

Vocabulary-Wide Credit Assignment for Training Image Captioning Models

Han Liu, Shifeng Zhang, Ke Lin, Jing Wen, Jianmin Li, Xiaolin Hu, *Senior Member, IEEE*

Abstract—Reinforcement learning (RL) algorithms have been shown to be efficient in training image captioning models. A critical step in RL algorithms is to assign credits to appropriate actions. There are mainly two classes of credit assignment methods in existing RL methods for image captioning, assigning a single credit for the whole sentence and assigning a credit to every word in the sentence. In this paper, we propose a new credit assignment method which is orthogonal to the above two. It assigns every word in vocabulary an appropriate credit at each generation step. It is called vocabulary-wide credit assignment. Based on this we propose a Vocabulary-Critical Sequence Training (VCST). VCST can be incorporated into existing RL methods for training image captioning models to achieve better results. Extensive experiments with many popular models validated the effectiveness of VCST.

Index Terms—Image Captioning, Reinforcement Learning, Artificial Intelligence

I. INTRODUCTION

GIVEN an image, automatically generating a description using natural language is called image captioning. This task integrates techniques in both computer vision (CV) and natural language processing (NLP), which has many applications. It has been studied for many years [1], [2], [3], [4]. In recent years, deep learning [5], [6], [7], [8] has made great progress on this task. Many excellent models have been proposed including the Show-and-Tell model [5], the Att2in model [7] and the Up-Down model [8]. Most of them follow the encoder-decoder framework [6]. The encoder is usually a convolutional neural network (CNN) that extracts image features from the input, and the decoder is usually a long short term memory (LSTM) [9] that decodes the image features to generate sentences.

To train the models, earlier methods adopted the Teacher-Forcing method, which is to maximize the likelihood of the next ground-truth word given the previous ground-truth words. This strategy leads to so called “exposure bias” [10] due to the absence of ground-truth words in previous steps during the generation of a sentence in inference. In addition, the maximum likelihood does not necessarily correspond to

This work was supported in part by National Key Research and Development Program of China under Grant 2017YFA0700904, National Natural Science Foundation of China under Grant U19B2034, Grant 61836014, and Grant 61620106010, and a grant from Samsung Research China, Beijing. (Corresponding author: Xiaolin Hu)

H. Liu, S. Zhang, J. Wen, J. Li and X. Hu are with the State Key Laboratory of Intelligent Technology and Systems, Beijing National Research Center for Information Science and Technology, Department of Computer Science and Technology, Institute for Artificial Intelligence, Tsinghua University, Beijing 100084, China (e-mail: xlihu@tsinghua.edu.cn)

K. Lin is with the Samsung Research China, Beijing

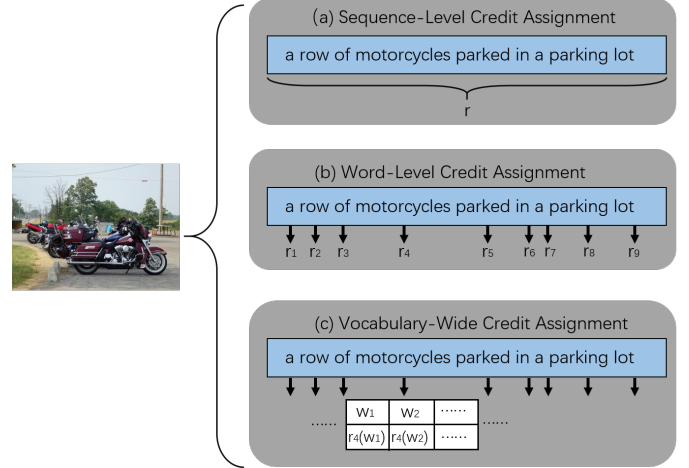


Fig. 1. Three kinds of credit assignments. (a) Sequence-level credit assignment assigns a single credit to the whole sentence. (b) Word-level credit assignment assigns different credits to all words in the sentence. (c) Vocabulary-wide credit assignment assigns different credits to all words in vocabulary at every generation step.

the maximum evaluation metrics developed in NLP such as CIDEr [11] and BLEU [12], which are commonly used to measure the performance of models on this task.

To solve both problems, reinforcement learning algorithms (RL) were introduced to this task. Ranzato et al. [13] firstly introduced REINFORCE to sequence generation tasks. Following that work, Rennie et al. [14] improved the baseline of REINFORCE and proposed Self-Critical Sequence Training (SCST), which has become the most popular RL algorithm for image captioning nowadays [8], [15], [16], [17]. Zhang et al. [18] used actor-critic algorithm in training image captioning model to replace sequence-level advantage with word-level advantage and Gao et al. [19] improved actor-critic algorithm by estimating state-action value with Monte-Carlo rollouts. With these RL methods, significant improvement has been achieved.

Among these RL methods, some assign a credit for the whole generated sentence [13], [14] and others assign word-level credit for each word in the generated sentence [18], [19], [20]. Both sequence-level and word-level credit assignments focus on the generated words. However, in these sequence-level and word-level methods, the rewards are usually given to evaluate the whole sentence, which care more about the structure of sentences than local connections of words.

To explore the effectiveness of local connections of words in the training of image captioning, we propose a *vocabulary-*

wide credit assignment that assigns credit to every word in the vocabulary at each generation step. Figure 1 shows these three kinds of credit assignment methods.

In vocabulary-wide credit assignment, rewards are given to estimate how the score would change with replacing one word, which focus on the local connections of words. From another point of view, the estimation of score variation propels the model to roughly explore these generation trajectories with word replacement, which helps to search for the best generation trajectory.

To achieve vocabulary-wide credit assignment, we design a “vocabulary-critic”. At every step of word generation in an image captioning model, the vocabulary-critic estimates the metric improvement induced by replacing the current word with every other word in the vocabulary, then the metric improvement is used as a reward to optimize the policy of word generation. With the vocabulary-wide credit assignment, we propose a training method named Vocabulary-Critical Sequence Training (VCST), which can be incorporated into existing RL methods to improve their performance.

Contributions To sum up, our contributions are as follows:

- We propose a novel RL method in image captioning called “Vocabulary-Critical Sequence Training” (VCST). In VCST, a vocabulary-critic module based on word replacement is designed to assign every word in the vocabulary a different credit at each generation step.
- To reduce the time cost of metrics calculation in VCST, we propose fast algorithms for computing the CIDEr-D metric and the BLEU metric. With our fast computing methods, the time costs for computing CIDEr-D and BLEU-4 are reduced by 97% and 72%, respectively.
- On the MSCOCO Karpathy test set, the combination of VCST and Self-Critical Sequence Training (SCST) [14] improved the CIDEr-D metric from 121.9 to 125.0 on a single Up-Down model [8], from 127.8 to 129.1 on a single SGAE model [16] and from 132.0 to 132.4 on a single X-LAN model [21]. Moreover, the experiments showed that VCST could also boost the performance of the Actor-Critic Sequence Training (ACST) [18].

II. RELATED WORK

In earlier years image captioning models were usually template-based [2], [3] or retrieval-based [1], [4]. With the development of deep learning, encoder-decoder framework [5] was introduced to this task where the encoder extracts features from images and the decoder generates natural language sentences from image features. Different models may use different encoders or decoders, which are usually neural networks [5], [7], [8], [22]. Based on the encoder-decoder architecture, many progresses have been made. Recent progress on this topic can be divided into two branches: designing new architectures and proposing better training methods. We review the researches in these two branches.

