Look Back and Predict Forward in Image Captioning

Yu Qin[1,2], Jiajun Du[1], Yonghua Zhang[2], Hongtao Lu[1*]
[1] Key Lab of Shanghai Education Commission for Intelligent Interaction and Cognitive Engineering,
Department of Computer Science and Engineering,
MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University, China
[2]AI-Lab Visual Search Team, Bytedance
{qinyu123,dujiajun,htlu}@sjtu.edu.cn, {mpeg21}@hotmail.com

Abstract

Most existing attention-based methods on image captioning focus on the current word and visual information in one time step and generate the next word, without considering the visual and linguistic coherence. We propose Look Back (LB) method to embed visual information from the past and Predict Forward (PF) approach to look into future. LB method introduces attention value from the previous time step into the current attention generation to suit visual coherence of human. PF model predicts the next two words in one time step and jointly employs their probabilities for inference. Then the two approaches are combined together as LBPF to further integrate visual information from the past and linguistic information in the future to improve image captioning performance. All the three methods are applied on a classic base decoder, and show remarkable improvements on MSCOCO dataset with small increments on parameter counts. Our LBPF model achieves BLEU-4 / CIDEr / SPICE scores of 37.4 / 116.4 / 21.2 with cross-entropy loss and 38.3 / 127.6 / 22.0 with CIDEr optimization. Our three proposed methods can be easily applied on most attention-based encoder-decoder models for image captioning.

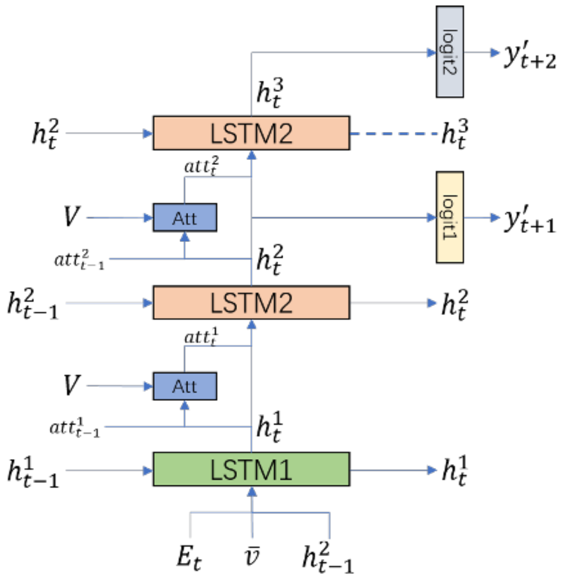

Figure 1. An overview of our proposed Look Back and Predict Forward (LBPF) method. The same color of LSTM2 and Att indicates they share the same parameters. E_t denotes the word embedding of y_t. The Att layer takes both hidden state h and previous attention result att_{t-1} as input to predict the current attention value, which shows the look-back part. The network predicts y'_{t+1} and y'_{t+2} at the same timestep, and shares the Att and LSTM2 parameters, which shows the predict-forward part.

1. Introduction

Image caption generation is to generate a sentence to describe an image in natural language from its visual contents and has attracted increasing attention in computer vision field. This task combines image understanding and natural language processing methods to predict an informative and fluent caption. The generated texts can do great help for people with visual impairment and image searching problems. Caption generation is a challenging task. First, the image needs to be well understood and the primary information should be extracted as features for text generation. Second, a language model should be employed to identify concrete information from the extracted features and generate descriptive texts. In the past few years, many neural encoder-decoder models [27, 25, 28] have been proposed to solve this problem, where encoder extracts semantic embeddings from the image based on Convolutional Neural Network (CNNs) and decoder predicts the texts using Recurrent Neural Networks (RNNs).

Attention mechanism is first introduced into this task

*Corresponding author.

in [29] and shows extraordinary advantage on all the objective metrics like BLEU [22], CIDEr [26] and SPICE [1]. Attention module takes a look at semantic embeddings from the encoder, and focuses on the crucial information related to the current expression. At each time step, the attention module calculates a relevance weight according to the given word which contributes to the current word generation. Soft attention [5] method employs the weighted average of all the feature vectors as attention result, while hard attention [4] method performs a sampling on the relevance weights. Although attention module in the decoder can provide precise and effective visual information for the text generation, the attention methods simply take the current word state h_t as input and calculate the attention result for only one output state h_{t+1}. This kind of attention ignores the visual relevance between adjacent words, *e.g.* "a blue bike" actually expresses one object with three words.

As far as we know, all existing works on image captioning generate word one by one and the predicted word y'_{t+1} is highly dependent on y_t. During the training phase, they take ground-truth y_t as the input of the language model to predict the word y'_{t+1}. However, in the inference phase, the input y'_t can only be sampled from the last time step, which can easily bring in accumulated error in final generated texts. Previous work like [9] proposes to increase the correlation between h_{t-1} and h_t by coupling with an auto-reconstructor network (ARnet). This reduces the difference between h_{t-1} and h_t, and can embed more information from the previous state. However, the regularization method using Euclidean distance may directly reduce the L-2 norm for each hidden state, and show little intuitionistic improvement.

