Pseudo-Convolutional Policy Gradient for Sequence-to-Sequence Lip-Reading

Anonymous FG2020 submission
Paper ID ****

***Abstract*— Lip-reading has always been a very challenging task due to the large gap between the very limited effective area of lips and the great diversity of words we can say. It is a typical sequence-to-sequence (seq2seq) problem which translates the input image sequence of lip movements to the text sequence of the speech content. However, the traditional learning process of the seq2seq models suffers two problems when used for the lip-reading task, which are the exposure bias resulted by the strategy of "teacher-forcing" and the inconsistency between the optimization target (usually the cross-entropy loss) and the final evaluation metric (usually the word error rate). In this paper, we introduce reinforcement learning (RL) to address these two problems for lip-reading. Specifically, we propose a new method called pseudo-convolutional policy gradient (PCPG) which can make good use of both the history and the future reward and widen the receptive field of the reward at each time-step to improve the efficiency in the training process and the performance of the model. Finally, we evaluate our method on three word-level and sentence-level datasets respectively. The results show a new state-of-the-art performance or competitive accuracy on the three benchmarks, which clearly demonstrate the effectiveness of our approach.**

I. INTRODUCTION

Lip-reading is one appealing manner for intelligent human-computer interaction and is receiving attention in recent years. Lip-reading aims to understand the speech content using the visual information [4] like the lip movements and so is robust to the ubiquitous audio noise, making it important to play as the complement role of the audio-based speech recognition (ASR) systems, especially in a noisy environment. Besides being a powerful support for ASR systems, there are also many other potential applications of lip-reading, such as transcribing and re-dubbing archival silent films, resolving multi-talker simultaneous speech, liveness verification and so on [8]. Benefiting from the vigorous development of deep learning (DL) and the emergence of large-scale lip-reading datasets GRID [11], LRW [4], LRW-1000 [43], LRS [8] etc, lip-reading has achieved great progresses these two years. For example, there are three main ways to solve this lip-reading problem. One is based on classification which regards lip-reading as a classifying task, the other one is based on decoding which is done with an encoder-decoder with attention system. The last one is based on connectionist temporal classification. In this paper, we take the task as a sequence-to-sequence (seq2seq) problem, which translates the lip movement sequence to a character sequence, as shown in Fig. 1. And lip-reading has a high similarity with other seq2seq tasks such as speech recognition [29], machine translation [22], image caption [6], video caption [37], and so on. Some seq2seq models especially encoder-decoder with attention based on RNN have been proved to be effective for these seq2seq tasks. These seq2seq models can model a strong temporal relation for the lip movement sequence and the character sequence. So the sentence or the word is also a character-level sequence. And in this way, the different lip-reading datasets can be trained with a common model.

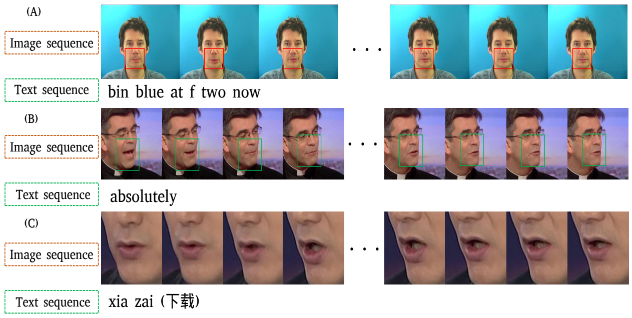

Fig. 1: Three different lip-reading samples. (A) is a lip-reading sample form GRID dataset. (B) is a lip-reading sample form LRW dataset. (C) is a lip-reading sample from LRW-1000 dataset.

However, there are two main drawbacks when using the traditional seq2seq models for lip-reading. The first one is the exposure bias, which is very common in most current seq2seq models. Most current seq2seq models are learned with a heavy dependence on the ground-truth words of each time-step to achieve an accurate prediction of the next time-step. This approach is called "teacher-forcing [28] and is used widely for natural language processing, machine translation, and also the early work of lip-reading [9]. Despite the fast convergence speed, the model has to make predictions depending on the previous prediction, not the ground-truth word, at the test time [7]. This is totally different with the input in the models learning process. This discrepancy between the learning process and the actual test process would inevitably yield inaccuracies and even errors, which would accumulate quickly and grow along the sequence. For example, when the condition is totally "teacher-forcing", in this case, the teacher-forcing rate is equal to 1, we used the ground truth as the input when decoding at every time-step. The second problem of most existing seq2seq models is the gap between the optimized target and the final non-differentiable evaluation metrics. Cross-entropy minimization is an usual optimization target in seq2seq models, and

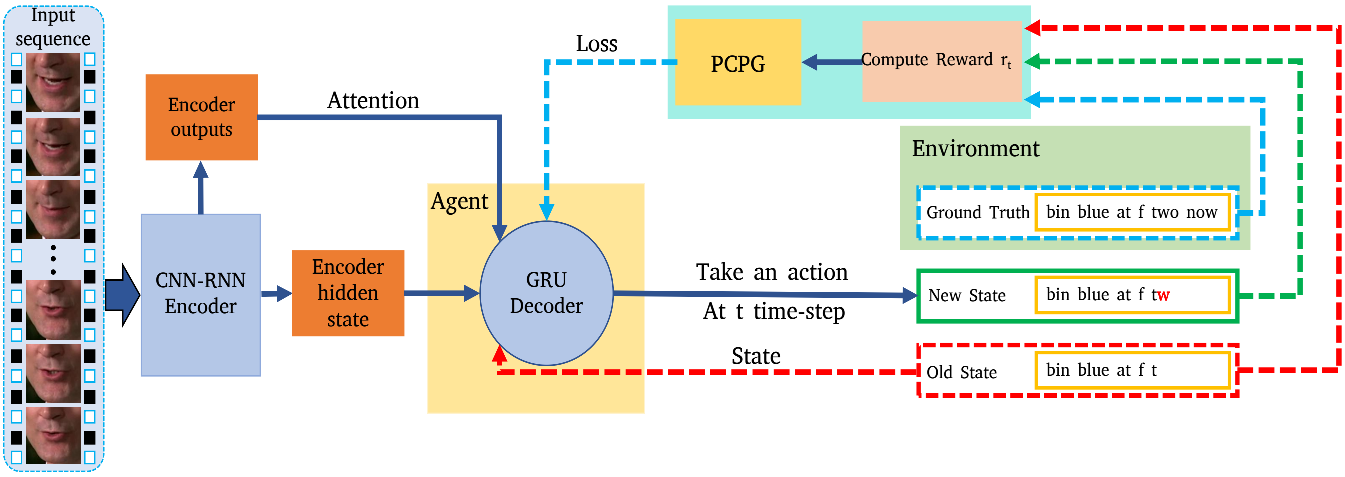

Fig. 2: Overview of our seq2seq model with PCPG for lip-reading. The GRU decoder is regarded as agent. And the ground truth is regarded as environment. In the training process, the agent observes the old state and then takes an action at each time-step. The agent reaches a new state and the environment will compute a reward for the action. Finally, the reward is processed by our PCPG and the loss is passed to the agent.