A. Model Architectures

Vinyals et al. [5] first introduced encoder-decoder framework to image captioning. They used a CNN pre-trained on

image classification task as encoder to extract image features and used an LSTM as decoder to generate captions. To improve the decoder in encoder-decoder framework, Xu et al. [7] introduced dynamic spatial attention and proposed soft attention and hard attention mechanisms. Following these works, many improvements have been made. Lu et al. [23] proposed adaptive attention which can automatically decided when to rely on image regions and when to rely on the language model in sequence generation. Chen et al. [24] proposed SCA-CNN which incorporated spatial and channel-wise attention in the encoder. Anderson et al. [8] improved the attention module with a novel 2-layer LSTM and replaced the encoder pre-trained on image classification task with a Faster R-CNN [25] pre-trained on the object detection task of Visual Genome [26] dataset. Yao et al. [15] proposed the GCN-LSTM model, which used a graph convolutional network to integrate the spatial and semantic relationship into the encoder. Yang et al. [16] proposed the SGAE model, which added a language prior to the encoder by training a shared dictionary Huang et al. [17] proposed the AoANet model, which extends attention mechanisms to determine the relevance between attention results and queries.

Recently transformer [27] was introduced to this task and obtained good results. Yu et al. [28] designed a Multimodal Transformer for image captioning and introduced multi-view visual features to their model. Pan et al. [21] modified the attention block of transformer then proposed the X-Linear Attention Networks. Cornia et al. [29] proposed a Meshed-Memory network, in which memory was introduced to the transformer encoder and a meshed-like connectivity was used to exploit both low-level and high-level features.

B. Training Methods

Earlier models [5], [7], [30], [23] adopted the Teacher-Forcing training method [10], [13], which uses the ground-truth captions as the input and trains the model by maximizing the likelihood of the next ground-truth word. However, the “exposure bias” [10] and the mismatch between training targets and evaluation metrics limit its performance.

A lot of study have been made to address these problems. Bengio et al. [10] proposed Scheduled Sampling which combines the Teacher-Forcing strategy and the Free-Running strategy. Lamb et al. [31] proposed Professor Forcing which used adversarial domain adaptation to train recurrent networks. Although these methods addressed the “exposure bias” to some extent, the mismatch between targets and evaluation metrics still remains.

Recently RL was shown to be effective in improving performance in terms of NLP evaluation metrics. Ranzato et al. [13] proposed the MIXER algorithm, the first RL method in sequence generation. The algorithm mixes maximum likelihood estimation (MLE) and policy gradient (PG) and gained large improvement. Later, many RL methods [20], [14], [19], [18] were introduced to image captioning. Liu et al. [32] proposed PG-SPIDER method which separated the MLE training and PG training and combined the SPICE metric and the CIDEr-D metric. Zhang et al. [18] introduced actor-critic structure to image captioning and proposed Actor-Critic Sequence Training.

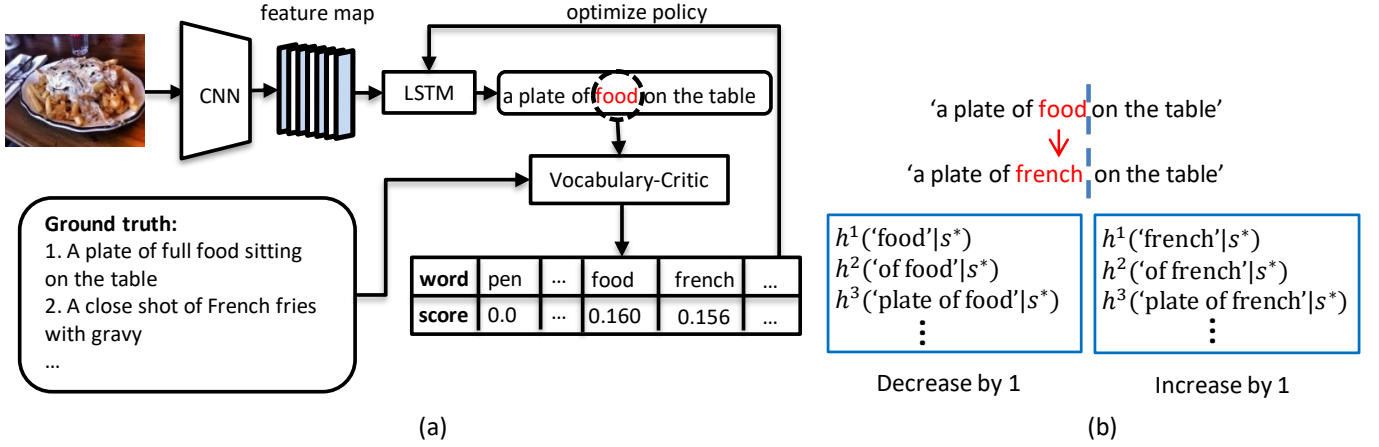


Fig. 2. Illustration of the proposed method. (a) The sketch of image captioning model with VCST. The vocabulary-critic takes in the ground-truth captions and the generated caption, then evaluates all the words in vocabulary at each generation step in training. (b) Change of the frequencies of n -grams after replacing a word ‘food’ with ‘french’. Note that only the words on the left side of the dashed line are considered according to the assumption stated in the text.

Rennie et al. [14] proposed Self-Critical Sequence Training, which uses the score of the sentence generated by the model with greedy policy as the baseline of RL. Gao et al. [19] reformulated the advantage and extended one step to n -step. These RL methods significantly improved the performance of image captioning models.

III. METHODS

A. Background Knowledge

1) *Self-Critical Sequence Training*: Self-Critical Sequence Training (SCST) is a general training method in image captioning. In SCST, the captioning model is treated as an “agent”, and the external information like inputs and ground-truth constitutes the “environment”. At each step t , the agent performs an “action” to generate a word w_t^* , and then updates its own hidden state. After sentence generation, a metric Ψ (e.g., CIDEr-D, BLEU) takes in the generated sentence $s^* = \{w_1^*, \dots, w_T^*, w_{T+1}^*\}$ and the ground-truth sentences set $\Xi = \{\xi_1, \dots, \xi_m\}$ to generate a reward $r(s^*) = \Psi(s^*|\Xi)$, where T denotes the length of the sentence s^* , w_{T+1}^* denotes “EOS” that indicates the end of generation, and ξ_k denotes the k -th sentence in the ground-truth set Ξ .

Let θ denote the parameters of the model and p_t^θ denote the probability distribution of word generation at step t . The probability distribution of a sentence $s = \{w_1, \dots, w_T, w_{T+1}\}$ is $p^\theta(s) = \prod_t p_t^\theta(w_t|w_1, \dots, w_{t-1})$. The target of SCST is to minimize the negative expected reward:

$$L_{scst}(\theta) = -\mathbb{E}_{s \sim p^\theta}[r(s)]. \quad (1)$$

According to the REINFORCE algorithm [33], for each training case the loss of SCST can be approximated as the follows:

$$L_{scst}(\theta) = -(r(s^*) - r(\hat{s})) \log p^\theta(s^*), \quad (2)$$

where s^* is the sentence sampled from the distribution $p^\theta(s)$, and \hat{s} is the sentence generated by the same model with the greedy policy.

2) *Actor-Critic Sequence Training*: Actor-Critic Sequence Training (ACST) consists of two parts: a policy network and a value network. The policy network is a captioning model for generating captions and the value network is a sequence model such as LSTM for predicting values of each state. Let θ denote the parameters of policy network, ζ denote the parameters of value network, $s_t^* = \{w_1^*, \dots, w_t^*\}$ denote the sentence composed of the first t words, $V_\zeta(s_t^*)$ denote the value of s_t^* predicted by the value network. The losses of value network L_v and policy network L_p can be written as follows:

$$L_v(\zeta) = \sum_{t=0}^T \|\gamma^{T-t} r(s^*) - V_\zeta(s_t^*)\|^2, \quad (3)$$

$$L_p(\theta) = - \sum_{t=0}^T A(s_t^*, w_{t+1}^*) \log p_t^\theta(w_{t+1}^*|s_t^*), \quad (4)$$

where

$$A(s_t^*, w_{t+1}^*) = \gamma^{T-t} r(s^*) - V(s_t^*). \quad (5)$$

γ is a discount factor and we set γ to 1 in our experiments. The loss of ACST is the combination of L_v and L_p :

$$L_{acst}(\zeta, \theta) = L_p(\theta) + \beta L_v(\zeta), \quad (6)$$

where β is a weighting coefficient.