In order to solve the problems mentioned above, we propose Look Back and Predict Forward method (LBPF) to utilize the visual information and language modeling ability. As illustrated in Figure 1, our LBPF method introduces two main designs denoted as Look-Back (LB) part and Predict-Forward (PF) part. The LB part concatenates the previous attention vector and the current hidden state h_t as the input of the attention module. It helps to embed visual information of the previous steps and suits human visual habit. PF part differs from traditional methods and serially predicts h_{t+1} and h_{t+2} according to h_t within the same time step. We directly regard h_{t+1} as an embedding of y'_{t+1} and pass it through the same parameters for h_t to predict h_{t+2}. This process generates two sequences denoted as seq_1 and seq_2, where seq_2 starts from the second word in the final sequence. In the training phase, we separately optimize seq_1 and $seq2$ according to the ground-truth. In the inference phase, we predict y'_t when $t \geq 2$ with $p1_t + \lambda p2_t$, and the first word y_1 depends on $p1_1$ itself ($p1$ denotes the predicted probability in seq_1, and $p2$ denotes the probability in seq_2).

We evaluate our proposed LB, PF and LBPF methods on MSCOCO Karpathy test split with both cross-entropy loss and CIDEr optimization. To verify the effectiveness of our methods, we employ Up-Down model [2] as the base decoder, and respectively implement LB, PF and LBPF model. With the detected feature pre-trained on Visual Genome dataset, all the three models show remarkable improvement over the base model, and the best LBPF model achieves BLEU-4 / METEOR / ROUGE-L / CIDEr / SPICE scores of 37.4 / 28.1 / 57.5 / 116.4 / 21.2 with cross-entropy loss and 38.3 / 28.5 / 58.4 / 127.6 / 22.0 with CIDEr optimization.

2. Related Work

A large number of methods have been proposed based on encoder-decoder frameworks [19, 14, 13] for image captioning. Oriol Vinyals *et al.* [27] proposed *show and tell* network, where the image was encoded into a feature vector by a pre-trained CNN and used as the first word embedding input for the language LSTM. Junhua Mao *et al.* [21] concatenated the image feature vector with each word embedding in order to maintain the visual information for later-generated words. Lisa Anne Hendricks *et al.* [3] separated visual information from language LSTM and only employed it before the logit layer. Kelvin Xu *et al.* [29] first introduced attention mechanism into caption generation tasks and also first initialized the hidden state of language LSTM with visual feature vector. Attention module has been proved to have huge improvements on image caption generation and thus applied to almost all recent methods [30, 8, 32, 11].

Attention-based methods for image captioning usually extract features with pre-trained CNN models on extra datasets. Famous image datasets like ImageNet [10] have a large number of images with labels for various common objects. Channel-wise features from pre-trained CNN models like VGG [24] and ResNet [12] show great representational capacity in object and scene identification. As newly released Visual Genome [15] dataset came out, detection-based encoder employed Fast R-CNN network to extract more explicit features for images. Peter Anderson *et al.* [2] pre-trained Fast R-CNN on Visual Genome dataset, and gathered the detected region vectors with high confidence coefficients as final features, which showed remarkable advantage over CNN pre-trained features. All these efforts focus on embedding more information on the encoded features, while ignoring the relevance of visual attention. Our proposed method instead considers the influence of the previous attention vector to the present one.

Besides attention mechanisms, the solution to accumulated error also played an important role in image captioning. Samy Bengio *et al.* [7] proposed scheduled sampling for sequence prediction with RNNs, which partially uses sampled y'_t to replace ground-truth y_t in the training phase. Beam search algorithm has been widely used in caption generation, as it reduces accumulated error caused by max-

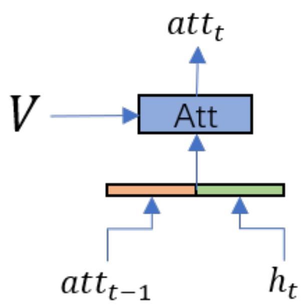

Figure 2. Look Back attention module. We concatenate previous attention result att_{t-1} and the current hidden state h_t together and feed it into Attention module to predict att_t.

sampling. Krueger *et al.* [16] put forward zoneout to regularize RNN, where each hidden state and memory cells chose to update their values or remain unchanged. Xinpeng Chen *et al.* [9] introduced auto-reconstructor network (ARnet) to regularize the transition between neighboring hidden states. However, all these methods are still stuck in the word-by-word generation process and have not considered predicting forward to decrease accumulated error caused by max-sampling. We propose to predict forward in both training and inference phase, and actively employ future information to adjust current selection.

3. Method

Like most existing methods, we employ encoder-decoder framework for image captioning. Given an image I, we first generate the detection feature $V = \{v_1, v_2, \cdots, v_k\}, v_i \in \mathbb{R}^D$ with pre-trained encoder model, and take the mean pooling vector $\bar{v}$ as the global information. Then we apply our LBPF method in the decoder architecture. In this paper, we employ classic Bottom-Up and Top-Down Attention model proposed by [2] as the base image captioning model. We first introduce Look-Back model in Sec 3.1 and then describe Predict-Forward model in Sec.3.2. In Sec.3.3, we explain how to combine these two models together as LBPF.

3.1. Look Back Model

Given k spatial image feature vectors $V = \{v_1, \cdots, v_k\} \in \mathbb{R}^{D\times k}$ and current hidden state $h_t \in \mathbb{R}^d$, conventional attention module employs the attention function f_{att} to calculate a weighted average vector att_t as:

$$att_t = f_{att}(V, h_t) \tag{1}$$

where each v_i of V is a D-dimensional representation of a detected region. In general, the calculated att_t is directly concatenated with h_t to predict the next word y'_{t+1}. However, the attention region should have visual coherence and att_t can contribute to later time steps.