always used to ensure each individual correct predictions at each time-step [4] but with little consideration of the context words at each time-step. The evaluation metrics for seq2seq tasks are always WER (word error rate), CER (character error rate), PER (phoneme error rate) and so on. Most of these metrics are based on an consideration of the whole sequence, which are often discrete and non-differentiable, and so they are not quite fit in with the cross-entropy optimization target.

To solve the above two problems when introducing seq2seq models for the lip-reading task, we propose a new reinforcement learning method called pseudo convolutional policy gradient (PCPG) to take the context at each time-step into consideration. On the one hand, when we take RL to lip-reading, we can optimize the evaluation metrics directly to avoid the mismatch between the optimized goal in the training process and the testing process. On the other hand, in our RL training process, we consider more rewards from the history and future at each time-step. By a thorough evaluation and comparison on several lip-reading benchmarks, we demonstrate both the effectiveness and the generalization ability of the proposed PCPG based seq2eq model. At the same time, we report a new state-of-the-art performance on two benchmarks and also give a competitive accuracy on another dataset.

II. Related work

A. Lip reading

Lip reading has made a great improvement since the emergence of deep learning (DL). Many methods have been proposed based on DL such as [10], [8], [9], [4], [25], [3], [2], [45], [43], [14]. The prior work for lip-reading can be divided into two strands.

The one is based on word-level. For example, the work in [9] proposed a large-scale word-level lip-reading dataset called LRW collected from TV broadcasts. And at the same time, the authors developed CNN architectures for lip-reading. Moreover, the work in [32], the authors proposed an end-to-end deep learning architecture which is a combination of spatiotemporal convolutional, residual and bidirectional Long Short-Term Memory networks. Another work in [25] presented an end-to-end audiovisual model based on residual networks and Bidirectional Gated Recurrent Units (BGRUs). And this model consisted of two streams, one for each modality, which extract features directly from mouth regions and raw waveforms. In [9], the author used multi-view datasets for lip-reading. There are some works which belong to word-level recognition are based on a softmax classifier.

The other one is based on sentence-level. The lip-reading based on sentence-level, which required us to translate one long lip-reading video to a corresponding sequence text, is a very challenging task compared to word-level. And a lot of good work done in this level. Large scale datasets for sentence-level lip-reading are available such as LRS2 [8]. For example, the work in [3] presented a model called LipNet which uses a spatiotemporal CNN and Bi-GRU network and CTC to compute the probability of a sequence by marginalizing over all sequences that are defined as equivalent to this sequence [3]. And the work in [2] used CTC and beam search [41] for lip-reading. The other work is based on the LSTM based encoder-decoder architecture with attention, such in [8], where the model can combine the audio and visual input streams. The attention mechanism [42], [36], [15], [23], which is very popular recently, has become a necessary part of seq2seq models. The work in [2], the authors showed three architectures: a recurrent model based on LSTMs, a fully convolutional model and the recently popular transformer model. And the authors got good performance based on the transformer model.

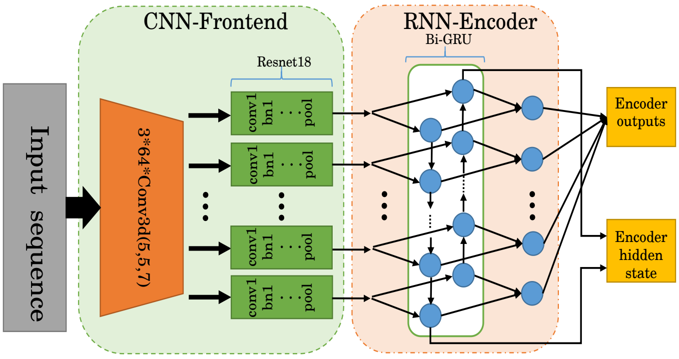

Fig. 3: The video encoder is consisted of Conv3d networks, Resnet18 networks and Bi-GRU networks. The lip sequence images is mapped to encoder output and encoder hidden state.

B. *Seq2Seq models*

Sequence-to-Sequence (Seq2Seq) models, which mainly refer the models consisted of encoder and decoder based on RNN, has been more and more popular in many sequence tasks with the development of deep learning (DL) and the presence of large scale datasets these days. The encoder receives the sequence input (such as images, audios, texts) and output a single vector. The decoder receives the single vector from the encoder and produces a sequence. The seq2seq models needn't consider the length and order of the sequence. Due to these advantages, the seq2seq models have been used in many tasks and there are many works based on seq2seq models such as [33], [37], [26], [20] and so on.

However, the usual seq2seq models still face two main problems: the exposure bias and the mismatch metrics. The exposure bias leads to error accumulation during the testing process. For example, in the training process, the seq2seq model uses the ground truth words (or other modal data) as context while at inference the entire sequence is generated by the resulting model on its own and hence the previous words generated by the model are fed as context [44]. As a result, the predicted words at training and inference are drawn from different distributions, namely, from the data distribution as opposed to the model distribution. Moreover, Most of seq2seq models are trained with the cross entropy while evaluated based on the entire generated sequence with discrete and non-differentiable metrics such as BLEU [21], ROUGE [18], METEOR [5], CIDEr [34], WER (word error rate), CER (character error rate), SER (sentence error rate) etc. Even though we get a very small cross entropy loss during training, we may not get a good performance at inference. These two problems hinder the performance of seq2seq models.