B. Vocabulary-Wide Credit Assignment With Vocabulary-Critic

As we discussed in Section I, vocabulary-wide credit assignment means that we need to assign a reasonable score to every word in the vocabulary Ω at every generation step, which is a heavy work. To achieve vocabulary-wide credit assignment, we design a “vocabulary-critic” based on word-replacement.

Assuming that the model generates a sentence $s^* = \{w_1^*, \dots, w_t^*, \dots, w_T^*, w_{T+1}^*\}$, for a word w_t^* at step t , we calculate its score by estimating the improvement on the NLP metric induced by replacing the word w_t^* generated at

step t with w'_t . If we replace the t -th word w_t^* in s^* with w'_t and continue the generation to form the new sentence $s' = \{w_1^*, \dots, w_{t-1}^*, w'_t, \dots, w_{T'}^*, w_{T'+1}^*\}$, the score of w'_t at the time t is

$$r_t(w'_t|s^*) = \Psi(s'|\Xi) - \Psi(s^*|\Xi). \quad (7)$$

We call this strategy the *full evaluation* strategy. Let $|\Omega|$ denote the size of the vocabulary and L denote the maximum length of generated sentences. To calculate (7) for every word in Ω at every step t , this strategy entails $O(|\Omega|L)$ times of LSTM feedforward processes (from the first word to the last word). Note that $|\Omega| \sim 10^4$ and $L \sim 20$ in a typical image captioning task. On a mainstream server with one GTX Titan X GPU (memory 12GB), an LSTM feedforward process takes about 50ms on average. Training one epoch of the MSCOCO Karpathy training images will take about 110 days with batchsize 50. The full evaluation strategy is computationally prohibitive.

The main reason for the high computational cost is that for every word in Ω we need to perform an LSTM feedforward process and a metric calculation process, because once w_t^* in s^* changes, all subsequent words may change. To reduce the computational cost, we propose to perform the LSTM feedforward process once by assuming that subsequent words do not change after w_t^* changes. In this way, only one sentence is generated and the number of LSTM feedforward processes is reduced from $O(|\Omega|L)$ to $O(L)$. Of course the score obtained for each word is a rather rough approximation of that defined in (7), but our experiments show that the strategy is both effective and efficient.

Note that the most commonly used NLP metrics such as BLEU and CIDEr-D are determined by n -gram frequency and the length of caption. More specifically, the score in (7) can be interpreted as how the replacement influences the n -gram frequency. Let $h^n(x|s)$ denote the times that n -gram x occurs in sentence s and $\mathbf{h}^n(s)$ denote the vector composed of those occurring times: $\mathbf{h}^n(s) = [h^n(x^1|s), h^n(x^2|s), \dots]$ where x^i is the i -th n -gram. We rewrite Ψ as a function of $\mathbf{H}(s)$ and $l(s)$:

$$\Psi(s|\Xi) = \psi(\mathbf{H}(s), l(s)|\Xi), \quad (8)$$

where $\mathbf{H}(s) = \{\mathbf{h}^1(s), \dots, \mathbf{h}^N(s)\}$ and $l(s)$ is the length of s . N is the maximum n in calculating metrics. It indicates that the reward is affected by the change of the n -grams (Figure 2b).

According to the above stated assumption, at step t , we only consider the change of the reward of the sentence s^* by replacing w_t^* with w'_t . We further assume that only the frequencies of n -grams ending with w'_t and w_t^* change. Specifically, let $h^n(x|s_{w_t^* \rightarrow w'_t}^*)$ denote the number of n -gram x after the replacement, where $s_{w_t^* \rightarrow w'_t}^* = \{w_1^*, \dots, w_{t-1}^*, w'_t, w_{t+1}^*, \dots, w_{T+1}^*\}$. Then

$$h^n(x|s_{w_t^* \rightarrow w'_t}^*) = \begin{cases} h^n(x|s^*) + 1 & x = \{w_{t-n+1}^*, \dots, w'_t\} \\ h^n(x|s^*) - 1 & x = \{w_{t-n+1}^*, \dots, w_t^*\} \\ h^n(x|s^*) & \text{else.} \end{cases} \quad (9)$$

An example is illustrated in Figure 2b. A new n -gram fre-

quency vector $\mathbf{h}^n(s_{w_t^* \rightarrow w'_t}^*)$ is obtained:

$$\mathbf{h}^n(s_{w_t^* \rightarrow w'_t}^*) = \{h^n(x^1|s_{w_t^* \rightarrow w'_t}^*), h^n(x^2|s_{w_t^* \rightarrow w'_t}^*), \dots\}. \quad (10)$$

With these assumptions, the length $l(s^*)$ does not change. So the score can be rewritten as

$$r_t(w'_t|s^*) = \psi(\mathbf{H}(s_{w_t^* \rightarrow w'_t}^*), l(s^*)|\Xi) - \psi(\mathbf{H}(s^*), l(s^*)|\Xi), \quad (11)$$

where $\mathbf{H}(s) = \{\mathbf{h}^1(s), \dots, \mathbf{h}^N(s)\}$.

Moreover, we consider the influence of changing the length of s^* when replacing $w_{T+1}^* = \text{“EOS”}$ with other words. Similar to the definition of the score of a word, we define the score of the length as follows:

$$r_{len}(s^*) = \psi(\mathbf{H}(s^*), l(s^*) + 1|\Xi) - \psi(\mathbf{H}(s^*), l(s^*)|\Xi). \quad (12)$$

So we can estimate an NLP metric improvement caused by replacing a word in the generated sentence with any other word in the vocabulary.

The Vocabulary-Critic algorithm is sketched in Algorithm 1.

Algorithm 1 Vocabulary-Critic Algorithm

Input: Sentence $s = \{w_1, \dots, w_t, \dots, w_T, w_{T+1}\}$; Evaluation Metric ψ ; Ground-truth set Ξ ; Vocabulary Ω ;
Output: $r_t(w|s)$, $t \in [1, T+1]$, $w \in \Omega$; $r_{len}(s)$;
1: Compute $\mathbf{H}(s)$ and $l(s)$ (defined in Sec III-B)
2: Compute $y_{base} = \psi(\mathbf{H}(s), l(s)|\Xi)$
3: **for** each $t \in [1, T+1]$ **do**
4: **for** each $w \in \Omega$ **do**
5: Compute $\mathbf{H}(s_{w_t \rightarrow w})$ with (10)
6: Compute $y_{w_t \rightarrow w} = \psi(\mathbf{H}(s_{w_t \rightarrow w}), l(s)|\Xi)$
7: Compute $r_t(w|s) = y_{w_t \rightarrow w} - y_{base}$
8: **end for**
9: **end for**
10: Compute $y_{len} = \psi(\mathbf{H}(s), l(s) + 1|\Xi)$
11: Compute $r_{len}(s) = y_{len} - y_{base}$

C. Fast Calculation of Metrics

At every step t , one needs to calculate the evaluation metrics for the sentence s^* with w_t^* replaced with every w'_t in Ω . This entails $O(|\Omega|L)$ times of metric calculation for replacing every word in s^* . It takes the CPU on our server about 239 ms for calculating the CIDEr-D score. Training one epoch of the MSCOCO Karpathy training images will take about 7 hours, which is still computationally expensive. The reason is that computing $\psi(\mathbf{H}(s), l(s)|\Xi)$ for a sentence s involves $O(L)$ multiplications and additions. Fortunately, we found that there exist efficient algorithms for calculating CIDEr-D and BLEU scores given the scores of the original sentence s^* . The algorithms involve $O(1)$ multiplications and additions. Note that CIDEr-D is one of the most important metrics in evaluating image captioning models [5], [7], [34], [35], [8], [14] and BLEU is widely used across a lot of NLP areas [36], [37], [38], [34].

