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

BMVC 2019 Submission # 1175

Abstract

Lip-reading has been proved to be a very challenging task due to the large gap between the very limited effective area of lips and the great variety of words we can say. Most existing methods perform this task as a classification task or use the CTC loss for sentence-level lip-reading. But in fact, we think that lip-reading can be seen as a typical sequence-to-sequence (seq2seq) task which translates the input sequence of lip movements to the output sequence of the corresponding speech content. However, the traditional learning process of the seq2seq models could not fit the lip-reading problem very well for two main reasons: 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 the above two problems for seq2seq lip-reading. Specifically, we propose a new method called pseudo-convolutional policy gradient (PCPG) for lip-reading. As far as we know, we are the first to explore the power of reinforcement learning for lip-reading. And we are also the first to introduce the concept of convolution to reinforcement learning, which may provide some insights for the community in related research fields. Finally, extensive ablation study and comparison on both the word-level and sentence-level datasets are given, which show that the proposed PCPG-based seq2seq model not only outperformed the traditional seq2seq models by a large margin, but also achieved the state of the art on the largest word-level lip-reading datasets. Especially, Our results on LRW-1000 achieve a new state-of-the-art on lip-reading task, improving the best result in terms of accuracy from 38.19% to 38.7%.

1 Introduction

Lip-reading is a new way of human-computer interaction which has received increasing attention in recent years. It is a very attractive skill and aims to understand speech using only visual information[3]. It can be used as a strong complement of the automatic audio-based speech recognition systems. Lip-reading can also play an important role in many scenarios, such as transcribing and re-dubbing archival silent films, resolving multi-talker simultaneous speech, liveness verification and so on[8]. Lip-reading is a challenging task which involves several types of technologies simultaneously, including technologies about action recognition, temporal modeling, cross-modality learning and so on. With the widespread success of deep learning (DL) and the emergence of large-scale databases such as OuluVS1[36], OuluVS2[13], GRID[11], LRW[3], LRW-1000[35], LRS[8] etc, lip-reading has achieved great progresses these two years.

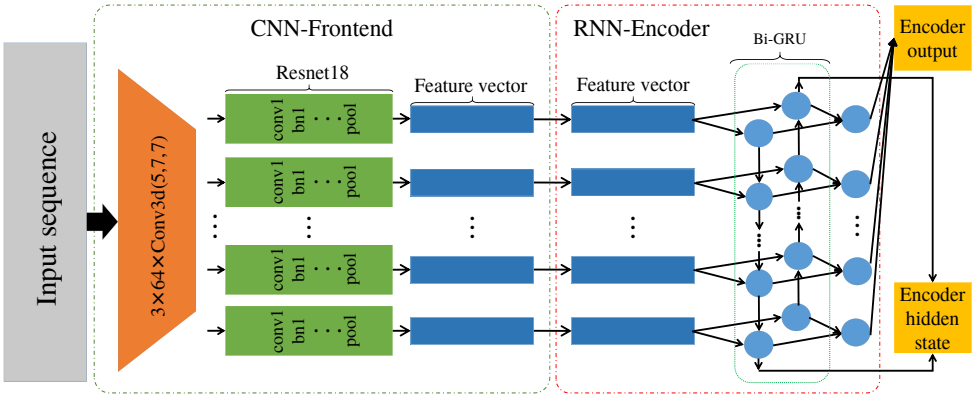

Figure 1: Video encoder.