In recent years, to address the above two main problems, reinforcement learning (RL) methods are used in seq2seq models. On the one hand, in RL, the model generates the current target based on the previous predicting characters instead of the previous ground-truth characters in the training process. In this way, the model can learn a stronger relationship between characters in output character sequence than using "teacher-forcing" and avoid the problem called the mismatch metrics. On the other hand, in RL, we can optimize directly the evaluation metrics according the reward system in the training process. Most of tasks get a much better performance with RL methods, such as [35], [28], [7], [30], [39], [27], [24], [17], [19]. For example, the work in [35] proposed an alternative strategy for training a seq2seq ASR by RL methods. The other work in [28], based on an image caption task, proposed an optimization approach that called self-critical sequence training (SCST) and got a much better result than other ways. And specifically in [7], Ranzato et al. used the REINFORCE algorithm [40] to directly optimize non-differentiable evaluation metrics and overcome the exposure bias in different tasks (summarization, translation, image caption). However, there is no attempt to apply RL to lip-reading. And we still face some problems that how to enhance contextual relevance to get a much better generalization ability of seq2seq model, the instability and the inefficiency in the training process with RL.

III. The Proposed work

In this section, we introduce the proposed PCPG (Pseudo-Convolutional Policy Gradient) based seq2seq for lip-reading in detail. Specifically, we give the details from three aspects. Firstly, we describe the general model architecture in the first subsection. Then we introduce the proposed PCPG algorithm in the next subsection. Finally, we give the reward function together with the training loss in the third subsection.

A. *The General Model architecture*

As shown in Fig 2 and Fig 4, we model the lip-reading task as a sequence-to-sequence model. The input video sequences would be decoded by the video encoder (as shown in Fig 3) into encode outputs and encoder hidden state. In the training process, we introduce the PCPG to minimize the convolutional reward-based loss. Finally, each time-step would output a character-based prediction probability, which would be used to compose and obtain the final predicted sentence or word. In the following, we would introduce the video encoder and RNN-based decoder in details.

The CNN and RNN based Encoder: As shown in Fig 3, there are two main parts in our encoder: the CNN based frontend to encode the short-term spatial-temporal patterns, and the RNN based back-end to fuse the short-term patterns and encode the long-term global spatial-temporal patterns. In our implementation, we use a 3D Convolutional layer together with the Resnet-18 [13] as the CNN based fronted and a two-layer Bi-GRU as the back-end encoder. Finally, the encoder's output $\mathbf{o}^v = (o_1^v, o_2^v, \dots, o_m^v)$ and hidden-state vector h_e^v are used to record the patterns of the input video $\mathbf{x}^v = \left(x_1^v, x_2^v, \dots, x_k^v\right)$, which is defined as:

$$h_e^v, \mathbf{o}^v = Encoder\_CNN\_RNN(\mathbf{x}^v). \tag{1}$$

The PCPG based RNN Decoder: Given the representation vectors of each input sequence, a two-layer RNN is

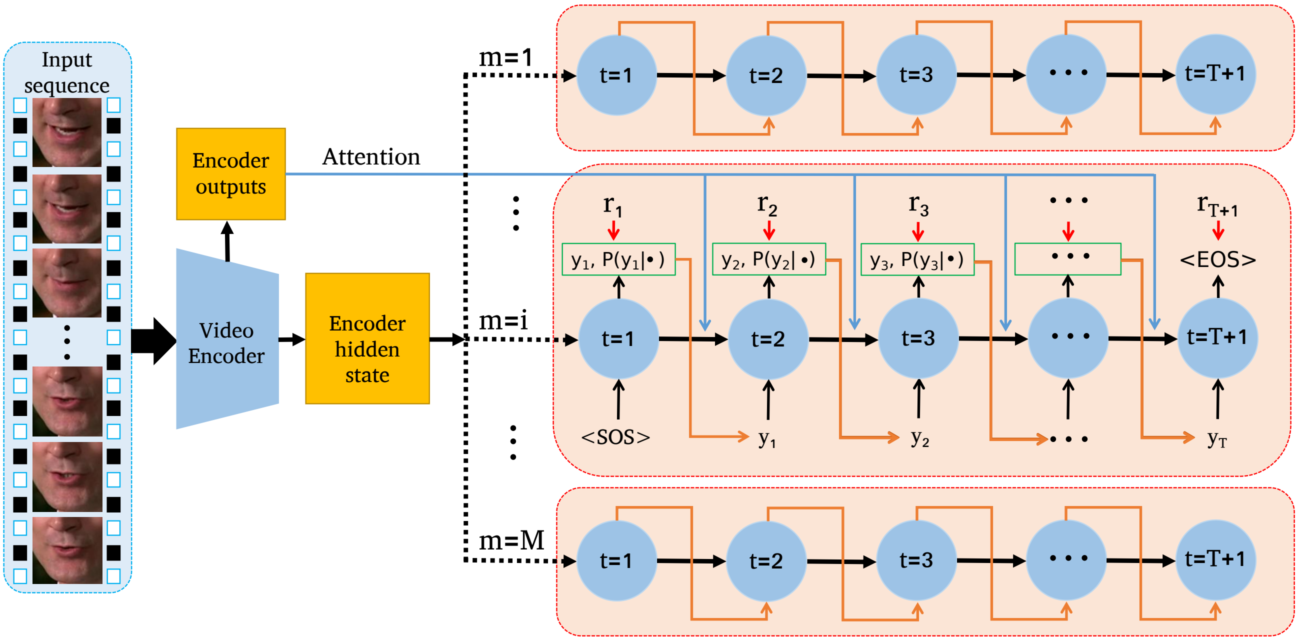

Fig. 4: Our seq2seq lip-reading decoding process with RL. The input is sequence frames and the output is sequence characters. The seq2seq model generates the target character by character. The previous output will feedback to the agent as the next input. For example, the model generates (‘a’,‘b’,‘o’,‘u’,‘t’) successively when the ground truth is the word “about”. In RL, we utilize Monte Carlo sampling to sample M transcription sequences from our model to calculate the policy gradient.

followed to predict the character at each time-step, where GRU is used as the RNN unit. The PCPG based RNN decoder is shown as Fig 4. To learn the long-term dependencies and get some key information for a more comprehensive learning sequence, we introduce the attention mechanism into the decoding process, which takes advantage of the context information in the sequence to aid for the decoding of each time step. With the decoder GRU’s hidden state $h^d_{j-1}(j = 1,2,...,n)$, we can compute the attention weight a_j of each input time-step $i(i = 1,2,...,M)$ to the current time-step j and the corresponding output y_j at time-step j as

$$\begin{aligned} e_{ji} &= attention\left(h^d_{j-1}, o^v_i\right) (i <= m), \\ \alpha_{ji} &= \tfrac{e_{ji}}{\sum_i e_{ji}}; \quad a_j = \textstyle\sum_i \alpha_{ji} o^v_i, \\ y_j &= GRU\left(h^{\mathrm{d}}_{j-1}, y^d_j, a_j\right). \end{aligned} \tag{2}$$