In what follows, we introduce our fast computing method for CIDEr-D only, since in the image captioning field, more

and more works (e.g. [19], [14], [18]) emphasized on improving CIDEr-D value. And the method for BLEU is in the same spirit and can be found in Appendix A.

Following our notations in Section III-A1 and III-B, $s^* = \{w_1^*, \dots, w_{T+1}^*\}$ denotes the generated sentence of a image, and $\Xi = \{\xi_1, \dots, \xi_m\}$ denotes the ground-truth captions set, where ξ_j is the j -th ground-truth caption in Ξ . In additional, $\text{sum}(x)$ denotes the element-sum of a vector x , $\min(x, y)$ denotes the element minimum of two vectors x and y , $x \cdot y$ denotes inner product of x and y , $\|x\|$ denotes the 2-norm of x , and ∂ denotes the symbol of partial derivative.

CIDEr-D metric consists of two factors, length factor A and frequency factor B . The length factor is

$$A_j(l(s^*)) = e^{\frac{-(l(s^*)-l(\xi_j))^2}{2\sigma^2}}, \quad (13)$$

with $\sigma = 6$, and the frequency factor is shown as follows:

$$B_j^n(s^*) = \frac{\min(\mathbf{g}^n(s^*), \mathbf{g}^n(\xi_j)) \cdot \mathbf{g}^n(\xi_j)}{\|\mathbf{g}^n(s^*)\| \|\mathbf{g}^n(\xi_j)\|}, \quad (14)$$

where $\mathbf{g}^n(s) = [g^n(x^1|s), g^n(x^2|s), \dots]$ (x^i is the i -th n -gram) denotes the TF-IDF weighting for the n -grams of s . Denote $\text{IDF}^n(x)$ as the IDF of the n -gram x which is computed with the training captions, the elements of $\mathbf{g}^n(s)$ is defined as

$$g^n(x|s) = \frac{h^n(x|s)}{\text{sum}(\mathbf{h}^n(s))} \text{IDF}^n(x). \quad (15)$$

The CIDEr-D for n -gram is defined as

$$\text{CIDEr-D}_n(s^*|\Xi) = \frac{10}{m} \sum_j A_j(l(s^*)) B_j^n(s^*), \quad (16)$$

where m is the number of captions in Ξ . The final CIDEr-D score is a weighted sum of these CIDEr-D _{n} :

$$\text{CIDEr-D}(s^*|\Xi) = \sum_{n=1}^N w_n \text{CIDEr-D}_n(s^*|\Xi), \quad (17)$$

and we usually set $w_n = 1/N$ and $N = 4$ in (17).

The reward of replacing w_t^* with w'_t is

$$r_t(w|s^*) = \sum_{n=1}^N \frac{10w_n}{m} \sum_j A_j(l(s^*)) \triangle B_j^n(s^*|w_t^* \rightarrow w'_t), \quad (18)$$

where

$$\triangle B_j^n(s^*|w_t^* \rightarrow w'_t) = B_j^n(s_{w_t^* \rightarrow w'_t}^*) - B_j^n(s^*). \quad (19)$$

We estimate (19) with the gradient of $B_j^n(s^*)$:

$$\triangle B_j^n(s^*|w_t^* \rightarrow w'_t) \approx \frac{\partial B_j^n(s^*)}{\partial h^n(x'_t|s^*)} - \frac{\partial B_j^n(s^*)}{\partial h^n(x_t^*|s^*)}, \quad (20)$$

where $x'_t = \{w_{t-n+1}^*, \dots, w'_t\}$ and $x_t^* = \{w_{t-n+1}^*, \dots, w_t^*\}$.

Following (18) and (20), we can calculate $r_t(w'_t|s^*)$ in $O(1)$ time complexity of multiplications and additions if the gradient of $B_j^n(s^*)$ have been calculated for all n and j .

To investigate the efficiency of the proposed fast metric calculation algorithms, we compared the time costs of this algorithm and the standard algorithm for calculating vocabulary-wide credit assignment in 5000 sentences generated by the

Up-Down model [8], which are presented in Table I. It is seen that the proposed algorithm saved the time cost for calculating CIDEr-D and BLEU-4 by 97% and 72%, respectively.

TABLE I
TIME COST OF THE STANDARD AND SIMPLIFIED ALGORITHMS
FOR CALCULATING VOCABULARY-WIDE CREDIT ASSIGNMENT IN
ONE SENTENCE. MEAN \pm STANDARD DEVIATION.

	Standard	Simplified
CIDEr-D	239.6 \pm 1.8(ms)	7.4 \pm 0.9(ms)
BLEU-4	62.3 \pm 0.9(ms)	17.6 \pm 1.9(ms)

D. Vocabulary-Critical Sequence Training (VCST)

With the vocabulary-critic, at every step t of sentence s^* we can assign a score $r_t(w'_t|s^*)$ to every word w'_t in the vocabulary Ω . These scores are used to guide the sentence generation. Similar to other policy-based RL methods, our training target is to minimize the negative expected reward at every time step t :

$$\begin{aligned} L_{vcst}(\theta|s^*) &= - \sum_{t=1}^{T+1} \mathbb{E}_{w'_t \sim p_t^\theta} [r_t(w'_t|s^*)] \\ &\quad - \mathbb{E}_{w'_{T+1} \sim p_{T+1}^\theta} [r_{len}(s^*)] \\ &= - \sum_{t=1}^{T+1} \sum_{w'_t \in \Omega} r_t(w'_t|s^*) p_t^\theta(w'_t) \\ &\quad + r_{len}(s^*) p_{T+1}^\theta(\text{"EOS"}) + \text{const.} \end{aligned} \quad (21)$$

The second equality holds because

$$\begin{aligned} \mathbb{E}_{w'_{T+1} \sim p_{T+1}^\theta} r_{len}(s^*) &= \sum_{w'_{T+1} \in \Omega \setminus \text{"EOS"}} r_{len}(s^*) p_{T+1}^\theta(w'_{T+1}) \\ &= r_{len}(s^*) (1 - p_{T+1}^\theta(\text{"EOS"})). \end{aligned} \quad (22)$$

Note that $r_{len}(s^*)$ is independent with θ , thus is a constant. So the gradient of VCST loss is

$$\begin{aligned} \nabla_\theta L_{vcst}(\theta|s^*) &= - \sum_{t=1}^{T+1} \sum_{w'_t \in \Omega} r_t(w'_t|s^*) \nabla_\theta p_t^\theta(w'_t) \\ &\quad + r_{len}(s^*) \nabla_\theta p_{T+1}^\theta(\text{"EOS"}). \end{aligned} \quad (23)$$

E. Combining VCST with Other RL Methods

Since the VCST loss is derived under rather strong assumptions, VCST cannot get a good performance without help of other RL methods. We combine their losses with the loss of VCST for training an image captioning model. For SCST, the combined loss is:

$$L_{all} = L_{scst} + \alpha L_{vcst}, \quad (24)$$

where α is a hyper-parameter to be tuned in different experiments. For ACST, the combined loss is:

$$L_{all} = L_{acst} + \alpha L_{vcst}. \quad (25)$$

IV. EXPERIMENT SETTINGS

A. Dataset

The models were trained and tested on the MSCOCO 2014 caption dataset [39], [40] which contains 123287 images. Each image has 5 or 6 captions as its ground-truth. We followed the Karpathy's split [41] where there are 113287 images in the training set, 5000 images in the validation set and 5000 in the test set. We report the results on BLEU, ROUGE-L [42], METEOR [43], CIDEr-D and SPICE metrics.

In the experiments on the Att2in [7], Top-Down [8], Up-Down [8] and X-LAN [21] models, we pruned the words which appear less than 5 times and obtained a vocabulary of size 9487 words. In the experiments on the SGAE model [16], we followed the pre-processing in the SGAE model [16] which has a vocabulary of 10369 words.