We thus propose Look Back method, which helps to take the previous attended result into consideration. As illustrated in Figure.2, we bring in att_{t-1}, which is the previous attention result, and concatenate it with current hidden state h_t as the input of f_{att}. We denote $\circ$ as the concatenation operation, and we get new att_t as:

$$att_t = f_{att}(V, h_t \circ att_{t-1}) \tag{2}$$

We denote $H_t = h_t \circ att_{t-1}$, and then employ H_t to distinguish the importance of different feature vectors v_i and the weights are calculated as follows:

$$u_{i,t} = w_u \tanh(W_{vu}v_i + W_{hu}H_t) \tag{3}$$
$$\alpha_t = softmax(u_t) \tag{4}$$

where W_{vu}, W_{hu} and w_u are parameters in f_{att}, $\alpha_t = \{\alpha_{1,t}, \alpha_{2,t}, \cdots, \alpha_{k,t}\} \in \mathbb{R}^k$ is a k-dimensional vector which sums to 1. The final attention att_t is generated by:

$$att_t = \sum_{i=1}^{k} \alpha_{i,t} v_i \tag{5}$$

It is worth mentioning that, we treat att_{t-1} as previous attended regions and employ it for current attention generation. In our Look Back model, we simply employ the value of att_{t-1} and cut off the back propagation through it to the previous time step. The attention module may become too complicated and get hard to converge if all the gradients are accumulated together.

3.2. Predict Forward Model

In most existing methods of sequence generation, the current word embedding E_t of y_t is fed into the RNN-based architecture, and the next word y'_{t+1} is predicted at this time step. In the inference phase, y'_{t+1} depends heavily on y'_t and the probability of wrong sampling results in inevitable accumulated error on sequence generation. We hereby propose Predict Forward method to predict y'_{t+1} and y'_{t+2} in one time step to alleviate this problem.

As illustrated in Figure. 3, given h_t^1 from the attention LSTM and visual features V, we get att_t^1 with Att module and feed them together into LSTM2 (the language LSTM). Here we denote attention function as f_{att}, the mathematical operation of LSTM2 as F_2 and concatenation operation as $\circ$, then h_t^2 can be calculated as:

$$att_t^1 = f_{att}(V, h_t^1) \tag{6}$$
$$h_t^2 = F_2(h_t^1 \circ att_t^1, h_{t-1}^2) \tag{7}$$

where h_{t-1}^2 is the hidden state of LSTM2 from last time step. Normally, h_t^2 is fed into the logit layer and retrieve the

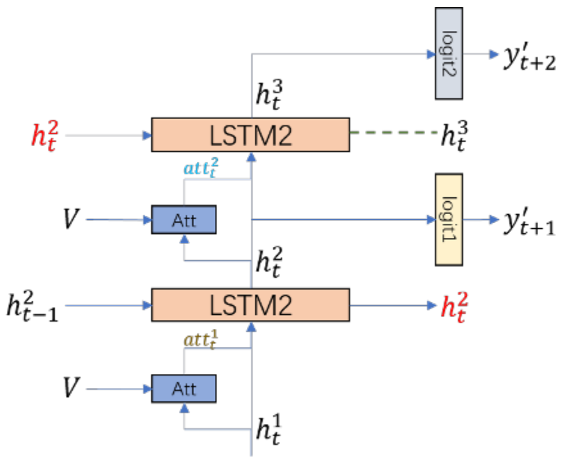

Figure 3. Predict Forward module. With current hidden state h_t^1 from the first LSTM, we operate the word generation twice with the same parameters in Attention module and the language *LSTM2*. First we generate att_t^1 and h_t^2, and then we move forward to predict h_t^3 by replacing h_t^1 with h_t^2 and feed it through *Att* and *LSTM2*. h_t^2 is employed both as the hidden state to generate h_t^3 and as the hidden state for *LSTM2* in next time step. h_t^2 and h_t^3 are fed into two independent logit layers to predict y'_{t+1} and y'_{t+2} respectively.

probability p_t^2 for y'_{t+1}, here we denote logit layer for p_{t+1} as $logit_1$, and calculate p_t^2:

$$p_t^2 = softmax(logit_1(h_t^2)) \tag{8}$$

As the training proceeds, h_t^2 tends to embed more precise information of word y_{t+1}, because it can be mapped to y_{t+1} by a simple logit layer. This offers us enough support to regard h_t^2 as a special embedding of y_{t+1}, and it is reasonable to predict the next word by employing history information and h_t^2.

As shown in Figure.3, in our proposed PF module, besides being fed into the logit layer, h_t^2 further go through *Att* and *LSTM2* and predict h_t^3. The same color indicates that the *Att* module and *LSTM2* are identical to which h_t^1 goes through. The main difference from the original method is that the hidden state of *LSTM2* is updated into h_t^2 while h_{t-1}^2 is employed for h_t^1. Then h_t^3 is generated by the following:

$$att_t^2 = f_{att}(V, h_t^2) \tag{9}$$

$$h_t^3 = F_2(h_t^2 \circ att_t^2, h_t^2) \tag{10}$$

next, h_t^3 is connected to another logit layer $logit_2$ to predict the probability p_t^3 for y'_{t+2} as:

$$p_t^3 = softmax(logit_2(h_t^3)) \tag{11}$$

In Figure.3, the dashed line connecting *LSTM2* and h_t^3 indicates that h_t^3 will not be passed to the next time step and is simply used to generate y'_{t+2}. On the contrary, h_t^2 is stored both for h_t^3 and h_{t+1}^2 in next time step. We denote the Predict Forward module as PF-LSTM unit for further explaination.

