Directly sampling the state number sequence $\mathbf{c}$ from $p(\mathbf{c}|\mathbf{x})$ is impractical as its length may not match the length of the gloss sequence $\mathbf{y}$, and causes a large proportion of invalid samples at the beginning of training (as $p(\mathbf{y},\mathbf{c}|\mathbf{x}) = 0$ for such cases). To tackle this problem, we further simplify the model by assuming

(1) the most probable gloss sequence dominates the others given a video, and (2) the numbers of states is only dependent on the corresponding gloss. Thus, we have

$$\begin{aligned} p(\mathbf{c}|\mathbf{x}) &= \sum_{\mathbf{y}} p(\mathbf{y}|\mathbf{x}) p(\mathbf{c}|\mathbf{x}, \mathbf{y}) \\ &\approx \max_{\mathbf{y}} p(\mathbf{y}|\mathbf{x}) \prod_{l=1}^{L} p(c_l|y_l) \ . \end{aligned} \tag{13}$$

In our implementation, $p(c_l|y_l)$ is a categorical distribution produced by the SFL module, which is a simple two-layer feed-forward neural network. It takes a gloss y_l as input and outputs the corresponding distribution state number distribution $p(c_l|y_l)$. In training, we follow the empirical distribution and set $p(\mathbf{y}|\mathbf{x}) = 1$ for the data sample $(\mathbf{x}, \mathbf{y})$ in the training dataset. As the distribution of $\mathbf{c}$ has been changed from being conditioned on $\mathbf{x}$ to $\mathbf{y}$, we also update our baseline from $b(\mathbf{x})$ to $b(\mathbf{y})$. For the baseline estimation, we choose another two-layer feed-forward network and train it with the MSE loss between $b(\mathbf{y}) = \frac{1}{L}\sum_{l=1}^{L} b(y_l)$ and the uncalibrated reward $\mathcal{R}(\mathbf{c}^{(i)})$.

4 Experiments

4.1 Dataset and Metrics

PHOENIX-2014 PHOENIX-2014 [12] is a popular German sign language dataset collected from weather forecast broadcast. The dataset contains a total of 963k frames captured by an RGB camera in 25 frames per second. It has a vocabulary size of 1081. There are 5672, 540 and 629 data samples in the training, development, and testing sets, respectively. The dataset contains 9 signers who appear in all three splits.

PHOENIX-2014-T PHOENIX-2014-T [4] is an extension to the PHOENIX-2014 dataset with different sentence boundaries. It is designed for sign language translation but can also be used to evaluate the CSLR task. The dataset has a vocabulary size of 1085. There are 7096, 519 and 642 samples in the training, development, and testing sets, respectively. Similar to the PHOENIX-2014 dataset, there are 9 signers who appear in all three splits.

Metrics For evaluation, we use word error rate (WER) as the metric, which is defined as the minimal summation of the substitution, insertion and deletion operations to convert the recognized sentence to the corresponding reference sentence:

$$\text{WER} = \frac{\#\text{ substitutions } + \#\text{ insertions } + \#\text{ deletions}}{\#\text{ glosses in reference}} \tag{14}$$

For both datasets, the evaluation script comes along with the dataset is used for computing the WER.

4.2 Basic Settings

Data Processing and Augmentation For PHOENIX-2014 and PHOENIX-2014-T dataset, we follow the commonly used setting to adopt the full-frame videos with a resolution of 210×260. The frames are first resized to 256×256 and then cropped to 224×224. During training, random cropping is used and in testing, center cropping is adopted.

Model Hyperparameters For the visual model, we adopt a 18-layer 2D ResNet pre-trained on ImageNet [18] with the fully connected layer removed. For the contextual model, we use a 2-layer transformer encoder with 4 heads ($h = 4$), model dimension $d = 512$ and position-wise feed-forward layer dimension $d_{\text{ff}} = 2048$.