B. Image Features

We used two sets of image features in the experiments.

(a) Features extracted from layer4 of ResNet-101 [44] trained on the ImageNet [45] classification task. We resized the image to 448×448 , and applied average pooling to the feature map of layer4. The shape of our final features was $14 \times 14 \times 2048$. We used the 14×14 vectors as the input of attention mechanism and the average of the 14×14 vectors as the feature of the whole image.

(b) Features extracted from Faster R-CNN [25] with ResNet-101, the same as the representation used in [8]. The Faster R-CNN generated regions of interest (RoI), and outputs the feature map of RoI pooling5. The number of RoIs was in the range of [10, 100]. The dimension of each feature of RoI was 2048. We used these features of RoIs as the input of attention mechanism and the average of all RoI features as the feature of the whole image.

C. Captioning Models

We experimented with five recent models: the Att2in model [7], the Top-Down model [8], the Up-Down model [8], the SGAE model [16] and the X-LAN model [21]. In the experiments, the Att2in model and the Top-Down model took features (a) as input. The Up-Down model, the SGAE model and the X-LAN model took features (b) as input.

D. Implementation Details

1) *Combination of VCST and SCST*: Since SCST is a general training method in image captioning, our experiments mainly combined VCST with SCST. We applied the combination of VCST and SCST to the Att2in, Top-Down, Up-Down and SGAE models. Both dimensions of LSTM hidden layer and word embedding were set to 1024 for the Att2in, Top-Down and Up-Down models, and 1000 for the SGAE model.

A three-stage training process was adopted in our experiments. The Att2in, Top-Down and Up-Down models were trained from scratch. In the first stage, cross-entropy (XE) loss was adopted to train the model with random initialization. The learning rate was set to 5×10^{-4} initially. We performed the learning rate decay every 3 epochs by a factor of 0.8 and

TABLE II
TEST SCORES (%) OF THE UP-DOWN MODEL TRAINED WITH DIFFERENT LOSSES.

Method	BLEU-4	CIDEr-D
Up-Down+XE	33.7	109.0
Up-Down+XE+VCST	33.4	107.6
Up-Down+SCST	37.3	121.9
Up-Down+SCST+VCST	38.0	125.0
Up-Down+VCST	30.2	98.9

increased the probability of schedule sampling [10] every 3 epochs, from 0.05 to 0.25. In the second stage, SCST was adopted to optimize the model. In SCST, we used the learning rate of 5×10^{-5} and decayed the learning rate every 4 epochs by a factor of 0.8. In the last stage, We used the loss in (21) to train the model with a learning rate 5×10^{-5} . For the SGAE model, we started from the pre-trained model in [16], then trained the model with the method described in the third stage. The learning rate was set to fixed 5×10^{-6} . For the X-LAN model, we started from the pre-trained model in [21], and only trained the model in the third stage with a 5×10^{-7} learning rate. The ADAM [46] algorithm was used in all training stages. The three stages for the first three models took about 6h, 30h and 70h, respectively, with batch size 50 on a single GTX Titan X, and the third stage for the SGAE model and the X-LAN model took about 70h and 30h, respectively.

2) *Combination of VCST and ACST*: We tested the combined algorithm on the Att2in model and the Up-Down model. The settings of both models were the same as those in Section IV-D1. Different from SCST, ACST needs a pre-trained value network. The value network is an LSTM and the dimension of that LSTM hidden layer is 1024. We pre-trained the value network for 3 epoches using Adam optimizer with a initial learning rate 5×10^{-5} and the learning rate decayed by a factor 0.8 for each epoch. Following [18], we set β in (6) to 0.5. A three-stage training process similar to Section IV-D1 was adopted in training. Different from Section IV-D1, the loss of SCST in the second stage was replaced with loss of ACST, and the combined loss in the third stage was replaced with (25). For all the three stages of training both of the Att2in model and the Up-Down model, we followed the settings of learning rate and α in Section IV-D1.

V. RESULTS

A. Time Cost of the Fast Metric Calculation Algorithms

As we have shown that the fast metric calculation algorithms can reduce the time cost of metric calculation in VCST significantly. However, the time cost of these algorithms inevitably increases with larger vocabulary. To analyse the changing of time cost with different vocabulary, we tested the average time costs of the fast calculation algorithms for vocabulary-wide credit assignment in a minibatch of the Up-Down model with increasing size of vocabulary. The batch size was 50. We fixed the ground truths and generated captions in this experiment for all vocabulary sizes. As shown in Figure 3, the average time costs increased linearly with the vocabulary

TABLE III
THE NUMBER OF BAD-ENDINGS IN CAPTIONS GENERATED BY DIFFERENT MODELS ON THE MSCOCO KARPATY TEST SET.

Method	CIDEr-D	Bad-Ending
Up-Down+XE	109.0	0
Up-Down+SCST	121.9	338
Up-Down+SCST+VCST(standard)	126.2	1529
Up-Down+SCST+VCST(modified)	125.0	9
SGAE+XE	116.7	0
SGAE+SCST	127.8	264
SGAE+SCST+VCST(standard)	130.1	1683
SGAE+SCST+VCST(modified)	129.1	114

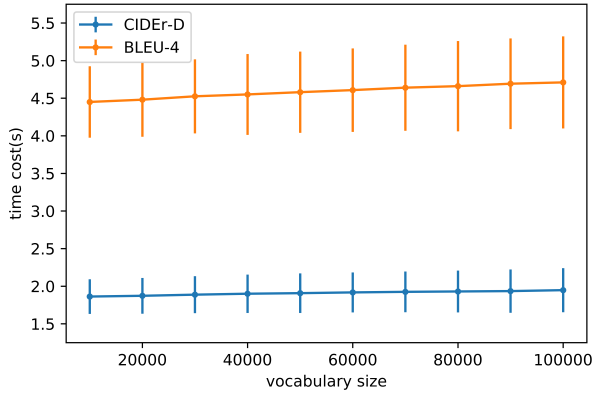


Fig. 3. The average time costs of the fast calculation algorithms for CIDEr-D and BLEU-4 in a minibatch against the vocabulary size. The errorbars denote the standard deviation.

size, but the increases were only about 0.09s and 0.29s for CIDEr-D and BLEU-4, respectively, when the vocabulary size increased from 10k to 100k. In training, the total time costs per minibatch were 4.42s and 8.20s in the two cases. These increases were 0.09/4.42=2.0% and 0.29/8.20=3.5% of the total time costs, which were negligible. Therefore, the time cost of our proposed algorithm are not very much sensitive to the vocabulary size, and has potential to be applied to many tasks.

B. Effect of Hyper-Parameter α

As discussed in Section III-E, we should combine VCST with other RL methods to achieve good performance. To investigate the effect of α , we trained the Att2in, Top-Down, Up-Down, SGAE and X-LAN models with the combination of VCST and SCST, and varied α from 0.1 to 0.9. As shown in Figure 4a, with α increasing, the performance of every model first increased and then decreased. For each model, there was a range of α that obtained significant increase. The Att2in, Top-Down and Up-Down models obtained more than 1 point of CIDEr-D with α in (0.1,0.5), (0.2,0.5), (0.1,0.8), respectively. The SGAE model obtained more than 0.7 points with α in (0.6,0.9). The X-LAN model obtained more than 0.3 points with α in (0.6,0.8). Denote the best α for every model by α^* , one interesting observation was that better models had higher

α^* , indicating that VCST played a more important role on better baseline models.

Another interesting observation in experiments is that we found that we could obtain a higher performance by increasing α gradually than fixing it in training. Figure 4b compares the curves of CIDEr-D metric on the validation set with fixed α and gradually increasing α . It is seen that the models with increasing α performed better.

We think the two observations are highly correlative. Since better models needed higher α , we needed to increase α in training with the performance of models improving.