In training phase with cross-entropy loss, the PF-LSTM unit is executed for T steps where T refers to the length of ground-truth Y. This process will generate two predicted sequences: $Y^{1'} = \{y_1^{1'}, y_2^{1'}, \cdots, y_T^{1'}, EOS\}$ and $Y^{2'} = \{y_2^{2'}, y_3^{2'}, \cdots, y_T^{2'}, EOS, EOS\}$, where $Y^{1'}$ corresponds to h_t^2 and $Y^{2'}$ for h_t^3. The input of training phase starts from Begin-of-Sentence (BOS), which is normally zero vector, and the prediction sequence ends with End-of-Sentence (EOS). h_t^3 depends on h_t^2 and thus there is no $y_1^{2'}$ in $Y^{2'}$. We also abandon the last EOS in $Y^{2'}$ when h_t^2 educes EOS. Then we define the loss as:

$$loss_1 = -\frac{1}{T}\sum_{t=1}^{T} \log(p_t^2(y_t|y_{1:t-1})) \tag{12}$$

$$loss_2 = -\frac{1}{T-1}\sum_{t=2}^{T} \log(p_t^3(y_t|y_{1:t-2})) \tag{13}$$

$$loss = loss_1 + loss_2 \tag{14}$$

$loss_1 + loss_2$ treats $Y^{1'}$ and $Y^{2'}$ equally, which prompts the model to predict accurate y'_{t+1} and y'_{t+2}. To utilize this advantage, we combine the predicted probability of $y_t^{1'}$ and $y_t^{2'}$ together by:

$$p'_t = p_t^2 + \lambda p_{t-1}^3 \tag{15}$$

where p_t^2 is calculated from h_t^2, p_{t-1}^3 is retrieved from h_{t-1}^3 and λ is a trade-off coefficient to balance the importance of p_t^2 and p_{t-1}^3. During training process, both p_t^2 and p_{t-1}^3 approach to promote ground-truth y_{t+1}, and we combine them together to predict the next word. Beam search method is employed in our inference phase, and beam-size of 3 can basically guarantee the existence of optimal word from both of the two probability distributions. This operation enables the prediction of y'_{t+1} to not simply depend on sampled y'_t, but can also directly look to the generation result of the previous time step, which effectively reduces the accumulated error by wrong sampling.

For completeness of comparison with state-of-the-art work [2], we also apply our model with self-critical (SC) optimization on CIDEr [23]. Conventional self-critical learning on caption generation focuses on optimizing CIDEr score on the predicted sequence. The training process is to minimize the negative expected reward:

$$L_r(\theta) = -\mathbb{E}_{y_{1:T}\ p_\theta}[r(y_{1:T})] \tag{16}$$

where $r(y_{1:T})$ is the score function of the predicted sequence $y_{1:T}$ which is CIDEr in this paper and θ denotes the parameters of the network. Following the reinforcement learning based method proposed in [23], the gradient can be approximated as:

$$\nabla_\theta L_r(\theta) \approx -(r(y^s_{1:T}) - r(y^m_{1:T}))\nabla_\theta \log p_\theta(y^s_{1:T}) \quad (17)$$

where $y^s_{1:T}$ represents a sampled sequence and $y^m_{1:T}$ denotes the max-sampling sequence. The baseline is set to be $r(y^m_{1:T})$ to reflect the current capacity of the network.

In our PF model, we generate $Y^{1'}$ and $Y^{2'}$ simultaneously, and self-critical sequence training (SCST) certainly takes one sequence to be optimized. To apply our PF model with SCST, we select $Y^{1'}$ to be trained with CIDEr score optimization, and we force $Y^{2'}$ to approach $Y^{1'}$ by appending cross-entropy loss of $Y^{2'}$ with $Y^{1'}$ as the ground-truth. The cross-entropy loss L_c is calculated as:

$$L_c = -\frac{1}{T-1}\sum_{t=2}^{T} \log(p^3_t(y^{1'}_t | y^{1'}_{1:t-2})) \quad (18)$$

Then we combine self-critical loss L_r and cross-entropy loss L_c together and apply the total loss for SCST as:

$$L_{sc} = L_r + L_c \quad (19)$$

In the inference phase of SCST, we also employ the probability of $Y^{1'}$ and $Y^{2'}$ together, and follow Eq.(15) to generate the final sequence.

3.3. Look Back and Predict Forward Model

Look Back model focuses on attention module and take attended result of the last time step as input, which helps the model embed information from previous visual attention. Predict Forward model aims to predict next two words in one time step. We can combine these two models together by simply reserving two attention results (att^1_t and att^2_t) in one time step for the next one, as illustrated in Figure.1. By bringing in both previous and future information, our LBPF model is capable of predicting the next word by both contexts of the situation and visual information, which provides more details and guidance.

4. Experiments

4.1. Datasets

4.1.1 MSCOCO

We evaluate the performance of our proposed LBPF methods on MSCOCO 2014 caption dataset [18]. MSCOCO is the largest English image caption dataset containing 164,062 images. It is split with rate 2:1:1 for training, validation and test. In this paper, we employ the widely-used 'Karpathy' splits [13] for offline evaluation, which chooses 113,287 images for training and 5000 for validation and test each. Each image in MSCOCO is associated with at least 5 captions, and we select 5 per image for quantitative performance evaluation on BLEU [22], METEOR [6], ROUGE-L [17], CIDEr [26] and SPICE [1].