In this paper, we introduce a new reinforcement learning (RL) algorithm called PCPG for lip-reading, especially for the decoder’s process. The PCPG improves the performance of the model by considering the history and future reward while general RL algorithms just only consider the future reward. And we will introduce the PCPG algorithm in details in the following sections. In traditional optimization methods, we often train our models by minimizing the cross-entropy(CE) loss. The cross-entropy loss L_{CE} is computed as following:

$$L_{CE} = -\log p\left(c_1, c_2, \dots, c_n\right)$$

$$= -\log \prod_{t=1}^{n} p\left(c_t | c_1, c_2, \dots, c_{t-1}\right)$$

$$= -\sum_{t=1}^{n} \log p\left(c_t | c_1, c_2, \dots, c_{t-1}\right). \tag{3}$$

where c_i is the class label index of each character.

B. The Seq2Seq model with Policy Gradient

In this work, we use policy gradient (one kind of reinforcement learning) for lip-reading. As shown in Fig 2, we view our seq2seq model as an ’agent’ that interacts with an external ’environment’ (video frames and words or sentences). The parameters of the model, denoted as θ, can be viewed as a policy p_θ leading to an ’action’ (choosing a character to output). At each time step j, the agent will get new internal ’state’ (the attention a_j, the previous hidden state h_{j-1} and the generated character y_j) and an immediate reward r_j contributing to the total reward R. In this case, the training goal is to maximize the expected reward $E_y[R|p_\theta]$, and so the loss function is $L_{PG} = -E_y[R|p_\theta]$.

We update the parameters as follows if get a generated pair of transcription $(\mathbf{x}^v = \left(x^v_1, x^v_2, \dots, x^v_k\right), \mathbf{y} = \left(y_1, y_2, \dots, y_n\right))$ from the model:

$$\nabla_\theta E_{\mathbf{y}}\left[R|p_\theta\right] = \nabla_\theta \int P\left(\mathbf{y}|\mathbf{x}^v;\theta\right) R d\mathbf{y}$$

$$= E_{\mathbf{y}}\left[\nabla_\theta \log P\left(\mathbf{y}|\mathbf{x}^v;\theta\right) R\right]. \tag{5}$$

The total reward at current time step R_j can be computed with the equation $R_j = \sum_{i=j}^{n} \gamma^{i-j} r_i$ (γ is the discount factor) and the final reward for the whole sequence R can be computed with the equation $R = \sum_{j=1}^{n} R_j$. And the gradient can be computed as follows:

$$\nabla_\theta E_{\mathbf{y}}\left[R|p_\theta\right] = \nabla_\theta E_y \left[\sum_{j=1}^{n} R_j | p_\theta\right] =$$

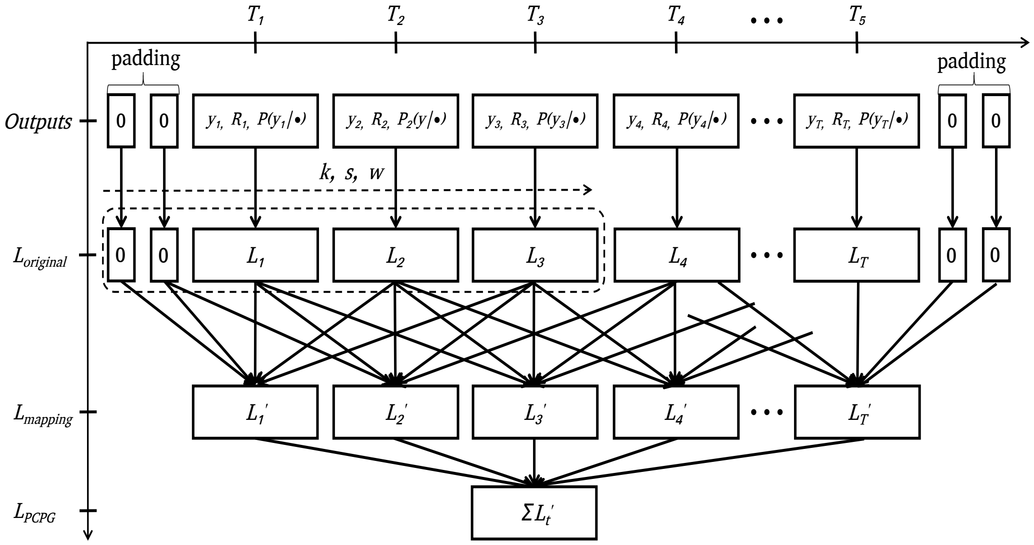

Fig. 5: Pseudo-convolutional Policy Gradient. In this figure, we set the kernel size k to 5, the stride s to 1, and the kernel weights w to $[\frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}]$.

$$E_{\mathbf{y}}\left[\sum_{j=1}^{n} R_j \nabla_\theta \log P\left(y_j|\mathbf{y}<n, \mathbf{x}^v;\theta\right)\right]. \tag{6}$$

In the real world, it is impractical to integrate all possible transcription $\mathbf{y}$ to compute the gradient of the expected reward by Eq.6. So we utilize Monte Carlo sampling to sample M transcription sequences $\mathbf{y}^{(m)}$, which is shown in Fig 4. So the gradient can be computed with Eq.7:

$$\begin{aligned}\nabla_\theta E_{\mathbf{y}}\left[R|p_\theta\right] &= E_{\mathbf{y}}\left[\sum_{j=1}^{n} R_j \nabla_\theta \log P\left(y_j|\mathbf{y}<n, \mathbf{x}^v;\theta\right)\right] \\ &\approx \frac{1}{M}\sum_{m=1}^{M}\sum_{j=1}^{n_m} R_j^m \nabla_\theta \log P\left(y_j^m|\mathbf{y}_{<j}^m, \mathbf{x}^v;\theta\right). \end{aligned} \tag{7}$$

So we can get the final gradient:

$$\frac{\partial L_{PG}(\theta)}{\partial \theta} = -\frac{1}{M}\sum_{m=1}^{M}\sum_{j=1}^{n_m} R_j^m \nabla_\theta \log P\left(y_j^m|\mathbf{y}_{<j}^m, \mathbf{x}^v;\theta\right). \tag{8}$$