Then in subsequent experiments, we adopted the strategy that the α was increased by 0.025 when there was no increase of the CIDEr-D performance on the validation set for 3 epoches. The initial α was 0.15, 0.15, 0.2, 0.7, 0.7 for the Att2in, Top-Down, Up-Down, SGAE and X-LAN models, respectively. And the final α was 0.5, 0.5, 0.5, 0.9, 0.8 for these models, respectively.

If we set α to a very large value, the combination of VCST and SCST performs like single VCST. To investigate that extreme case, we started from the model trained with cross-entropy (XE) loss and trained the model with the VCST loss (21) alone. Table II shows the performance of that extreme case on the Up-Down model. It is seen that the VCST loss alone did not perform well. There are two possible reasons. The first is that VCST is derived under strong assumptions. The second is that VCST cares too much about local connections of words, which ignores the sentence structure. In VCST we only care about the immediate impact of replacing a word in the generated sentence with another word and ignore the subsequent effects. We also tested the combination of XE loss and VCST loss. In that experiment, we set the weights of the VCST loss and the XE loss to be 0.5 and 1.0, respectively. However, it did not perform well. This might due to the discrepancy between the optimization goals of the XE loss and the VCST loss.

C. Bad-Ending Problem on VCST

A caption with bad-ending means that the caption ends with a word which shouldn't appear at the end such as "a" and "the". Bad-ending is an inherent drawback for maximizing CIDEr-D.

Table III shows the number of captions ended with four words "a", "the", "of" and "with", generated by the Up-Down model and the SGAE model with CIDEr-D as the target on the MSCOCO Karpathy test set. It is seen that SCST yielded many such captions. By combining SCST and VCST, the models generated even more such captions. To alleviate this problem for the proposed method, we followed Rennie et al.'s work [14] and adopted a modified CIDEr-D for training. In the standard CIDEr-D calculation, "EOS" is treated as an indicator to the end of generation only and it is not involved in the calculation of CIDEr-D. In the modified CIDEr-D, "EOS" is treated as both an indicator and a word, i.e. n -grams like $\{w_{T-n+2}^*, \dots, w_T^*, \text{"EOS"}\}$ also participate in CIDEr-D calculation.

Table III shows that, compared to maximizing the standard CIDEr-D, the proposed method yielded slightly lower CIDEr-

TABLE IV

TEST SCORES (%) OF DIFFERENT MODELS ON THE MSCOCO KARPATY TEST SPLIT. THE MODELS WITHOUT “(BLEU)” WERE OPTIMIZED WITH RESPECT TO CIDER-D WHILE THOSE WITH “(BLEU)” WERE OPTIMIZED WITH RESPECT TO BLEU-4. IN OUR IMPLEMENTATION OF THE ATT2IN, TOP-DOWN AND UP-DOWN MODELS, BEAM SEARCH WAS NOT USED DURING TESTING. IN THE SGAE AND THE X-LAN MODEL, BEAM SIZES WERE SET TO 5 AND 3, RESPECTIVELY. THE BEST SCORES OVER THE METHODS IN A BLOCK ARE MARKED IN BOLD.

Method	BLEU-1	BLEU-4	ROUGE-L	METEOR	CIDEr-D	SPICE
Att2in+SCST [14]	-	33.3	55.3	26.3	111.4	-
StackCap+SCST [47]	78.6	36.1	56.9	27.4	120.4	20.9
Up-Down+SCST [8]	79.8	36.3	56.9	27.7	120.1	21.4
CAVP+SCST [48]	-	38.6	58.5	28.3	126.3	21.6
GCN-LSTM+SCST [15]	80.9	38.3	58.5	28.6	128.7	22.1
AoANet+SCST [17]	80.2	38.9	58.8	29.2	129.8	22.4
Att2in+SCST (our implem.)	77.2	33.3	55.5	26.3	113.1	19.6
Att2in+SCST+VCST	77.6	33.6	55.9	26.8	115.8	19.7
Top-Down+SCST (our implem.)	77.0	33.5	55.4	26.2	113.7	19.6
Top-Down+SCST+VCST	76.9	33.7	55.4	26.5	117.1	19.7
Up-Down+SCST (our implem.)	81.4	37.3	58.1	28.2	121.9	21.9
Up-Down+SCST+VCST	81.1	38.0	58.6	28.5	125.0	21.9
SGAE+SCST [16]	80.8	38.4	58.6	28.4	127.8	22.1
SGAE+SCST+VCST	80.9	38.9	58.7	28.6	129.1	22.0
XLAN+SCST [21]	80.8	39.5	59.2	29.5	132.0	23.4
XLAN+SCST+VCST	81.0	39.6	59.7	29.7	132.4	23.4
Up-Down+SCST(BLEU)	76.7	37.9	57.4	27.2	112.0	20.5
Up-Down+SCST+VCST(BLEU)	77.4	38.4	57.6	27.6	113.8	20.8

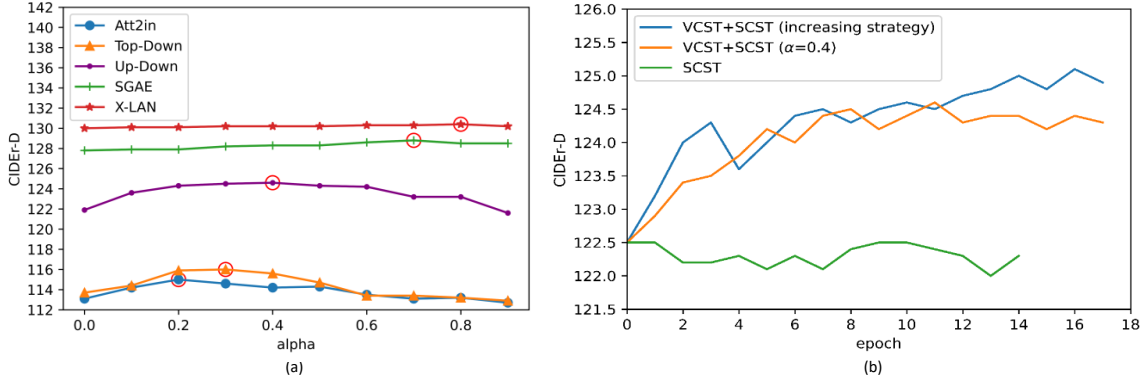


Fig. 4. (a) Performances of the Att2in, Top-Down, Up-Down, SGAE and X-LAN models trained with different fixed α . The performances of these models with their α^* are circled in red. (b) Performance of the Up-Down model trained with different training methods in our third stage as a function of training epoch on the validation set of Karpathy’s MSCOCO splits. The green curve shows the performance of single SCST in the third stage (without VCST).

TABLE V

TEST SCORES (%) OF DIFFERENT MODELS TRAINED WITH SINGLE ACST AND COMBINATION OF VCST AND ACST.

	BLEU-4	CIDEr-D
Att2in+XE	31.6	100.9
Att2in+ACST	33.5	115.0
Att2in+ACST+VCST	33.9	116.5
Up-Down+XE	33.7	109.0
Up-Down+ACST	37.0	121.0
Up-Down+ACST+VCST	37.4	123.5

In the paper we report the results of the Up-Down model and the SGAE model by optimizing the modified CIDEr-D. Note that the modified CIDEr-D was only used in training, and in testing we used the standard CIDEr-D.

D. VCST Improves SCST

Table IV shows our results on MSCOCO Karpathy test split with the combination of VCST and SCST. In the first block of Table IV, we report results of current advanced models. From the second block to the fifth block are the results of optimizing these models by VCST on CIDEr-D. We also applied our VCST on BLEU-4 to the Up-Down model. The results are shown in the last block of Table IV. For all models, we adopted the increasing α strategy in Section V-B.

D scores, but significantly alleviated the bad-ending problem.