4.1.2 Visual Genome

Visual Genome (VG) [15] dataset contains 108K images with dense annotations. There are 5.4 million region descriptions and 42 for each image on average, where each description is a phrase with 1 to 16 words. These annotations include bounding boxes, classifications and attributes of main objects, and even the relationships among different instances are identified. Totally, the dataset contains 3.8 million object instances, 2.8 million attributes and 23 million relationships. In this paper, we employ detected feature pre-trained [20] on VG dataset.

4.2. Implementation Details

4.2.1 Encoder and Features

We employ detected vectors proposed by [2] as features, which are generated by pre-trained Fast R-CNN model [20] on VG dataset. In pre-training process, only object and attribute data is employed. 98K images are split for training and 5K for validation and test each [2, 20].

The annotations of objects and attributes are phrases with 1 to 16 words, thus data cleaning is performed over the dataset. Fast R-CNN model is trained on this dataset for multiple instances detection, and we only preserve ROI (region of interest) pooling vectors with the size of 2048 whose confidence coefficient is at least 0.2 to guarantee the expressive ability of our selected features. We also set the selected number for each image to be 10-100 to balance the distribution among different images.

4.2.2 Decoder

Our base decoder employs the Bottom-Up and Top-Down Attention model. For fair comparison with the previous work, we directly employ the same hyper-parameters which are proposed in [2]. We use hidden units of 1000 in both LSTMs and set hidden units of the atttention module as 512. The size of input word embedding is also 1000. We employ Adam optimizer with the learning rate initialized as $5e^{-4}$ and decaying by 0.8 exponentially every 3 epoches. Due to the limit of the graphic memory of GeForce GTX 1080Ti, we set the batch size to 64. For self-critical learning, we start from the best saved model during optimization using cross-entropy loss. The learning rate for SCST starts from $5e^{-5}$ and decays by rate 0.1 every 50 epoches.

Models	Cross-Entropy Loss						CIDEr Opitimization					
	B-1	B-4	METEOR	ROUGE-L	CIDEr	SPICE	B-1	B-4	METEOR	ROUGE-L	CIDEr	SPICE
SCST:Att2in [23]	-	31.3	26.0	54.3	101.3	-	-	33.3	26.3	55.3	111.4	-
SCST:Att2all [23]	-	30.0	25.9	53.4	99.4	-	-	34.2	26.7	55.7	114.0	-
ARnet [9]	74.0	33.5	26.1	54.6	1.034	19.0	-	-	-	-	-	-
Up-Down [2]	77.2	36.2	27.0	56.4	113.5	20.3	79.8	36.3	27.7	56.9	120.1	21.4
Ours: LB	77.4	36.7	27.6	57.0	114.3	20.8	79.6	37.7	28.4	58.1	124.4	21.8
Ours: PF	77.4	37.0	27.9	57.2	115.7	20.9	**80.7**	38.3	28.4	58.4	126.9	21.9
Ours: LBPF	**77.8**	**37.4**	**28.1**	**57.5**	**116.4**	**21.2**	80.5	**38.3**	**28.5**	**58.4**	**127.6**	**22.0**

Table 1. Performance comparisons on MSCOCO Karpathy test split using cross-entropy loss and CIDEr optimization, respectively. Our baseline is Up-Down model proposed in [2], the existing state-of-the-art model employing bottom-up attention mechanisms. With the pre-trained features, we apply our Look Back (LB) method, Predict Forward (PF) method and the combined Look Back and Predict Forward (LBPF) method on the base decoder. Testing results show that our methods bring huge boost with both cross-entropy loss / CIDEr optimization and outperform all the previous work with simple feature. In the table, B-1 and B-4 represent BLEU-1 and BLEU-4.

λ	PF Model					
	B-1	B-4	M	R	C	S
0.1	76.5	36.3	27.7	56.7	112.7	20.6
0.3	76.9	36.6	27.7	57.0	113.2	20.6
0.5	77.0	36.7	27.7	57.0	**113.8**	20.8
0.7	77.2	36.7	27.7	57.0	113.7	20.8
0.9	77.4	36.6	27.6	56.9	113.5	20.8
	LBPF Model					
λ	B-1	B-4	M	R	C	S
0.1	76.7	36.7	27.9	57.0	113.9	20.7
0.3	76.9	37.0	27.9	57.1	114.1	20.8
0.5	77.2	36.9	27.9	57.2	**114.3**	20.8
0.7	77.2	36.8	27.8	57.1	114.2	20.9
0.9	77.3	36.6	27.7	56.9	113.8	20.9

Table 2. Performances evaluation of our PF and LBPF models with different λ on MSCOCO Karpathy validation dataset where λ is the trade-off parameter in Eq.(15). B-1 / B-4 / M / R / C / S refers to BLEU-1 / BLEU-4 / METEOR / ROUGE-L / CIDEr / SPICE scores respectively. This table shows the performance trained with cross-entropy loss.

4.3. Captioning Evaluation Results

4.3.1 Model Selection with λ

In our proposed PF and LBPF methods, we combine two generated sequences together with a trade-off parameter λ in Eq.(15) and we do model selection through it. During the training process, we noticed that the converged loss of $Y^{2'}$ is slightly larger than that of $Y^{1'}$, thus we set λ from 0 to 1 to put more emphasis on $Y^{1'}$. In our experiments, we employ beam search method to sample output sequence. Beam size of 3 can effectively cover the optimal words in both sequences and setting λ less than 1 tends to select the second promoted word in $Y^{1'}$. The results of PF and LBPF models trained with cross-entropy loss are shown in Table.2.