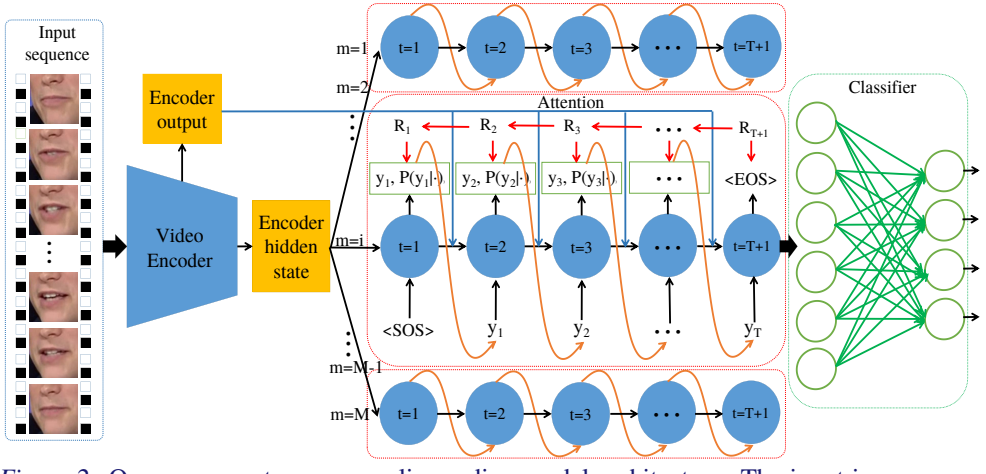

Figure 2: Our sequence-to-sequence lip-reading model architecture. The input is sequence frames and the output is sequence characters. The seq2seq model generates the target character by character. For example, the model generates ("a","b","o","u","t") successively when the ground truth is the word "about". The classifier can work on word-level lip-reading datasets. We utilize Monte Carlo sampling to sample M transcription sequences from our model to calculate the policy gradient.

In fact, lip-reading is a typical sequence-sequence (seq2seq) task which takes the lip movement sequence as input and output a character sequence. It has a high similarity with other seq2seq tasks such as speech recognition[25], machine translation[19], image caption[6], video caption[30], and so on. Seq2seq models (especially encoder-decoder with attention) have been proved to be effective for these seq2seq tasks.

However, there are still two main drawbacks when we use the seq2seq models directly for lip-reading. The one is exposure bias[7]. Most seq2seq models perform prediction at each time-step with a heavy dependence on the previous ground-truth words in the training process. This approach is called "teacher-forcing"[24] which has been used widely in NLP and other fields. But when coming to the testing process, the prediction at each time-step is generated by feeding the prediction results of the previous time-step as the input, which is totally different with the input in the training stage. This discrepancy leads to exposure

bias. The other one is the gap between the optimized target and the final non-differentiable evaluation metrics is also one key problem for most seq2seq models[7]. One popular target is the cross-entropy. It has one key problem that it trains the models to be good at greedily predicting the next word at each time-step without considering the whole sequence[7]. But the evaluation metrics for seq2seq tasks are often based on the whole output sequence.

In this paper, we use seq2seq models for the lip-reading task. Our framework is shown in Figure 1 and Figure 2. We set this task at a character-level to adapt different-level lip-reading datasets with one model and avoid the Out-Of-Vocabulary (OOV) problem. To address the exposure bias problem and the mismatch metrics problem, we introduce RL methods (especially the REINFORCE algorithm) to seq2seq lip-reading. Moreover, we propose a novel policy gradient training method called pseudo-convolutional policy gradient (PCPG), which can not only improve the generalization ability of seq2seq model but also solve the inefficiency and the instability existed in the usual REINFORCE algorithm in the training process. We demonstrate the performance of the PCPG in seq2seq lip-reading on three different-level lip-reading datasets (the sentence-level lip-reading dataset: GRID, the two large-scale word-level lip-reading datasets: LRW, LRW-1000) by experiments.

2 Related work

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 [1, 2, 3, 8, 9, 10, 22, 35, 37]. The prior work for character-level lip-reading can be divided into two strands[1]. The one is based on CTC (Connectionist Temporal Classification)[12]. For example, the work in [2] presented a model called LipNet which uses a spatio-temporal 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[2]. And the work in [1] used CTC and beam search[33] for lip-reading. The other one is based on seq2seq models (encoder-decoder with attention), such in [8], where the model can combine the audio and visual input streams. The attention mechanism[14, 20, 29, 34], which is very popular recently, has become a necessary part of seq2seq models. And 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. Such in [22], the end-to-end model got 82% accuracy of word classification by a softmax classifier for lip-reading. But the model based on classifying can't be used for sentence-level lip-reading tasks.