I		Output(SCST): two young children playing tennis on a tennis court Output(SCST+VCST): two young children holding tennis rackets on a tennis court GT.1: two young children standing in a tennis court holding tennis gear GT.2: two children wearing shorts hold tennis rackets on a tennis court GT.3: two young children holding tennis rackets and a ball	holding	CIDEr-D: 0.631 BLEU-4: 0.151
			standing	CIDEr-D: 0.471 BLEU-4: 0.151
			wearing	CIDEr-D: 0.182 BLEU-4: 0.021
II		Output(SCST): a little girl standing in a cage with an umbrella Output(SCST+VCST): a little girl is standing in a shopping cart GT.1: a baby girl standing in a shopping cart holding an umbrella GT.2: a little girl inside of a shopping cart GT.3: a small child stands in a shopping cart with an umbrella	shopping	CIDEr-D: 0.573 BLEU-4: 0.128
			little	CIDEr-D: 0.041 BLEU-4: 0.000
			baby	CIDEr-D: 0.033 BLEU-4: 0.000
III		Output(SCST): a man wearing a black hat and a hat Output(SCST+VCST): a man wearing a black hat and a tie GT.1: a man with a hat and a tie on GT.2: a man with earrings wearing a top hat GT.3: a man with a black top hat and suit is staring at the viewer	tie	CIDEr-D: 0.192 BLEU-4: 0.106
			top	CIDEr-D: 0.024 BLEU-4: 0.000
			hat	CIDEr-D: 0.000 BLEU-4: 0.000

Fig. 5. Some examples of word scores in captions evaluated by vocabulary-critic. The red words are the bad words in the captions. The words in the right column are those suggested by vocabulary-critic.

As shown in Table IV, we obtained overall improvements on most of the evaluation metrics when applying the combination of VCST and SCST on the models. When optimizing on CIDEr-D metric, we gained 2.7%, 3.4%, 3.1%, 1.3% and 0.4% of CIDEr-D on the Att2in, Top-Down, Up-Down, SGAE and X-LAN models, respectively, compared with single SCST. However, not all the metrics increased consistently. For example, BLEU-1 decreased 0.1% and 0.3% on the Top-Down and the Up-Down model, respectively, and SPICE decreased 0.1% on the SGAE model. The reason is that different metrics are not necessarily positively correlated. The inconsistency on different metrics can also be seen in other works [17]. Finally, we also gained 0.5% improvement when optimizing on BLEU-4 metric for training the Up-Down model.

E. VCST Improves ACST

Since we could not find the official released code of ACST, we implemented ACST on the Att2in model and the Up-Down model by ourselves. As shown in Table V, incorporated by ACST method, our combined algorithm gained 1.5% improvement on CIDEr-D with Att2in model and 2.5% improvement with the Up-Down model by optimizing CIDEr-D metric. Same to experiments on the combination of VCST and SCST, We also adopted the increasing α strategy in this experiment.

F. Qualitative Analysis

To validate the rationality of VCST, we picked up some bad captions generated by the Up-Down model after SCST (the second stage), and checked how VCST corrected the wrong words in captions. Figure 5 shows three words with top-3 scores assigned by the vocabulary-critic at the highlighted positions. The vocabulary-critic assigned higher scores to the words appeared in the ground-truth captions, which could guide the model to correct the captions. In fact, the most appropriate words “holding”, “shopping” and “tie” received the highest CIDEr-D and BLEU-4 scores, which met our expectation. But some inappropriate words also received positive scores such as “little” and “baby” in image II and “top” in

image II. This problem occurred because the vocabulary-critic is based on n -gram frequency of the ground-truth captions. For example, in the output captions of images II “little” and “baby” are next to “a”, so the vocabulary-critic may give the word w a positive score if w has the form $\{\dots, “a”, w, \dots\}$ in the ground-truth captions.

VI. CONCLUSION

We present a method for training image captioning models called Vocabulary-Critical Sequence Training (VCST). A score is assigned to every word in the vocabulary at every step during training. Combined with the other RL methods, the proposed method can significantly improve the performance of image captioning models. Note that VCST is not limited to image captioning task. It has potential to be applied to other sequence generation tasks such as video captioning and neural machine translation.

APPENDIX A

FAST CALCULATION OF BLEU

BLEU is an important metric in NLP which is widely used in sequence generation tasks like NMT and image captioning. There are two factors in the BLEU metric, length factor A and frequency factor B . Let $s^* = \{w_1^*, \dots, w_{T+1}^*\}$ denote the generated sequence of a image, and $\Xi = \{\xi_1, \dots, \xi_m\}$ denote the ground truth captions set, where ξ_j is the j -th ground truth. $\text{sum}(\mathbf{x})$ denotes the element-sum of a vector \mathbf{x} , \min and $\max(\mathbf{x}, \mathbf{y})$ denotes the element maximum and minimum of two vectors \mathbf{x} and \mathbf{y} , respectively.

Let ξ^* be the ground truth caption in Ξ with the most similar length to s^* . The length factor A is computed as

$$A(l(s^*)) = \begin{cases} 1 & l(s^*) > l(\xi^*) \\ e^{1 - \frac{l(\xi^*)}{l(s^*)}} & l(s^*) \leq l(\xi^*). \end{cases} \quad (26)$$

For each n , P^n is a modified n -gram precision score such that

$$P^n(s^*) = \frac{\text{sum}(\min(\mathbf{h}^n(s^*), \max_j(\mathbf{h}^n(\xi_j))))}{\text{sum}(\mathbf{h}^n(s^*))}. \quad (27)$$

The frequency factor B is the geometric average of $\{P^1, \dots, P^n\}$ such that

$$B(s^*) = \left(\prod_{n=1}^N P^n(s^*) \right)^{\frac{1}{N}}, \quad (28)$$

and the BLEU_N score is the product of these two factors:

$$\text{BLEU}_N(s^*|\Xi) = A(l(s^*))B(s^*). \quad (29)$$

In our experiments about BLEU, we used BLEU-4 in which $N = 4$. Note that BLEU involves $O(L)$ computations where L is the length of the ground-truth captions.

Consider replacing w_t^* with w_t' in the sequence s^* at time step t . We denote $s_{w_t^* \rightarrow w_t'} = \{w_1^*, \dots, w_{t-1}^*, w_t', w_{t+1}^*, \dots, w_{T+1}^*\}$ as the sequence after the replacement. As illustrated in Sec. 3.2, the replacement involves the changes of 2 n -grams such that $x_t' = \{w_{t-n+1}^*, \dots, w_{t-1}^*, w_t'\}$ and $x_t^* = \{w_{t-n+1}^*, \dots, w_t^*\}$. It is clear that $\text{sum}(h^n(s_{w_t^* \rightarrow w_t'}^*)) = T - n + 1$. According to (27), the precision P^n after the replacement is

$$P^n(s_{w_t^* \rightarrow w_t'}^*) = P^n(s^*) + \frac{u(x_t', x_t^* | s_{w_t^* \rightarrow w_t'}^*) - u(x_t', x_t^* | s^*)}{T - n + 1}, \quad (30)$$

where

$$\begin{aligned} u(x^1, x^2 | s) = & \min(h^n(x^1 | s), \max_j(h^n(x^1 | \xi_j))) \\ & + \min(h^n(x^2 | s), \max_j(h^n(x^2 | \xi_j))). \end{aligned} \quad (31)$$

It is clear that $u(x_t', x_t^* | s_{w_t^* \rightarrow w_t'}^*)$ and $u(x_t', x_t^* | s^*)$ only involve $O(1)$ multiplications and additions.

With (30), the BLEU score after the replacement is

$$\text{BLEU}_N(s_{w_t^* \rightarrow w_t'}^*) = A(l(s^*)) \left(\prod_{n=1}^N P^n(s_{w_t^* \rightarrow w_t'}^*) \right)^{\frac{1}{N}}, \quad (32)$$

and the reward is

$$r_t(w_t' | s^*) = \text{BLEU}_N(s_{w_t^* \rightarrow w_t'}^*) - \text{BLEU}_N(s^*). \quad (33)$$

The reward only involves $O(1)$ multiplications and additions according to (30)-(32), and this computing is very efficient with $O(1)$ time complexity.