Training and Decoding The model is trained with a batch size of 8 using the Adam optimizer [10] with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. We schedule the learning rate for the i-th epoch as $\eta_i = \eta_0 \cdot 0.95^{\lfloor \frac{i}{2} \rfloor}$ with initial learning rate $\eta_0 = 1 \times 10^{-4}$. Each model is trained for 30 epochs. For stochastic fine-grained labeling (SFL) based model, the number of Monte Carlo samples N is set to 32.

During training, if the length of the target sequence exceeds the number of input frames, the CTC loss of the corresponding sample will be zeroed out. If it is related to Monte Carlo sampling of $p(\mathbf{c}^{(i)}|\mathbf{y})$, the calibrated reward $\hat{\mathcal{R}}(\mathbf{c}^{(i)})$ will be set to zero. For decoding, we adopt the prefix beam search [7] algorithm. All testing results are generated using a beam width of 10. In multiple-state models, we adopt a pseudo-language model which follows the rules below:

1. Within one gloss, the transition is strictly left to right without skipping.
2. For transition between two glosses, only the transition to the first state of the destination glosses from the last state of the source gloss is allowed.

4.3 Results and Analysis

In this section, we discuss the effectiveness of our proposed methods based on the experimental results on PHOENIX-2014 dataset.

Stochastic Frame Dropping We conduct experiments with 0%, 25%, 50% and 75% frame dropping rates p_{drop}. To keep the window size M of relative position encoding consistent, we use $M = 16, 12, 8$ and 4 for $p_{\text{drop}} = 0\%$, 25%, 50% and 75%, respectively. In this stage, the number of states for all glosses is set to 1 and no gradient is stopped. Figure 2 (Left) shows the performance with different dropping rates. The results show that if the dropping rate is low

(i.e., 0% and 25%), the model tends to give worse results as there are fewer variants in the random dropping and the model tends to overfit the retained salient frames. On the other hand, if the dropping rate is too high (i.e, 75%), the model may lose too much information resulting in a slight increase of WER. We select $p_{\text{drop}} = 50\%$ as the default setting for the following experiments.

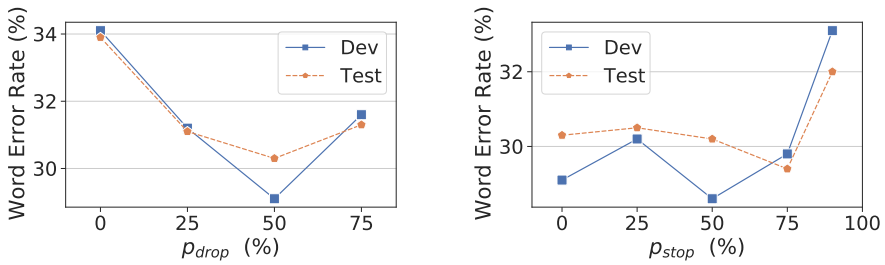

Fig. 2. WER of models trained with $p_{\text{drop}} = 0\%$, 25%, 50% and 75% (Left), and WER of models trained with $p_{\text{stop}} = 0\%$, 25%, 50%, 75% and 90% given $p_{\text{drop}} = 50\%$ (Right).

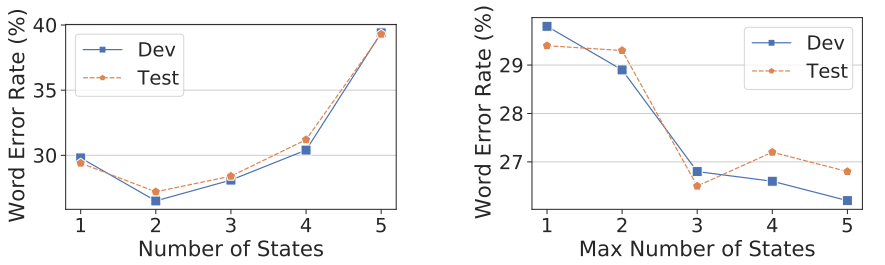

Fig. 3. Comparison between different numbers of states under deterministic fine-grained labeling (Left) and different maximal numbers of states under stochastic fine-grained labeling (Right).