Reinforcement learning for seq2seq models: According to the above, the usual seq2seq optimized methods have two main problems: the exposure bias and the mismatch metrics. The exposure bias leads to error accumulation during the testing process. 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[18], ROUGE[16], METEOR[9], CIDEr[27], WER (word error rate), CER (character error rate), SER (sentence error rate) etc. These two problems hinder the performance of seq2seq models.

In recent years, to address the above two main problems, RL methods are used in more and more seq2seq tasks, such as machine translation, news headline generation, automatic speech recognition (ASR), text summarization, image caption, video caption, and so on[5]. In RL, the model generates the current target based on the previous predicting characters instead of the previous ground-truth characters in training process. In this way, the model can learn a stronger relationship between characters in output character sequence than using "teacher-forcing". Most of tasks get a much better performance with RL methods, such as [7,

[15, 17, 21, 23, 24, 26, 28, 31]. For example, the work in [28] proposed an alternative strategy for training a seq2seq ASR by RL methods. The other work in [24], 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[32] 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.

3 The Proposed work

In this section, we introduce our whole work that PCPG (Pseudo-Convolutional Policy Gradient) for seq2seq lip-reading in detail. This section mainly includes three parts: (1) Model architecture — a typical seq2seq model which includes a video encoder, a character-decoder with attention and a softmax classifier shown in Figure 1, Figure 2 and Section 3.1. (2) REINFORCE algorithm— a kind of policy gradient algorithms shown in Section 3.2.(3) Reward function and PCPG(Pseudo-Convolutional Policy Gradient) shown in Section 3.3.

3.1 Model architecture

Video encoder: As shown in Figure 1, this video encode can extract the spatio-temporal information by other representations from the lip-reading video. So with this video encoder, the input video frame sequence $\mathbf{x}^v = \left(x_1^v, x_2^v, \ldots, x_k^v\right)$ can be re-represented as the encoder output $\mathbf{o}^v = (o_1^v, o_2^v, \ldots, o_m^v)$ and a fixed-dimensional hidden-state vector encoder hidden h_e^v. We can define this process with equation:

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

Character decoder with attention: We define the ground-truth output character sequence as $\mathbf{c} = [c_1, c_2, \ldots, c_n]$, and the generated output character sequence as $\mathbf{y} = [y_1, y_2, \ldots, y_n]$. Here, we use GRU as the decoding unit $Decoder_j (j <= n)$ which generates y_j at time-step i and receives three inputs: the previous generated output y_{j-1}, the previous GRU decoding unit's hidden state h_{j-1}^d and the attention a_j. The character decoder with attention is shown in Figure 2.

$$\begin{aligned} e_{ji} &= attention\left(h_{j-1}^d, o_i^v\right) (i <= m), \\ \alpha_{ji} &= \tfrac{e_{ji}}{\sum_i e_{ji}}; \quad a_j = \textstyle\sum_i \alpha_{ji} o_i^v, \\ y_j &= Decoder_j\left(h_{j-1}^{\mathrm{d}}, y_j^d, a_j\right). \end{aligned} \tag{2}$$

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:

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

Classifier: As shown in Figure 2, there is a simple fully connected classifying neural network at the end of character decoder, which includes a fully connected layer and a softmax layer. The classifier's loss function $L_{classify}$ can be computed as follows:

$$L_{Classify} = -\sum_k P_k log P_k. \tag{4}$$

3.2 REINFORCE algorithm

We introduce REINFORCE algorithms[32] to seq2seq lip-reading. In this work, our model can be viewed as an 'agent' that interacts with an external 'environment' (words or sentences and video frames). And the parameters of the model, θ, define a policy p_θ, that results in an 'action' (choosing a character). After each time step j, the agent will get new internal 'state' (the attention, the previous hidden state and the generated character) and the agent will receive an immediate reward r_j and the final reward R. So our training goal is to maximize expected reward $E_y[R|p_\theta]$, and $L_{PG} = -E_y[R|p_\theta]$ is our loss function. We update the parameters as follows if get a generated pair of transcription ($\mathbf{x}^v = \left(x_1^v, x_2^v, \ldots, x_k^v\right)$, $\mathbf{y} = (y_1, y_2, \ldots, y_n)$) from the model:

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

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 $R = \sum_{j=1}^{n} R_j$. So the gradient can be computed as follows:

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

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 Figure 2. So the gradient can be computed with Eq.7:

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

So we can get the final gradient:

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

3.3 Reward function & PCPG (Pseudo-Convolutional Policy Gradient)

Reward function: In lip-reading tasks, the performance of the model is evaluated on CER and WER. And both of them are computed 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:

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

$ED(\cdot,\cdot)$ refers to CER, which is computed by edit-distance algorithm, $\mathbf{y}_{1:j}$ is the sub-string of $\mathbf{y}$ from index 1 to j. and $|\mathbf{c}|$ is the ground-truth length.

PCPG(Pseudo-Convolutional Policy Gradient): As we known, the output in character-level lip-reading is a character sequence. How to get a correct and smooth character sequence is not only a recognition task but also an NLP task. The output character at each time step depends not only on the encoding power of the model but also on the context. Inspired by

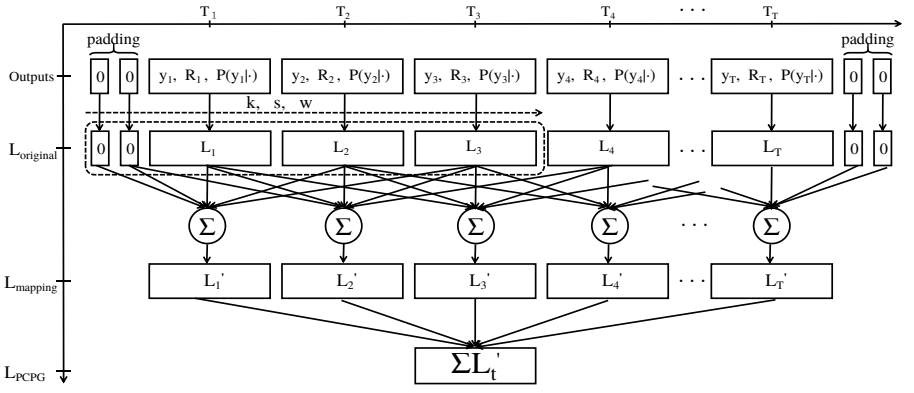

Figure 3: Pseudo-convolutional Policy Gradient. In this figure, we set the kernel size k to 5, the stride s to 1, and the kernel weights w to $[\frac{1}{5},\frac{1}{5},\frac{1}{5},\frac{1}{5},\frac{1}{5}]$.

the idea of convolution, especially the temporal convolutional network (TCN), we propose a novel Pseudo-Convolutional Policy Gradient (PCPG) method to optimize the seq2seq model used for lip-reading. In TCN, the convolution kernel increases the receptive field of input at each time step when encoding. So the TCN has a stronger power for encoding sequence information. Similar to TCN, as is shown in Figure 3, in PCPG, the existence of the convolution kernel increase the receptive field of the output character y_t at each time step in the decoding process. In PCPG, the convolution kernel has size k, stride s and weight w like TCN. Here, we illustrate the PCPG based on the case that k=5, s=1, w=$[\frac{1}{5},\frac{1}{5},\frac{1}{5},\frac{1}{5},\frac{1}{5}]$. We also pad input the front and back to keep the original sequence length. These pads are as follows: $L_{-1}=0, L_0=0, L_{T+1}=0, L_{T+2}=0$.