By this way, it takes only one full BLEU_N computing, one BLEU_N gradient computing, and another $O(|\Omega|L)$ multiplications and additions to evaluate all words in the vocabulary at all time steps.

REFERENCES

- [1] V. Ordonez, G. Kulkarni, and T. L. Berg, "Im2text: Describing images using 1 million captioned photographs," in *Advances in Neural Information Processing Systems*, 2011, pp. 1143–1151.
- [2] P. Kuznetsova, V. Ordonez, T. L. Berg, and Y. Choi, "Treetalk: Composition and compression of trees for image descriptions," *Transactions of the Association for Computational Linguistics*, vol. 2, pp. 351–362, 2014.
- [3] G. Kulkarni, V. Premraj, V. Ordonez, S. Dhar, S. Li, Y. Choi, A. C. Berg, and T. L. Berg, "Babytalk: Understanding and generating simple image descriptions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 12, pp. 2891–2903, 2013.
- [4] M. Hodosh, P. Young, and J. Hockenmaier, "Framing image description as a ranking task: Data, models and evaluation metrics," *Journal of Artificial Intelligence Research*, vol. 47, pp. 853–899, 2013.
- [5] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: Lessons learned from the 2015 mscoco image captioning challenge," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 652–663, 2017.
- [6] K. Cho, B. van Merriënboer, C. Gulcehre, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," in *Conference on Empirical Methods in Natural Language Processing*, 2014.
- [7] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *International Conference on Machine Learning*, 2015, pp. 2048–2057.
- [8] P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang, "Bottom-up and top-down attention for image captioning and visual question answering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6077–6086.
- [9] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [10] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, "Scheduled sampling for sequence prediction with recurrent neural networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 1171–1179.
- [11] R. Vedantam, C. Lawrence Zitnick, and D. Parikh, "Cider: Consensus-based image description evaluation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 4566–4575.
- [12] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation," in *Proceedings of the 40th annual meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 2002, pp. 311–318.
- [13] M. Ranzato, S. Chopra, M. Auli, and W. Zaremba, "Sequence level training with recurrent neural networks," in *4th International Conference on Learning Representations*, 2016. [Online]. Available: <http://arxiv.org/abs/1511.06732>
- [14] S. J. Rennie, E. Marcheret, Y. Mroueh, J. Ross, and V. Goel, "Self-critical sequence training for image captioning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7008–7024.
- [15] T. Yao, Y. Pan, Y. Li, and T. Mei, "Exploring visual relationship for image captioning," in *The European Conference on Computer Vision*, September 2018.
- [16] X. Yang, K. Tang, H. Zhang, and J. Cai, "Auto-encoding scene graphs for image captioning," in *The IEEE Conference on Computer Vision and Pattern Recognition*, June 2019.
- [17] L. Huang, W. Wang, J. Chen, and X.-Y. Wei, "Attention on attention for image captioning," in *The IEEE International Conference on Computer Vision*, October 2019.
- [18] L. Zhang, F. Sung, F. Liu, T. Xiang, S. Gong, Y. Yang, and T. M. Hospedales, "Actor-critic sequence training for image captioning," in *NIPS Workshop on Visually-Grounded Interaction and Language*, 2017.
- [19] J. Gao, S. Wang, S. Wang, S. Ma, and W. Gao, "Self-critical n-step training for image captioning," in *The IEEE Conference on Computer Vision and Pattern Recognition*, June 2019.
- [20] S. Liu, Z. Zhu, N. Ye, S. Guadarrama, and K. Murphy, "Improved image captioning via policy gradient optimization of spider," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 873–881.
- [21] Y. Pan, T. Yao, Y. Li, and T. Mei, "X-linear attention networks for image captioning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 10971–10980.
- [22] T. Yao, Y. Pan, Y. Li, Z. Qiu, and T. Mei, "Boosting image captioning with attributes," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 4894–4902.
- [23] J. Lu, C. Xiong, D. Parikh, and R. Socher, "Knowing when to look: Adaptive attention via a visual sentinel for image captioning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 375–383.
- [24] L. Chen, H. Zhang, J. Xiao, L. Nie, J. Shao, W. Liu, and T.-S. Chua, "Sea-cnn: Spatial and channel-wise attention in convolutional networks for image captioning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5659–5667.
- [25] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 91–99.
- [26] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D. A. Shamma, M. Bernstein, and L. Fei-Fei, "Visual genome: Connecting language and vision using crowdsourced dense image annotations," 2016. [Online]. Available: <https://arxiv.org/abs/1602.07332>

- [27] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.
- [28] J. Yu, J. Li, Z. Yu, and Q. Huang, "Multimodal transformer with multi-view visual representation for image captioning," *IEEE Transactions on Circuits and Systems for Video Technology*, 2019.
- [29] M. Cornia, M. Stefanini, L. Baraldi, and R. Cucchiara, "Meshed-memory transformer for image captioning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 10 578–10 587.
- [30] Q. You, H. Jin, Z. Wang, C. Fang, and J. Luo, "Image captioning with semantic attention," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4651–4659.
- [31] A. M. Lamb, A. G. A. P. Goyal, Y. Zhang, S. Zhang, A. C. Courville, and Y. Bengio, "Professor forcing: A new algorithm for training recurrent networks," in *Advances in Neural Information Processing Systems*, 2016, pp. 4601–4609.
- [32] P. Anderson, B. Fernando, M. Johnson, and S. Gould, "Spice: Semantic propositional image caption evaluation," in *European Conference on Computer Vision*. Springer, 2016, pp. 382–398.
- [33] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine learning*, vol. 8, no. 3-4, pp. 229–256, 1992.
- [34] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 2625–2634.
- [35] J. Mao, W. Xu, Y. Yang, J. Wang, and A. L. Yuille, "Deep captioning with multimodal recurrent neural networks (m-rnn)," in *3rd International Conference on Learning Representations*, 2015. [Online]. Available: <http://arxiv.org/abs/1412.6632>
- [36] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *3rd International Conference on Learning Representations*, 2015. [Online]. Available: <http://arxiv.org/abs/1409.0473>
- [37] K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder–decoder approaches," in *Proceedings of Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 103–111. [Online]. Available: <https://www.aclweb.org/anthology/W14-4012>
- [38] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 3104–3112. [Online]. Available: <http://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf>
- [39] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European Conference on Computer Vision*. Springer, 2014, pp. 740–755.
- [40] X. Chen, H. Fang, T.-Y. Lin, R. Vedantam, S. Gupta, P. Dollár, and C. L. Zitnick, "Microsoft coco captions: Data collection and evaluation server," *arXiv preprint arXiv:1504.00325*, 2015.
- [41] A. Karpathy and L. Fei-Fei, "Deep visual-semantic alignments for generating image descriptions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3128–3137.
- [42] C.-Y. Lin, "Rouge: A package for automatic evaluation of summaries," *Text Summarization Branches Out*, 2004.
- [43] S. Banerjee and A. Lavie, "Meteor: An automatic metric for mt evaluation with improved correlation with human judgments," in *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, 2005, pp. 65–72.
- [44] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.
- [45] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.
- [46] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations*, 2015. [Online]. Available: <http://arxiv.org/abs/1412.6980>
- [47] J. Gu, J. Cai, G. Wang, and T. Chen, "Stack-captioning: Coarse-to-fine learning for image captioning," in *Thirty-Second Association for the Advancement of Artificial Intelligence Conference on Artificial Intelligence*, 2018.
- [48] D. Liu, Z. Zha, H. Zhang, Y. Zhang, and F. Wu, "Context-aware visual policy network for sequence-level image captioning," in *ACM Multimedia Conference*, 2018, pp. 1416–1424. [Online]. Available: <https://doi.org/10.1145/3240508.3240632>