Reward function: In lip-reading tasks, the performance of the model is evaluated on CER and WER, both of which are usually obtained by the edit-distance or Levenshtein distance algorithm. Here, to make it easier to give a reward for each decoding action, we choose the negative CER as the immediate reward at each time step. The reward function is defined as follows:

$$r_j = \begin{cases} -\left(ED\left(\mathbf{y}_{1:j}, \mathbf{c}\right) - ED\left(\mathbf{y}_{1:j-1}, \mathbf{c}\right)\right) & \text{if } j > 1 \\ -\left(ED\left(\mathbf{y}_{1:j}, \mathbf{c}\right) - |\mathbf{c}|\right) & \text{if } j = 1 \end{cases} \tag{9}$$

where $ED(\cdot,\cdot)$ refers to CER, which is computed by edit-distance algorithm. $\mathbf{y}_{1:j} = \{y_1, y_2, \ldots, y_n\}$ refers to the predicted character sequence. $\mathbf{c}$ refers to the ground truth and $|\mathbf{c}|$ is the ground-truth length.

For example, as shown in Fig 2, we refer ('bin blue at f two now') as the ground-truth $\mathbf{c}$, ('bin blue at f t') as the old state (or the previous decoding sequence), $\mathbf{y}_{j-1}$. The model observes the old state and then takes an action (choosing a character 'w') and gets a new state $\mathbf{y}_j$, ('bin blue at f tw').

C. Pseudo-Convolutional Policy Gradient (PCPG)

As well as known, the output in character-level lip-reading is expected to be exactly the same to all the words in the sequence of the ground truth. The output character y_j at each time step depends not only on the encoding power of the model but also on the context to obtain correct prediction. In RL, the future expected reward at each time-step from the environment can be regarded as the context information. In fact, policy gradient is an algorithm based on turn updates. We can consider the history reward and the future reward at the same time. Moreover, in lip-reading, the character text output at each time step relies on context very much. To take the context of y_j at each time-step into consideration, we mimic the manner of the successful convolutional operations in traditional deep models, which would gradually increase the receptive field of input at each layer to take the context of each position into consideration. Specifically, we introduce pseudo-convolutional operations to the immediate loss sequence $\{L_1, L_2, \ldots, L_T\}$ before computing the loss. In this way, the model will consider the context when updating the model's parameters. Here, our pseudo-convolutional has many common places with the temporal convolutional operations in DL.

Local perception: As we can see in Fig 4, the k refers to our kernel size in PCPG. On the one hand, the existence of the kernel size can widen the reward receptive field and the decoder can have a local perception at each time-step. On the other hand, this local perception makes the relationship between adjacent rewards much closer. The reward shows how well the decoding output at every time-step. So we can get much context information for decoding by this local perception.

Weight sharing: In our PCPG, the kernel weight w is the same in different local region. It is said that the model learn same feature in all local region by sharing weight. This weight sharing is the same as the proposed temporal convolutional networks (TCN). At the same time, in this paper, we just use some limited value for kernel weight w by manual setup.

As the same to the traditional convolutional operations, we introduce the pseudo convolutional kernel into model, with size k, stride s and weight w like TCN. Here, we take one of the M transcription sequences as an example to illustrate the process, as shown in Fig 4. In a decoding turn, we first get the decoding result sequence $\{(y_i, R_i, P(y_i|\cdot)), i = 1, 2, \ldots, T\}$ by the *GRU* decoding unit. We pad the decoding result sequence with zeros at first and last time-step respectively to adapt the size k. According to the reinforcement learning algorithm, the immediate loss at each time step L_t which is computed by the following equation:

$$L_t = L_{original} = -R_t \cdot logP(y_t|\cdot;\theta). \tag{10}$$

However, this manner has not taken the context into consideration. We introduce the pseudo-convolutional context into the consideration in the optimizing process with the following equation:

$$L_t^{'} = L_{mapping} = L_{t-|k/2|:t+|k/2|} \cdot w = \sum_{i=t-|k/2|}^{t+|k/2|} w_i \cdot L_i. \tag{11}$$

The $|k/2|$ (above or below) takes integer. Finally, the total PCPG loss is denoted as

$$L_{PCPG} = \sum_{t=1}^{T} L_t^{'} = \sum_{t=1}^{T} w_i \cdot \sum_{i=t-2}^{t+2} L_i, \tag{12}$$

To update the parameters, we define $\nabla_\theta L_{PCPG}^t$ as the local gradient at time-step t and $\frac{\partial L_{PCPG}}{\partial \theta}$ as the final gradient. And they can be computed as

$$\frac{\partial L_{PCPG}}{\partial \theta} = \sum_{t=1}^{T} \frac{\partial L_{PCPG}}{\partial L_t^{'}} \frac{\partial L_t^{'}}{\partial \theta}$$

$$= w_i \cdot (\sum_{t=1}^{T} \frac{\partial L_{PCPG}}{\partial L_t^{'}} \cdot (\sum_{i=t-|k/2|}^{t+|k/2|} \frac{\partial L_t^{'}}{\partial L_i} \frac{\partial L_i}{\partial \theta})). \tag{13}$$

According to the above equations and the Fig 5, we can observe that the PCPG has a bigger optimized receptive field than the usual REINFORCE algorithm at time-step t. And this property could guide the model to obtain a more robust performance by considering the context in the optimizing process.

D. Compared with traditional Policy Gradient

Comparing with the traditional policy gradient, the PCPG has bigger optimized receptive field and more favorable context loss constraints.

On the one hand, the PCPG considers a bigger optimized receptive field than the traditional gradient (PG). In traditional PG, there is no optimized receptive field and the loss function couldn't get make use of the context information. However, if we consider lip-reading as a linguistic task, the context will be very important. And the parameters in model is based on a round with PG, which allows us to naturally focus on contextual loss information when optimizing a step of policy parameters. The PCPG can enable our model to establish stronger semantic relationships when optimizing.

On the other hand, we know it is usually unstable for the models to train with RL. There will be a big gradient change due to the randomness when deciding with traditional PG algorithm. According to the Eq.13, we can find the immediate loss L_t at each time-step will be an average value based on multiple time-steps in PCPG. This mean constraint is necessary in PG and the model can find the optimal value in a smaller range. At the same time, the existence of the overlapping parts will make the local gradient value not change dramatically at an adjacent time step. For example, in Fig 5, when we set the kernel size k to 5, the stride s to 1, the overlapping parts'size is 4. So the model can obtain more favorable context loss constraints which can help the training process more stable and faster with PCPG. To ensure the value of the gradient in PCPG at a same quantitative level with the traditional PG, we set

$$\sum_{i=1}^{k} w_i = 1. \tag{14}$$

(where w_i is kernel weight.)