Recently, CIDEr has become the most important score which is well-accepted to best reflect the information and smoothness of the sentences. The results in Table.2 show that both PF and LBPF models achieve their best performances at $\lambda = 0.5$. This is reasonable, because if λ is too small the sequence $Y^{2'}$ might not contribute to the final sampling, and on the contrary, if λ is too large the sequence $Y^{2'}$ might lead the sampling, while the training loss shows that $Y^{1'}$ converges slightly better than $Y^{2'}$. Thereafter, we eventually set $\lambda = 0.5$ and test the performance over Karpathy test split.

For self-critical learning, we perform the same model selection procedure through λ and eventually select $\lambda = 0.3$ for PF model and $\lambda = 0.5$ for LBPF model. The results are similar to those in Table. 2, and not shown here.

4.3.2 Evaluation Results

After model selection on the validation dataset according to λ, we evaluate our methods on MSCOCO Karpathy test split which contains 5000 images. The LB, PF and LBPF model are trained independently from randomly initialization, and are validated every 0.5 epoch. The weights showing best performance on validation dataset are eventually selected to perform testing, and the validation procedure also employs beam searching methods with the beam size of 3. All the three models are trained with both cross-entropy loss and CIDEr optimization, where the self-critical learning starts from the best weights stored in the training procedure with cross-entropy loss. The evaluation results are shown in Table. 1.

Our baseline is the Up-Down model proposed in [2], and we also employ the network as our base decoder. The test results in Table. 1 show that all the three methods bring with noteworthy improvement over the baseline. The LB model improves 0.8 percent on CIDEr score with cross-entropy loss, and 4.3 percent with CIDEr optimization. The

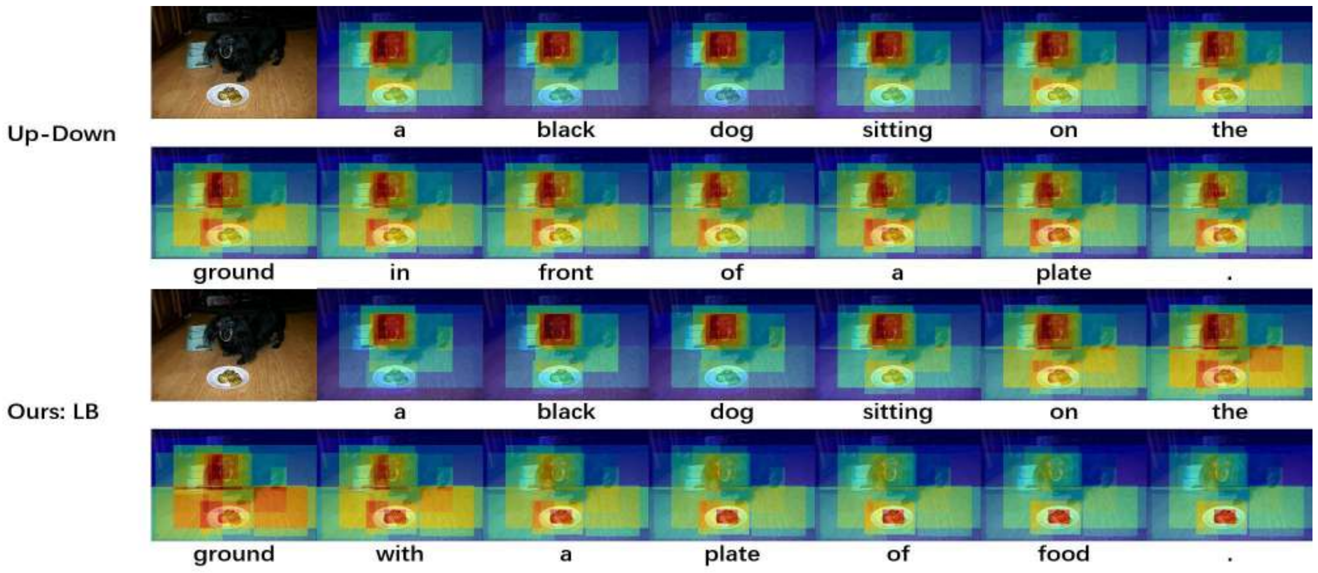

Figure 4. Qualitative analysis of attention results using Up-Down [2] model and our LB model.

Models	Cross-Entropy Loss					
	B-1	B-4	M	R	C	S
Up-Down [2]	76.5	36.3	27.7	56.7	112.7	20.6
PF $\lambda = 0$	76.7	36.6	27.8	56.9	114.4	20.8
PF $\lambda = 0.5$	77.4	37.0	27.9	57.2	115.7	20.9
LBPF $\lambda = 0$	76.7	36.6	27.9	56.9	114.8	20.8
LBPF $\lambda = 0.5$	77.8	37.4	28.1	57.5	116.4	21.2

Models	CIDEr Optimization					
	B-1	B-4	M	R	C	S
Up-Down [2]	79.8	36.3	27.7	56.9	120.1	21.4
PF $\lambda = 0$	80.4	37.9	28.4	58.4	126.5	21.9
PF $\lambda = 0.3$	80.7	38.3	28.4	58.4	126.9	21.9
LBPF $\lambda = 0$	80.4	38.1	28.5	58.4	127.1	21.9
LBPF $\lambda = 0.5$	80.5	38.3	28.5	58.4	127.6	22.0

Table 3. comparison of our proposed PF and LBPF model with best λ value and $\lambda = 0$. B-1 / B-4 / M / R / C / S refers to BLEU-1 / BLEU-4 / METEOR / ROUGE-L / CIDEr / SPICE scores. This table shows the performance trained both with cross-entropy loss and CIDEr optimization.