Here, we use one of M transcription sequences based on Monte Carlo sampling to illustrate the PCPG. According to the REINFORCE algorithm, the loss at each time step L_t can be computed with the following equation:

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

In PCPG, we define $\nabla_\theta L^t_{PCPG}$ as the local gradient and $\frac{\partial L_{PCPG}}{\partial \theta}$ as the final gradient. And they can be computed as follows:

$$L_t^{'} = L_{mapping} = L_{t-2:t+2} * w = \frac{1}{5}\sum_{i=t-2}^{t+2} L_i = \frac{1}{5}\left[L_{t-2}+L_{t-1}+L_t+L_{t+1}+L_{t+2}\right], \tag{11}$$

$$L_{PCPG} = \sum_{t=1}^{T} L_t^{'} = \sum_{t=1}^{T}\frac{1}{5}\sum_{i=t-2}^{t+2} L_i, \tag{12}$$

$$\frac{\partial L_{PCPG}}{\partial \theta} = \sum_{t=1}^{T}\frac{\partial L_{PCPG}}{\partial L_t^{'}}\frac{\partial L_t^{'}}{\partial \theta} = \frac{1}{5}\sum_{t=1}^{T}\frac{\partial L_{PCPG}}{\partial L_t^{'}}\sum_{i=t-2}^{t+2}\frac{\partial L_t^{'}}{\partial L_i}\frac{\partial L_i}{\partial \theta}, \tag{13}$$

$$\nabla_\theta L^t_{PCPG} = \frac{1}{5}\frac{\partial L_{PCPG}}{\partial L_t^{'}}\sum_{i=t-2}^{t+2}\frac{\partial L_t^{'}}{\partial L_i}\frac{\partial L_i}{\partial \theta}, \tag{14}$$

$$\nabla_\theta L^{t+1}_{PCPG} = \frac{1}{5}\frac{\partial L_{PCPG}}{\partial L_{t+1}^{'}}\sum_{i=t-1}^{t+3}\frac{\partial L_t^{'}}{\partial L_i}\frac{\partial L_i}{\partial \theta}. \tag{15}$$

According to Eq.14, we can find that the $\nabla_\theta L^t_{PGPC}$ has bigger optimized receptive field than the usual REINFORCE algorithm at time-step t. Combining with Eq.10, we can know that the $\nabla_\theta L^t_{PGPC}$ depends on the rewards at multiple times. In this way, there will be a stronger timing dependency between the output characters. And it also will guide the model to learn a parameter distribution which can make the model get a bigger regional comprehensive reward at each time step. So the PCPG can make the model have a better generalization ability for lip-reading.

Moreover, 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 making a decision in RL. According to the Eq.10 and Eq.14, we can find that the updated value of the gradient at each time step will be constrained by the reward value at multiple times. So the PCPG can restrain the instability in training process. The kernel weight $w = [w_1, w_2, \ldots, w_k]$ can also ensure the value of the gradient in PCPG at a same quantitative level with the REINFORCE method by the following equation:

$$\sum_{i=1}^{k} w_i = 1. \tag{16}$$

By comparing Eq.14 and Eq.15, we can know that the local gradients at the adjacent time steps have many overlapping parts. In this way, the model can adjust the local gradient for many times at time step t to get a bigger overall reward. And due to the existence of the overlapping parts, the local gradient at adjacent time step will not change dramatically. So the PCPG can not only improve efficiency but also make more stable when training.

In the above example, there are two extreme cases: (1) when s=5, there is no overlapping parts in this case. (2) when k=1, s=1, this is the usual REINFORCE method. So the usual REINFORCE algorithm is a special PCPG case.