IV. Experiments

In this section, we evaluate our method on both the word-level and sentence-level lip-reading datasets. At the same time, we also discuss the effects of the convolutional kernel's hyperparameters k, w, s through a thorough ablation study. By comparing with several other related work, a clear demonstration of advantages of the proposed PCPG is shown for word-level and sentence-level lip-reading.

A. Datasets

We evaluate on three datasets in total, including the sentence-level benchmark, GRID and the large-scale word-level dataset, LRW, LRW-1000.

GRID [11], released in 2006, is a widely used sentence-level benchmark for lip-reading methods. There are 33 speakers and each speak out 1000 sentence, leading to about 33,000 sentence videos in total. The videos are all recorded with a fixed clear background and the speakers are required to be in the frontal view in the speaking process. This dataset is used widely as a sentence-level benchmark [3], [16], [38].

LRW [4], released in 2016, is the first large scale word-level lip-reading datasets in the wild. The videos are all collected from BBC TV broadcasts, including many different fields. We try to evaluate on this word-level dataset to explore the possibility of seq2seq models on the word-level tasks. We also have classifying experiments to evaluate the PCPG's effects for extracting features in this dataset.

LRW-1000 [43], released in 2018, is a naturally-distributed large-scale benchmark for word-level lip-reading. There are 1000 Mandarin words and more than 700 thousand samples in total. Besides a diversified range of speakers' pose, age, make-up, gender and so on, one another property of this dataset is that there is not length limit in the 1000 words, forcing the corresponding model to be robust and adaptive to words with both few and many characters. These properties make LRW-1000 very challenging for most lip-reading methods. Like LRW, we also have classifying experiments to evaluating the PCPG's effects for extracting features in this dataset.

B. Implementation details

In our experiments, all the images are normalized with respect to the overall mean and variance. When fed into models, the frames in each sequence are cropped in the same random position for training and centrally cropped for validation and test. All of the images are resized to a fixed size of 112*112 and then cropped to a size of 100*50. First, we put the image sequence to the encoder (the encoder architecture as shown in Fig 2) and get encoder outputs and encoder hidden state. We set all GRU modules to two-layer. Then the decoder (GRU unit) will receive the encoder outputs and the encoder hidden state and generates one and one character.

Our implementation is based on PyTorch and the model is trained on servers with four NVIDIA Tian X GPUs, with 12GB memory of each one. We use the Adam optimizer with an initial learning rate of 0.001. We also apply dropout with probability 0.5 for the last layer of the model. In our experiments, we try to evaluate the performance of the PCPG by the seq2seq model (shown as Fig 3) and the classifying model (shown in Fig 2). For PCPG, we consider the following three situations: (1) k=1, s=1 which is the usual REINFORCE algorithm (traditional PG algorithm). (2) k=5, s=5 which has no overlapping parts. (3) k=5, s=1. Here, we use CER and WER (lower is better) as our evaluation metrics. And we set the kernel's weight w to [1/5, 1/5, 1/5, 1/5, 1/5].

Baseline: We use the L_{CE} (shown as Eq.3) as the baseline.

$$\begin{aligned} L_{CE} &= -\log p\left(y_1, y_2, \ldots, y_n\right) \\ &= -\sum_{t=1}^{n} \log p\left(y_t \mid y_1, y_2, \ldots, y_{t-1}\right). \end{aligned}$$

Loss for the seq2seq model: The loss used to train the model is

$$L_{combine} = (1-\lambda) * L_{CE} + \lambda * L_{PCPG}, \tag{15}$$

where the λ is a scalar weight to balance the two loss functions.

Loss for the classifier model: To evaluate the performance of the PCPG in another way, we have classifying experiment in LRW and LRW-1000 based a full connected (FC) classifier. First, we would train the seq2seq model in LRW and LRW-1000 with the loss $L_{combine}$ and get a trained encoder and a trained decoder. Then we just use the trained encoder and set it as the pre-trained encoder for the classifying model. The loss used for the FC classifier is $L_{Classify}$ (shown as Eq.4).

$$L_{Classify} = -\sum_{k} P_k log P_k.$$

C. Ablation study

As has said above, the PCPG has receptive field (**RF**) and overlapping parts (**OP**) because of the convolutional operation in PCPG. We also explain that the existences of the **RF** and the **OP** can make the model obtain more robust performance and much better optimizing efficiency in section III-C. So here, we mainly have ablation study to verify our the performance of our PCPG. And we refer '+' to with **RF** (or **OP**) and '−' to no **RF** (or **OP**).

According to TABLE I, the model with RL but no **RF** and no **OP** (k=1, s=1) achieves better performance than the baseline. This shows that RL is effective for seq2seq lip-reading. The model with **RF** and **OP** ($k = 5$, s=1) achieves the best performance on all datasets. We can know that the proposed PCPG is more competitive than others for lip-reading. And according to Fig 6, 7 and 8, the PCPG can make the models more stable and take less time to converge than others during the training process.

TABLE I: The Experimental Results with Seq2Seq model on three different lip-reading datasets. (**RF**: receptive field, **OP**: overlapping parts)

Dataset	Method	Case	CER	WER
GRID	L_{CE}(*baseline*)	/	8.4%	18.3%
	− **RF** and − **OP**	k=1, s=1	7.6%	16.6%
	+ **RF** and − **OP**	k=5, s=5	6.9%	15.3%
	+ **RF** and + **OP**	k=5, s=1	**5.9%**	**12.3%**
LRW	L_{CE}(*baseline*)	/	17.5%	28.3%
	−**RF** and− **OP**	k=1, s=1	15.2%	24.8%
	+ **RF** and − **OP**	k=5, s=5	15.0%	26.5%
	+ **RF** and + **OP**	k=5, s=1	**14.1%**	**22.7%**
LRW-1000	L_{CE}(*baseline*)	/	52.1%	68.2%
	− **RF** and − **OP**	k=1, s=1	51.4%	67.7%
	+ **RF** and − **OP**	k=5, s=5	51.6%	67.2%
	+ **RF** and + **OP**	k=5, s=1	**51.3%**	**66.9%**

D. Evaluation of the kernel size k in PCPG

In order to explore the impact of the kernel size k on the PCPG'performance, we experiment on GRID. Here, we keep s at 1 to make the model have more overlapping parts to be more stable when training. To make the convolutional kernel have the same attention to the reward value at each time step, the kernel weight w is set to $[\frac{1}{k}, \frac{1}{k}, \ldots, \frac{1}{k}]$. The results are shown in Table II.