PF method shows better performance over LB method and outperforms the baseline 2.2 / 6.8 percent with two training methods respectively. LBPF method combines LB and PF together, and achieves BLEU-4 / CIDEr / SPICE scores of 37.4 / 116.4 / 21.2 with cross-entropy loss, and 38.3 / 127.6 / 22.0 with CIDEr optimization. The results are persuasive to show the effectiveness of our proposed model as we outperform the baseline over every quantitative score by great promotion. With the simple detected feature, we even approach the result in the latest work proposed in [31], which employs complicated semantic and spacial information.

The qualitative analysis over the attention results for our LB method is shown in Figure.4. For the same image, Up-Down model only consider the main aera while our LB model show a smooth transition from "dog" to "gound", "plate" and "food". The caption shows this improvement can also bring more information of the image like "food".

Some examples of our generated captions with CIDEr optimization are shown in Table. 4. From the table we can see, all our three models retrieve most important information from the image, and models with PF method can observe more precise visual information from the image like "car", "couch" and "sitting". Some information we generated does not even exist in the ground-truth, which in fact appears in the image like the "coach" in the second image.

4.3.3 PF method Analysis

Our PF model generates two sequences ($Y^{1'}$ and $Y^{2'}$) together. In the training phase with cross-entropy loss, we respectively calculate the losses of the two sequences and add them together as the final loss in Eq.(14). In self-critical training process, we optimize $Y^{1'}$ with CIDEr score and force $Y^{2'}$ to be close to $Y^{1'}$ by summing reward loss and cross-entropy loss in Eq.(19). In the training phase, we employ a trade-off parameter λ to balance the contributions of thte two sequences. In Table.3, we show the comparison of our selected model with the best λ value and with $\lambda = 0$.

Evaluation results show that, even with $\lambda = 0$, our proposed PF and LBPF models still show remarkable improvement over the baseline Up-Down model. This comparison indicates that, besides predicting two words at one time

Images	Our Captions	Ground Truth Captions
	LB:a wooden cutting board topped with a pizza PF:a pizza sitting on top of a wooden cutting board LBPF:a couple of slices of pizza on a wooden cutting board	1.A vegetarian pizza is half eaten on a pizza holder 2.A couple of pieces of pizza with vegetable slices on them. 3.A wooden pan serving tray with a pizza on it 4.A pizza on a cutting board is half gone 5.A Pizza is nearly finished with only three pieces left
	LB:a man holding a nintendo wii game controller PF:a man sitting on a couch holding a wii game controller LBPF:a man sitting on a couch holding a wii game controller	1.A young man playing a game with a remote controller 2.A man playing a game with a remote controller 3.A man is playing the wii while his face is being stroked 4.A hand strokes the face of a man playing a video game 5.A guy sitting down playing a video game
	LB:a person taking a picture of a dog in a rear view mirror PF:a person taking a picture of a dog in a car rear view mirror LBPF:a woman taking a picture of a dog in a rear view mirror of a car	1.A man is taking a picture in a rear view mirror 2.A woman taking a picture of her and a dog in a mirror 3.The woman in the mirror is taking a picture of herself and the dog 4.A hand strokes the face of a man playing a video game 5.A woman sitting in the pasenger seat of a car with a dog in her lap and a camera in her hand

Table 4. Generated caption examples of our three models trained with CIDEr optimization. The results show that, all our three approaches retrieve most important information from the image and models with PF method can observe more precise information like highlighted "car", "couch", "sitting" and so on. (We rotate the image in line 2 to adapt to the table size)

Models	Parameters(M)
Up-Down	49.71
LB	50.20
PF	58.77
LBPF	59.26

Table 5. Parameter counts for Up-Down model, LB model, PF model and LBPF model, which is shown in M.

step, the PF method also helps the network embed more precise information on h_t^2. Models with the best λ exceeding ones with $\lambda = 0$ show that combination with λ better aggregates the information within two sequences.

4.3.4 Complexity comparison over decoder

As illustrated in Figure.2 Our proposed LB method simply increases the input size of the Attention module, and PF method adds another logit layer into the network as shown in Figure.3. Table.5 shows the total parameters in the four models, including the base Up-Down model. The parameter numbers of LB model increases 0.99 percent, which lays on the attention module. PF method brings with 18% increment on the base model. All the increased parameter lays on the logit layer, which has no change on the main network structure. After the combination of LB and PF methods, LBPF model is totally 18.99 percent larger than the original architecture.

This comparison shows that, we achieve remarkable improvements over the base model while bringing only small increment in the parameter numbers. Our LBPF method can effectively help the base architecture embed more information and perform better on the caption generation task.

5. Conclusion

We propose Look Back method to embed previous visual information and Predict Forward approach to look into future for image captioning task. Our LB method takes attention value from the previous time step into the input of the current attention module which satisfies the visual coherence of human beings. PF approach generates two next words in one time step, which utilizes linguistic coherence and integrates future information. In the inference phase, the two generated probabilities are combined together to predict the current word. LBPF approach combines LB and PF together and achieves remarkable performance gains over the state-of-the-art method on MSCOCO dataset.