4 Experiments

In this section, the experiments on both the word-level and sentence-level datasets show a clear demonstration of the advantages of the proposed PCPG for lip-reading. At the same time, we also discuss the effects of convolution kernel hyperparameters on PCPG through experiments. In our experiments, the model converges difficultly if only use the L_{PCPG} to optimize. In our experiments, we set L_{CE} as the baseline. To ensure the training process's stability and a good optimization direction, we give the model a pre-training model trained by the CE loss and use $L_{combine}$ that combines the cross-entropy loss L_{CE} with the reward loss L_{PCPG} to continue training the model. The $L_{combine}$ is defined as follows:

$$L_{combine} = (1-\lambda) * L_{CE} + \lambda * L_{PCPG}, \tag{17}$$

where the λ is to balance the two loss functions. And we also explore the effectiveness of PCPG for classifying based on word-level by the following $L^{'}_{combine}$:

$$L^{'}_{Classify} = \alpha * L_{CE} + \beta * L_{Classify} + (1-\alpha-\beta) * L_{PCPG} \tag{18}$$

where the α and β are weight factors.

We use the L_{CE} as the baseline. For PCPG, we consider the following three situations: (1) k=1, s=1 which is the usual REINFORCE 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 $[\frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}]$.

4.1 Ablation study

We have our ablation study on three different-level datasets including GRID (the sentence-level dataset), LRW (the word-level dataset), LRW-1000 (the word-level dataset). The results trained with $L_{Classify}$ and $L'_{Classify}$ are based on classification of word level. The others are based on character level decoding with seq2seq models. All of results are shown in Table 1.

According to Table 1, Figure 4, 5 and 6, 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.

Dataset	Method	Case	CER	WER	Accuracy
GRID	$L_{CE}(baseline)$	/	8.4%	18.3%	81.7%
	No **RF** and no **OP**	k=1, s=1	7.6%	16.6%	83.4%
	With **RF** and no **OP**	k=5, s=5	6.9%	15.3%	84.7%
	With **RF** and with **OP**	k=5, s=1	**5.9%**	**12.3%**	**87.7%**
	LipNet (CTC)[2]	/	6.4%	11.4%	88.6%
LRW	$L_{CE}(baseline)$	/	17.5%	28.3%	71.7%
	No **RF** and no **OP**	k=1, s=1	17.2%	26.8%	73.2%
	With **RF** and no **OP**	k=5, s=5	17.0%	26.5%	73.5%
	With **RF** and with **OP**	k=5, s=1	**16.6%**	**25.7%**	**74.3%**
	$L_{Classify}$	/	/	21.5%	78.5%
	$L'_{Classify}$ (With **RF** and with **OP**)	k=5, s=1	/	**19.8%**	**80.2%**
LRW-1000	$L_{CE}(baseline)$	/	52.1%	68.2%	31.8%
	No **RF** and no **OP**	k=1, s=1	51.4%	67.7%	32.3%
	With **RF** and no **OP**	k=5, s=5	51.6%	67.2%	32.8%
	With **RF** and with **OP**	k=5, s=1	**51.3%**	**66.9%**	**33.1%**
	$L_{Classify}$	/	/	61.5%	38.5%
	$L'_{Classify}$ (With **RF** and with **OP**)	k=5, s=1	/	**61.3%**	**38.7%**
	Classify[35]	/	/	61.81%	38.19%

Table 1: The experimental results on GRID. Accuracy=1-WER. (**RF**: receptive field, **OP**: overlapping parts)

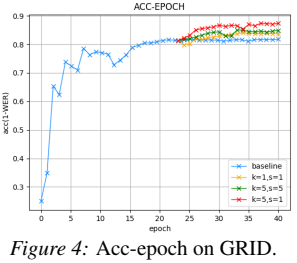

Figure 4: Acc-epoch on GRID.

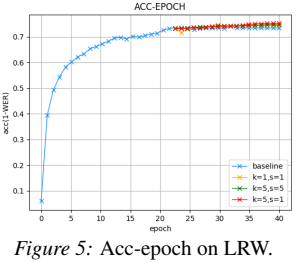

Figure 5: Acc-epoch on LRW.

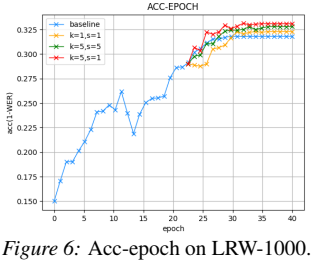

Figure 6: Acc-epoch on LRW-1000.