TABLE II: The experimental results when model has different k. (s=1)

Kernel size	WER
k=1	16.6%
k=2	16.0%
k=3	**12.1%**
k=5	12.3%
k=7	14.8%

As is shown in Table II, we get the best result when k=3 on GRID. If k is too small (such as k=1, 2) or too big (such as k=5, 7), the PCPG can't perform best. So we need to choose a proper receptive field according to our data.

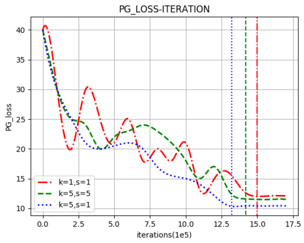

Fig. 6: L_{PCPG}-iteration on GRID.

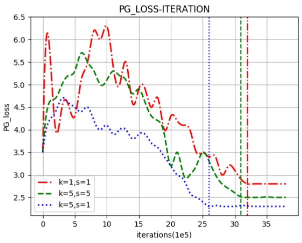

Fig. 7: L_{PCPG}-iteration on LRW.

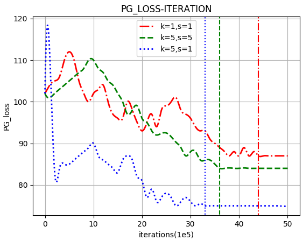

Fig. 8: L_{PCPG}-iteration on LRW-1000.

E. Evaluation of the kernel weight w in PCPG

Here, we also have some experiments to explore the impact of the kernel weight w on the PCPG's performance based on GRID. According to Section IV-D, we know the PCPG can performs best when k=3. So we set k to 3 and s to 1, here. The results are shown in Table III.

TABLE III: The experiments'results when model has different w. (k=3,s=1)

Kernel weight	WER
w=[1/3,1/3,1/3]	12.1%
w=[1/4,1/2,1/4]	**11.9%**
w=[1/3,1/2,1/6]	12.7%
w=[1/6,1/2,1/3]	12.6%

As is shown in Table III, when w=[1/4,1/2,1/4], the PCPG can perform best on GRID. It is important to balance the rewards of the current moment and the rewards of the moments before and after in PCPG. A proper weight w can improve the performance of the PCPG.

For the LRW and LRW-1000 are word-level lip-reading datasets, so it is natural for us to have classifying experiments to evaluate the PCPG'performance in extracting features for lip-reading. We first load the encoder's parameters which are trained by the PCPG in seq2seq model. Then we fix the encoder's parameters and only trained the classifier with $L_{Classify}$. At last, we set a small learning rate and train the encoder and classifier at the same time. The experiments'results are shown as TABLE IV. In classifying tasks, the accuracy is equal to (1-WER).

TABLE IV: The experimental results based on classifying on LRW and LRW-1000. (**NP**: no any pre-train, **FE**: fix encoder, **TE**: train encoder, **TC**: train classifier.)

Dataset	Method	Accuracy
LRW	**NP**	82.1%
	FE and **TC**	82.4%
	TE and **TC**	**83.5%**
LRW-1000	**NP**	37.8%
	FE and **TC**	38.5%
	TE and **TC**	**38.7%**

F. Comparison with state-of-the-art

For illustrating the effectiveness of our method, we compare our results with state-of-the-art based on other existed ways. We also show the results when considering the language model (LM) such as beam search on GRID. Here, the accuracy is equal to (1-WER). The results are shown as TABLE V and TABLE VI.

TABLE V: The experimental results based on seq2seq models on GRID and LRW. (The work in [1] is published in a poster from University of Oxford.)

Dataset	Method	WER
GRID	[38]	20.4%
	[12]	13.6%
	[3] (No LM)	13.6%
	ours (No LM)	**11.9%**
	[3] (With LM)	11.4%
	ours (With LM)	**11.2%**
LRW	[1]	23.8%
	ours	**21.5%**

TABLE VI: The experimental results based on classifying on LRW and LRW-1000. (**NP**: no any pre-train, **FE**: fix encoder, **TE**: train encoder, **TC**: train classifier.)

Dataset	Method	Accuracy
LRW	[25]	82.0%
	[32]	83.0%
	[31]	82.9%
	ours	**83.5%**
LRW-1000	[43]	38.19%
	ours	**38.70%**

According to these results from all of the experiments, we can find that our PCPG can also obtain a competitive performance comparing with other ways on sentence-level or word-level lip-reading. This proves that the PCPG is effective for lip-reading.

V. Conclusions

In this work, we proposed a pseudo-convolutional policy gradient (PCPG) based seq2seq model for the lip-reading task. Inspired by the principle of convolutional operation, we consider widen the policy gradient's receptive field and overlapping parts when training. We perform a thorough evaluation on both the word-level and the sentence-level dataset. Compared with the state-of-the-art results, we outperform or basically equal to the state-of-the-art performance, which proves the advantages of our proposed method. In fact, the PCPG can also be applied to other seq2seq tasks, such as machine translation, automatic speech recognition, image caption, video caption and so on. And in the future, we would try to make the process completely automatic, instead of manually setting the kernel's parameters.

References

[1] T. Afouras, J. S. Chung, and A. Zisserman. Deep Learning for Lip Reading. pages $https : //www.eng.ox.ac.uk/aims -$

$-cdt/wp--content/uploads/2017/11/poster2_{L}ipReading_{A}IMS--update.pdf$, 2017.

[2] T. Afouras, J. Son Chung, and A. Zisserman. Deep lip reading: A comparison of models and an online application. In *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, 2018.

[3] Y. M. Assael, B. Shillingford, S. Whiteson, and N. D. Freitas. LipNet: end-to-end sentence-level lipreading. 2016.

[4] J. S. C. B and A. Zisserman. Lip Reading in the Wild. *ACCV*, 2017.

[5] S. Banerjee and A. Lavie. METEOR : An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. *ACL*, 2005.

[6] T. H. Chen, Y. H. Liao, C. Y. Chuang, W. T. Hsu, J. Fu, and M. Sun. Show, Adapt and Tell: Adversarial Training of Cross-Domain Image Captioner. *ICCV*, 2017.