All the three approaches can be easily applied on most attention-based encoder-decoder models for image captioning. When applying them on a base decoder Up-Down architecture, LBPF model achieves BLEU-4 / CIDEr / SPICE scores of 37.4 / 116.4 / 21.2 with cross-entropy loss and 38.3 / 127.6 / 22.0 with CIDEr optimization.

Our proposed PF method can be intuitively applied to most sequence generation task like machine translation. In our future work, we will further explore the potential of PF method in both network structure and application fields.

6. Acknowledgement

This paper is supported by NSFC (No. 61772330, 61533012, 61876109), the Basic Research Project of Innovation Action Plan (16JC1402800), the advanced research project (No.61403120201), Shanghai authentication key Lab. (2017XCWZK01), Technology Committee the interdisciplinary Program of Shanghai Jiao Tong University (YG2015MS43), and AI-Lab VS Team, Bytedance.

References

[1] Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. Spice: Semantic propositional image caption evaluation. In *ECCV*, pages 382–398. Springer, 2016.

[2] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *CVPR*, volume 3, page 6, 2018.

[3] Lisa Anne Hendricks, Subhashini Venugopalan, Marcus Rohrbach, Raymond Mooney, Kate Saenko, and Trevor Darrell. Deep compositional captioning: Describing novel object categories without paired training data. In *CVPR*, pages 1–10, 2016.

[4] Jimmy Ba, Ruslan R Salakhutdinov, Roger B Grosse, and Brendan J Frey. Learning wake-sleep recurrent attention models. In *NIPS*, pages 2593–2601, 2015.

[5] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[6] Satanjeev Banerjee and Alon Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *ACL Workshop*, pages 65–72, 2005.

[7] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *NIPS*, pages 1171–1179, 2015.

[8] Long Chen, Hanwang Zhang, Jun Xiao, Liqiang Nie, Jian Shao, Wei Liu, and Tat-Seng Chua. Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning. In *CVPR*, pages 6298–6306. IEEE, 2017.

[9] Xinpeng Chen, Lin Ma, Wenhao Jiang, Jian Yao, and Wei Liu. Regularizing rnns for caption generation by reconstructing the past with the present. In *CVPR*, June 2018.

[10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255. Ieee, 2009.

[11] Hao Fang, Saurabh Gupta, Forrest Iandola, Rupesh K Srivastava, Li Deng, Piotr Dollár, Jianfeng Gao, Xiaodong He, Margaret Mitchell, John C Platt, et al. From captions to visual concepts and back. In *CVPR*, pages 1473–1482, 2015.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[13] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *CVPR*, pages 3128–3137, 2015.

[14] Jonathan Krause, Justin Johnson, Ranjay Krishna, and Li Fei-Fei. A hierarchical approach for generating descriptive image paragraphs. In *CVPR*, pages 3337–3345. IEEE, 2017.

[15] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *IJCA*, 123(1):32–73, 2017.

[16] David Krueger, Tegan Maharaj, János Kramár, Mohammad Pezeshki, Nicolas Ballas, Nan Rosemary Ke, Anirudh Goyal, Yoshua Bengio, Aaron Courville, and Chris Pal. Zoneout: Regularizing rnns by randomly preserving hidden activations. *arXiv preprint arXiv:1606.01305*, 2016.

[17] Chin-Yew Lin and Franz Josef Och. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *ACL*, page 605. Association for Computational Linguistics, 2004.

[18] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[19] Jiasen Lu, Caiming Xiong, Devi Parikh, and Richard Socher. Knowing when to look: Adaptive attention via a visual sentinel for image captioning. In *CVPR*, volume 6, page 2, 2017.

[20] Ruotian Luo, Brian Price, Scott Cohen, and Gregory Shakhnarovich. Discriminability objective for training descriptive captions. *arXiv preprint arXiv:1803.04376*, 2018.

[21] Junhua Mao, Wei Xu, Yi Yang, Jiang Wang, Zhiheng Huang, and Alan Yuille. Deep captioning with multimodal recurrent neural networks (m-rnn). *ICLR*, 2015.

[22] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *ACL*, pages 311–318. Association for Computational Linguistics, 2002.

[23] Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jarret Ross, and Vaibhava Goel. Self-critical sequence training for image captioning. In *CVPR*, volume 1, page 3, 2017.

[24] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*, 2015.

[25] Richard Socher, Andrej Karpathy, Quoc V Le, Christopher D Manning, and Andrew Y Ng. Grounded compositional semantics for finding and describing images with sentences. *ACL*, 2(1):207–218, 2014.

[26] Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. In *CVPR*, pages 4566–4575, 2015.

[27] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *CVPR*, pages 3156–3164, 2015.

[28] Zhilin Yang Ye Yuan Yuexin Wu and Ruslan Salakhutdinov William W Cohen. Encode, review, and decode: Reviewer module for caption generation. *NIPS*, 2016.

[29] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*, pages 2048–2057, 2015.

[30] Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alex Smola. Stacked attention networks for image question answering. In *CVPR*, pages 21–29, 2016.

[31] Ting Yao, Yingwei Pan, Yehao Li, and Tao Mei. Exploring visual relationship for image captioning. In *ECCV*, 2018.

[32] Quanzeng You, Hailin Jin, Zhaowen Wang, Chen Fang, and Jiebo Luo. Image captioning with semantic attention. In *CVPR*, pages 4651–4659, 2016.