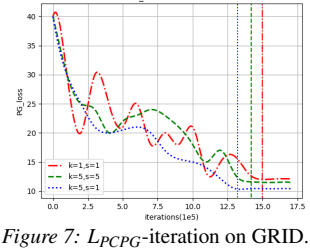

Figure 7: L_{PCPG}-iteration on GRID.

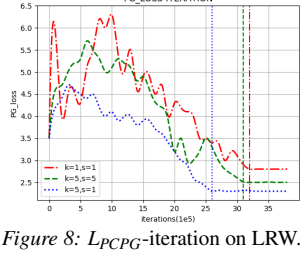

Figure 8: L_{PCPG}-iteration on LRW.

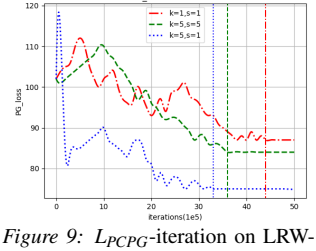

Figure 9: L_{PCPG}-iteration on LRW-1000.

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 Figure 7, 8 and 9, the PCPG can make the models more stable and take less time to converge than others during the training process.

4.2 Evaluation of the kernel size k in PCPG

In order to explore the impact of the kernel size k on the PCPG's 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 2.

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

Table 2: The experimental results when model has different k. (s=1)

As is shown in Table 2, 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.

4.3 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 4.2, 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 3.

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%

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

As is shown in Table 3, 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.

5 Conclusions

In this work, we introduce RL (reinforcement learning) into seq2seq lip-reading based on character level. And we put forward the PCPG (Pseudo-Convolutional Policy Gradient) to optimize the model. We verify the PCPG is effective for lip-reading by the experiments on three different-level datasets (GRID, LRW, LRW-1000). And the model can perform more stable and robust when using PCPG with proper convolutional kernel hyperparameters than the usual REINFORCE algorithm. Our method can also improve the performance of classifier to a certain degree. Specially, we believe that the PCPG can also be applied to other seq2seq tasks, such as machine translation, automatic speech recognition, image caption, video caption and so on.

References

[1] Triantafyllos Afouras, Joon Son Chung, and Andrew Zisserman. Deep Lip Reading : a comparison of models and an online application. *Interspeech*, 2018.

[2] Yannis M Assael, Brendan Shillingford, Shimon Whiteson, and Nando De Freitas. LipNet: end-to-end sentence-level lipreading. 2016.

[3] Joon Son Chung B and Andrew Zisserman. Lip Reading in the Wild. *ACCV*, 2017.

[4] Satanjeev Banerjee and Alon Lavie. METEOR : An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. *ACL*, 2005.

[5] Shan Chen and John Steinberger. Deep Reinforcement Learning for Sequence-to-Sequence Models. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2014.

[6] Tseng Hung Chen, Yuan Hong Liao, Ching Yao Chuang, Wan Ting Hsu, Jianlong Fu, and Min Sun. Show, Adapt and Tell: Adversarial Training of Cross-Domain Image Captioner. *ICCV*, 2017.

[7] Sumit Chopra, Michael Auli, and Wojciech Zaremba. SEQUENCE LEVEL TRAINING WITH RECURRENT NEURAL NETWORKS. *ICLR*, 2016.

[8] Joon Son Chung. Lip Reading Sentences in the Wild. *ICCV*, 2017.

[9] Joon Son Chung and Andrew Zisserman. Lip Reading in Profile. *BMVC*, 2017.

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

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

[12] Alex Graves and Santiago Fern. Connectionist Temporal Classification : Labelling Unsegmented Sequence Data with Recurrent Neural Networks. *ICMl*, 2006.

[13] Guoying Zhao Iryna Anina, Ziheng Zhou and Matti PietikÂĺainen. OuluVS2: a multi-view audiovisual database for non-rigid mouth motion analysis. *IEEE FG*, 2015.

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

[15] Lijun Li and Boqing Gong. End-to-End Video Captioning with Multitask Reinforcement Learning. *WACV*, 2018.