Stochastic Gradient Stopping For stochastic gradient stopping, we train models with different p_{stop} based on $p_{\text{drop}} = 50\%$ setting. As shown in Figure 2, the performance is not effected too much by gradient dropping probably because the visual model has already been pre-trained on ImageNet. Among the selected p_{stop}, the optimal p_{stop} is 75% for the Dev set and 50% for the Test set. In the following experiments, $p_{\text{stop}} = 75\%$ is selected as the default setting.

Table 1. The comparison between the REINFORCE method with a simple uniform distribution over the number of states. The maximal numbers of states C are set to 5.

	Dev (%)		Test (%)	
	del/ins	WER	del/ins	WER
Uniform	18.5/2.3	30.2	19.3/2.0	30.3
REINFORCE	7.9/6.5	26.2	7.5/6.3	26.8

Deterministic Fine-grained Labeling We first test our model on a deterministic fine-grained labeling (DFL) setting where the numbers of states for all glosses are fixed to a constant C. The distribution of the number of states c_l for the l-th gloss y_l is set as:

$$p(c_l|y_l) = \begin{cases} 1, c_l = C \\ 0, \text{otherwise} \end{cases} \tag{15}$$

As shown in Figure 3 (Left), when the number of states is set to $C = 2$, the WER drops to 26.5% on the Dev set and 27.2% on the Test set. The performance improves from the baseline ($C = 1$) when the number of states is 2 or 3, but worsens as the number of states further goes up. This is reasonable as the average number of frames per gloss for the PHOENIX-2014 training dataset is 12.2. With $p_{\text{drop}} = 50\%$, half of the frames are stochastically dropped and the average number of frames per gloss becomes 6.1. When the number of states is large (e.g., $C = 4, 5$), the shorter videos that contain more glosses will violate the CTC length constraint and hence discarded during training.

Stochastic Fine-grained Labeling We test different maximal numbers of states for the stochastic fine-grained labeling (SFL) method in Figure 3 (Right). The WER starts to decrease as the maximal number of states exceeds 2. To further show the necessity of the REINFORCE algorithm, we compare the the REINFORCE method with a model trained with a uniform distribution over the number of states, i.e., $p(c_l|y_l) = \frac{1}{C}$. As shown in Table 1, the performance of the model trained with uniformly distributed number of states is much worse than that of the model trained with the REINFORCE algorithm.

Qualitative Results Figure 4 shows the posteriors of sub-gloss states produced by our model. It can be seen that our proposed SFL method prefers to assign glosses with 2 or 4 states, and each state lasts for $1 \sim 3$ frames under the 50% frame dropping rate.

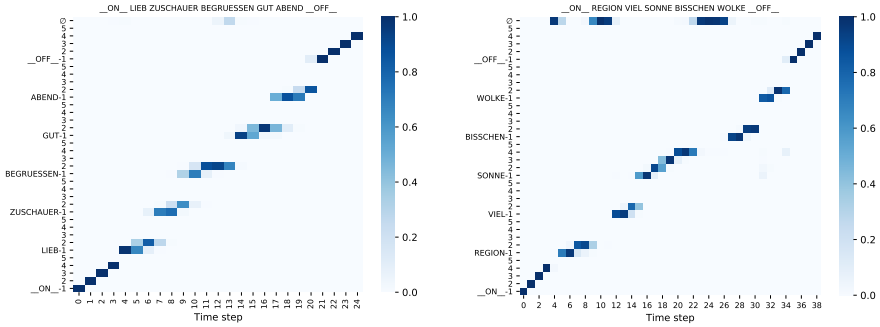

Fig. 4. Examples of sub-glosses posteriors. Left: `__ON__ LIEB ZUSCHAUER BEGRUESSEN GUT ABEND __OFF__`. Right: `__ON__ REGION VIEL SONNE BISSCHEN WOLKE __OFF__`. The number on the y-axis indicates the i-th state for particular glosses and $\varnothing$ indicates blank frame.