[7] S. Chopra, M. Auli, and W. Zaremba. SEQUENCE LEVEL TRAINING WITH RECURRENT NEURAL NETWORKS. *ICLR*, 2016.

[8] J. S. Chung. Lip Reading Sentences in the Wild. *ICCV*, 2017.

[9] J. S. Chung and A. Zisserman. Lip Reading in Profile. *BMVC*, 2017.

[10] J. S. Chung and A. Zisserman. Learning to lip read words by watching videos. *Computer Vision and Image Understanding*, (February), 2018.

[11] B. J. C. S. S. X. Cooke, M. An audio-visual corpus for speech perception and automatic speech recognition. 2006.

[12] S. Gergen, S. Zeiler, A. H. Abdelaziz, R. Nickel, and D. Kolossa. Dynamic stream weighting for turbo-decoding-based audiovisual ASR. *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, 08-12-September-2016(April 2017):2135–2139, 2016.

[13] K. He. Deep Residual Learning for Image Recognition.

[14] D. Hu, X. Li, and X. Lu. Temporal Multimodal Learning in Audiovisual Speech Recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016-December:3574–3582, 2016.

[15] T. M. Jianlong Fu, Heliang Zheng. Look Closer to See Better: Recurrent Attention Convolutional Neural Network for Fine-grained Image Recognition. *CVPR*, 2014.

[16] Y. Lan, R. W. Harvey, B.-J. Theobald, E.-J. Ong, and R. Bowden. Comparing Visual Features for Lipreading. *Avsp 2009*, pages 102–106, 2009.

[17] L. Li and B. Gong. End-to-End Video Captioning with Multitask Reinforcement Learning. *WACV*, 2018.

[18] C.-y. Lin and M. Rey. ROUGE : A Package for Automatic Evaluation of Summaries. *ACL*, 2004.

[19] R. McConnell, K. Berhane, F. Gilliland, S. J. London, T. Islam, W. J. Gauderman, E. Avol, H. G. Margolis, and J. M. Peters. Asthma in exercising children exposed to ozone: A cohort study. 2002.

[20] R. Nallapati, B. Zhou, C. dos Santos, C. Gulcehre, and B. Xiang. Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond. pages 280–290, 2016.

[21] K. Papineni, S. Roukos, T. Ward, and W.-j. Zhu. BLEU : a Method for Automatic Evaluation of Machine Translation. *ACL*, (July), 2002.

[22] A. P. Parikh and J. Uszkoreit. Sequence to Sequence Learning with Neural Networks. *NIPS*, 2014.

[23] A. P. Parikh and J. Uszkoreit. A Decomposable Attention Model for Natural Language Inference. *ACL*, 2016.

[24] R. Pasunuru and M. Bansal. Reinforced Video Captioning with Entailment Rewards. *EMNLP*, 2017.

[25] S. Petridis, T. Stafylakis, P. Ma, and F. Cai. END-TO-END AUDIOVISUAL SPEECH RECOGNITION. *ICASSP*, 2018.

[26] R. Prabhavalkar, K. Rao, T. N. Sainath, B. Li, L. Johnson, and N. Jaitly. A Comparison of sequence-to-sequence models for speech recognition. *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, 2017-Augus:939–943, 2017.

[27] Z. Ren, X. Wang, and N. Zhang. Deep Reinforcement Learning-based Image Captioning with Embedding Reward. *CVPR*, 2017.

[28] S. J. Rennie, E. Marcheret, Y. Mroueh, J. Ross, and V. Goel. Self-critical Sequence Training for Image Captioning. *CVPR*, 2017.

[29] T. N. S. B. L. L. N. J. Rohit Prabhavalkar, Kanishka Rao. A Comparison of Sequence-to-Sequence Models for Speech Recognition. *Interspeech*, 2017.

[30] Y.-l. Shen, C.-y. Huang, S.-s. Wang, Y. Tsao, H.-m. Wang, T.-s. Chi, C. Engineering, and N. Chiao. Reinforcement learning based speech enhancement for robust speech recognition. *ICASSP*, 2019.

[31] T. Stafylakis, M. H. Khan, and G. Tzimiropoulos. Pushing the boundaries of audiovisual word recognition using Residual Networks and LSTMs. *Computer Vision and Image Understanding*, 176-177:22–32, 2018.

[32] T. Stafylakis and G. Tzimiropoulos. Combining residual networks with LSTMs for lipreading. *Interspeech*, 2017.

[33] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. *Advances in Neural Information Processing Systems*, 4(January):3104–3112, 2014.

[34] V. Tech, C. L. Zitnick, and D. Parikh. CIDEr: Consensus-based Image Description Evaluation. *CVPR*, 2015.

[35] A. Tjandra, S. Sakti, and S. Nakamura. Sequence-to-Sequence ASR Optimization via Reinforcement Learning. *ICASSP*, 2018.

[36] A. Vaswani. Attention Is All You Need. *NIPS*, 2017.

[37] S. Venugopalan, M. Rohrbach, T. Darrell, J. Donahue, K. Saenko, and R. Mooney. Sequence to Sequence Video to Text.

[38] M. Wand, J. Koutník, and J. Schmidhuber. Lipreading with long short-term memory. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2016-May:6115–6119, 2016.

[39] X. Wang, W. Chen, J. Wu, and Y.-f. Wang. Video Captioning via Hierarchical Reinforcement Learning. *CVPR*, 2018.

[40] R. J. Willia. Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Machine Learning*, 8, 1992.

[41] S. Wiseman and A. M. Rush. Sequence-to-Sequence Learning as Beam-Search Optimization. *EMNLP*, 2016.

[42] K. Xu, A. Courville, R. S. Zemel, and Y. Bengio. Show , Attend and Tell : Neural Image Caption Generation with Visual Attention. *ICML*, 2015.

[43] S. Yang, Y. Zhang, D. Feng, M. Yang, C. Wang, J. Xiao, K. Long, S. Shan, and X. Chen. A Naturally-Distributed Large-Scale Benchmarkfor Lip Reading in the Wild. *IEEE FG*, 2018.

[44] W. Zhang, Y. Feng, F. Meng, D. You, and Q. Liu. Bridging the Gap between Training and Inference for Neural Machine Translation. 2019.

[45] Z. Zhou, G. Zhao, X. Hong, and M. Pietikäinen. A review of recent advances in visual speech decoding. *IMAVIS*, (9), 2014.