[16] Chin-yew Lin and Marina Rey. ROUGE : A Package for Automatic Evaluation of Summaries. *ACL*, 2004.

[17] Rob McConnell, Kiros Berhane, Frank Gilliland, Stephanie J. London, Talat Islam, W. James Gauderman, Edward Avol, Helene G. Margolis, and John M. Peters. Asthma in exercising children exposed to ozone: A cohort study. 2002.

[18] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-jing Zhu. BLEU : a Method for Automatic Evaluation of Machine Translation. *ACL*, (July), 2002.

[19] Ankur P Parikh and Jakob Uszkoreit. Sequence to Sequence Learning with Neural Networks. *NIPS*, 2014.

[20] Ankur P Parikh and Jakob Uszkoreit. A Decomposable Attention Model for Natural Language Inference. *ACL*, 2016.

[21] Ramakanth Pasunuru and Mohit Bansal. Reinforced Video Captioning with Entailment Rewards. *EMNLP*, 2017.

[22] Stavros Petridis, Themos Stafylakis, Pingchuan Ma, and Feipeng Cai. END-TO-END AUDIOVISUAL SPEECH RECOGNITION. *ICASSP*, 2018.

[23] Zhou Ren, Xiaoyu Wang, and Ning Zhang. Deep Reinforcement Learning-based Image Captioning with Embedding Reward. *CVPR*, 2017.

[24] Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. Self-critical Sequence Training for Image Captioning. *CVPR*, 2017.

[25] Tara N. Sainath Bo Li LeifJohnson Navdeep Jaitly Rohit Prabhavalkar, Kanishka Rao. A Comparison of Sequence-to-Sequence Models for Speech Recognition. *Interspeech*, 2017.

[26] Yih-liang Shen, Chao-yuan Huang, Syu-siang Wang, Yu Tsao, Hsin-min Wang, Tai-shih Chi, Computer Engineering, and National Chiao. Reinforcement learning based speech enhancement for robust speech recognition. *ICASSP*, 2019.

[27] Virginia Tech, C Lawrence Zitnick, and Devi Parikh. CIDEr: Consensus-based Image Description Evaluation. *CVPR*, 2015.

[28] Andros Tjandra, Sakriani Sakti, and Satoshi Nakamura. Sequence-to-Sequence ASR Optimization via Reinforcement Learning. *ICASSP*, 2018.

[29] Ashish Vaswani. Attention Is All You Need. *NIPS*, 2017.

[30] Subhashini Venugopalan, Marcus Rohrbach, Trevor Darrell, Jeff Donahue, Kate Saenko, and Raymond Mooney. Sequence to Sequence-Video to Text. *ICCV*, 2015.

[31] Xin Wang, Wenhu Chen, Jiawei Wu, and Yuan-fang Wang. Video Captioning via Hierarchical Reinforcement Learning. *CVPR*, 2018.

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

[33] Sam Wiseman and Alexander M Rush. Sequence-to-Sequence Learning as Beam-Search Optimization. *EMNLP*, 2016.

[34] Kelvin Xu, Aaron Courville, Richard S Zemel, and Yoshua Bengio. Show , Attend and Tell : Neural Image Caption Generation with Visual Attention. *ICML*, 2015.

[35] Shuang Yang, Yuanhang Zhang, Dalu Feng, Mingmin Yang, Chenhao Wang, Jingyun Xiao, Keyu Long, Shiguang Shan, and Xilin Chen. A Naturally-Distributed Large-Scale Benchmarkfor Lip Reading in the Wild. *IEEE FG*, 2018.

[36] Guoying Zhao, Mark Barnard, and Matti Pietikäinen. Lipreading with local spatiotemporal descriptors. *IEEE Transactions on Multimedia*, 2009.

[37] Ziheng Zhou, Guoying Zhao, Xiaopeng Hong, and Matti Pietikäinen. A review of recent advances in visual speech decoding. *IMAVIS*, (9), 2014.