Table 2. The comparison with state-of-the-art works on the PHOENIX-2014 dataset. Baseline stands for our network architecture without SFD, SGS, and SFD. LM stands for the use of the language model provided by the PHOENIX-2014 dataset during decoding. Some notes on experimental setting: For SFD, $p_{\text{drop}} = 0.5$; for SGS, $p_{\text{stop}} = 0.75$; for DFL, $C = 2$; for SFL, $C_{max} = 5$.

Method	Dev (%)		Test (%)	
	del/ins	WER	del/ins	WER
Deep Sign[13]	-	38.3	-	38.8
Re-sign [14]	-	27.1	-	26.8
SubUNets [2]	-	40.8	-	40.7
Staged-Opt[5]	-	39.4	-	38.7
LS-HAN [9]	-	-	-	38.3
Align-iOpt [17]	12.9/2.6	37.1	13.0/2.5	36.7
SF-Net [25]	-	35.6	-	34.9
DPD+TEM [27]	9.5/3.2	35.6	9.3/3.1	34.5
CNN-LSTM-HMM [11]	-	26.0	-	26.0
Baseline	9.0/9.3	34.1	8.9/9.2	33.9
SFD	10.1/6.4	29.1	9.9/6.6	30.3
SFD+SGS	9.9/6.9	29.8	9.3/6.6	29.4
SFD+SGS+DFL	8.0/6.5	26.5	8.1/6.3	27.2
SFD+SGS+SFL	7.9/6.5	26.2	7.5/6.3	26.8
SFD+SGS+SFL+LM	10.3/4.1	**24.9**	10.4/3.6	**25.3**

Table 3. The comparison with state-of-the-art works on the PHOENIX-2014-T dataset. Our SFD+SGS+SFL model is trained with only the full-frame stream and gloss annotation.

Method	Annotation				WER (%)	
	Gloss	Mouth	Hand	Text	Dev	Test
CNN-LSTM-HMM (1-Stream) [11]	✓				24.5	26.5
CNN-LSTM-HMM (2-Stream) [11]	✓	✓			24.5	25.4
CNN-LSTM-HMM (3-Stream) [11]	✓	✓	✓		22.1	24.1
SLT (Gloss) [3]	✓				24.9	24.6
SLT (Gloss+Text) [3]	✓			✓	24.6	24.5
SFD+SGS+SFL	✓				25.1	26.1

4.4 Comparison with State-of-the-arts

In this section, we compare our method with other state-of-the-art (SOTA) CSLR methods on the two datasets mentioned in Section 4.1.

PHONIEX-2014 Table 2 shows that our final result with the use of language model outperforms the best SOTA result on the PHOENIX-2014 dataset by 1.1% and 0.7% on the Dev and Test set, respectively.

PHONIEX-2014-T Table 3 shows our result compared to other SOTA results on the PHOENIX-2014-T dataset. Our model achieves comparable results among the models trained with gloss annotation only.

5 Conclusions

In this paper, we propose stochastic modeling of various components of a continuous sign language recognition architecture. We use ResNet18 as the visual model, transformer encoder as the contextual model and a stochastic fine-grained labeling version of connectionist temporal classification (CTC) as the alignment model. We addressed the issue of unsatisfactory performance when training the CNN-transformer model with CTC loss in an end-to-end manner by introducing the SFD, SGS, and further improve the model performance by introducing SFL. Our model outperforms the state-of-the-art results by 1.1%/0.7% on the dev set and the test set of the PHOENIX-2014 dataset, and achieves competitive results on the PHOENIX-2014-T dataset.

Acknowledgements

This work was supported by the Research Grants Council of the Hong Kong Special Administrative Region, China (Project Nos. HKUST16200118 and T45-407/19N-1).

References

1. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473 (2014)
2. Camgoz, N.C., Hadfield, S., Koller, O., Bowden, R.: SubUNets: End-to-end hand shape and continuous sign language recognition. In: 2017 IEEE International Conference on Computer Vision (ICCV). pp. 3075–3084. IEEE (2017)
3. Camgoz, N.C., Koller, O., Hadfield, S., Bowden, R.: Sign language transformers: Joint end-to-end sign language recognition and translation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10023–10033 (2020)
4. Cihan Camgoz, N., Hadfield, S., Koller, O., Ney, H., Bowden, R.: Neural sign language translation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 7784–7793 (2018)
5. Cui, R., Liu, H., Zhang, C.: Recurrent convolutional neural networks for continuous sign language recognition by staged optimization. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 7361–7369 (2017)
6. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
7. Hannun, A.Y., Maas, A.L., Jurafsky, D., Ng, A.Y.: First-pass large vocabulary continuous speech recognition using bi-directional recurrent DNNs. arXiv preprint arXiv:1408.2873 (2014)
8. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
9. Huang, J., Zhou, W., Zhang, Q., Li, H., Li, W.: Video-based sign language recognition without temporal segmentation. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
10. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
11. Koller, O., Camgoz, C., Ney, H., Bowden, R.: Weakly supervised learning with multi-stream CNN-LSTM-HMMs to discover sequential parallelism in sign language videos. IEEE transactions on pattern analysis and machine intelligence (2019)
12. Koller, O., Forster, J., Ney, H.: Continuous sign language recognition: Towards large vocabulary statistical recognition systems handling multiple signers. Computer Vision and Image Understanding **141**, 108–125 (2015)
13. Koller, O., Zargaran, O., Ney, H., Bowden, R.: Deep sign: hybrid CNN-HMM for continuous sign language recognition. In: Proceedings of the British Machine Vision Conference 2016 (2016)
14. Koller, O., Zargaran, S., Ney, H.: Re-Sign: Re-aligned end-to-end sequence modelling with deep recurrent CNN-HMMs. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4297–4305 (2017)
15. Liu, Z., Qi, X., Pang, L.: Self-boosted gesture interactive system with ST-Net. In: 2018 ACM Multimedia Conference on Multimedia Conference. pp. 145–153. ACM (2018)
16. Pham, N.Q., Nguyen, T.S., Niehues, J., Muller, M., Waibel, A.: Very deep self-attention networks for end-to-end speech recognition. arXiv preprint arXiv:1904.13377 (2019)

17. Pu, J., Zhou, W., Li, H.: Iterative alignment network for continuous sign language recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4165–4174 (2019)
18. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: ImageNet large scale visual recognition challenge. International journal of computer vision **115**(3), 211–252 (2015)
19. Shaw, P., Uszkoreit, J., Vaswani, A.: Self-attention with relative position representations. arXiv preprint arXiv:1803.02155 (2018)
20. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
21. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1–9 (2015)
22. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Advances in neural information processing systems. pp. 5998–6008 (2017)
23. Williams, R.J.: Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine learning **8**(3-4), 229–256 (1992)
24. Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R., Bengio, Y.: Show, attend and tell: Neural image caption generation with visual attention. In: International conference on machine learning. pp. 2048–2057 (2015)
25. Yang, Z., Shi, Z., Shen, X., Tai, Y.W.: SF-Net: Structured feature network for continuous sign language recognition. arXiv preprint arXiv:1908.01341 (2019)
26. Zhang, Z., Pu, J., Zhuang, L., Zhou, W., Li, H.: Continuous sign language recognition via reinforcement learning. In: International Conference on Image Processing (ICIP). pp. 285–289 (2019)
27. Zhou, H., Zhou, W., Li, H.: Dynamic pseudo label decoding for continuous sign language recognition. In: 2019 IEEE International Conference on Multimedia and Expo (ICME). pp. 1282–1287. IEEE (2019)