An Empirical Study of Language CNN for Image Captioning

Jiuxiang Gu[1], Gang Wang[2], Jianfei Cai[3], Tsuhan Chen[3]
[1] ROSE Lab, Interdisciplinary Graduate School, Nanyang Technological University, Singapore
[2] Alibaba AI Labs, Hangzhou, China
[3] School of Computer Science and Engineering, Nanyang Technological University, Singapore
{jgu004, asjfcai, tsuhan}@ntu.edu.sg, gangwang6@gmail.com

Abstract

Language models based on recurrent neural networks have dominated recent image caption generation tasks. In this paper, we introduce a language CNN model which is suitable for statistical language modeling tasks and shows competitive performance in image captioning. In contrast to previous models which predict next word based on one previous word and hidden state, our language CNN is fed with all the previous words and can model the long-range dependencies in history words, which are critical for image captioning. The effectiveness of our approach is validated on two datasets: Flickr30K and MS COCO. Our extensive experimental results show that our method outperforms the vanilla recurrent neural network based language models and is competitive with the state-of-the-art methods.

1. Introduction

Image caption generation is a fundamental problem that involves Computer Vision, Natural Language Processing (NLP), and Machine Learning. It can be analogous to "*translating*" an image to proper sentences. While this task seems to be easy for human beings, it is quite challenging for machines because it requires the model to understand the image content and express their relationships in a natural language. Also, the image captioning model should be capable of capturing implicit semantic information of an image and generating humanlike sentences. As a result, generating accurate captions for an image is not an easy task.

The recent surge of research interest in image caption generation task is due to the advances in Neural Machine Translation (NMT) [44] and large datasets [39, 29]. Most image captioning models follow the encoder-decoder pipeline [4, 24, 35, 19, 41]. The encoder-decoder framework is recently introduced for sequence-to-sequence learning based on Recurrent Neural Networks (RNNs) or Long-Short Term Memory (LSTM) networks. Both RNNs and LSTM networks can be sequence learners. However, due to the vanishing gradient problem, RNNs can only remember the previous status for a few time steps. LSTM network is a special type of RNN architecture designed to solve the vanishing gradient problem in RNNs [46, 15, 6]. It introduces a new component called memory cell. Each memory cell is composed of three gates and a neuron with the self-recurrent connection. These gates allow the memory cells to keep and access information over a long period of time and make LSTM network capable of learning long-term dependencies.

Although models like LSTM networks have memory cells which aim to memorize history information for long-term, they are still limited to several time steps because long-term information is gradually diluted at every time step [49]. Besides, vanilla RNNs-based image captioning models recursively accumulate history information without explicitly modeling the hierarchical structure of word sequences, which clearly have a bottom-up structure [28].

To better model the hierarchical structure and long-term dependencies in word sequences, in this paper, we adopt a language CNN which applies temporal convolution to extract features from sequences. Such a method is inspired by works in NLP which have shown CNN is very powerful for text representation [18, 48]. Unlike the vanilla CNN architecture, we drop the pooling operation to keep the relevant information for words representation and investigate the optimum convolutional filters by experiments. However, only using language CNN fails to model the dynamic temporal behavior. Hence, we still need to combine language CNN with recurrent networks (*e.g.*, RNN or LSTM). Our extensive studies show that adding language CNN to a recurrent network helps model sequences consistently and more effectively, and leads to improved results.

To summarize, our primary contribution lies in incorporating a language CNN, which is capable of capturing long-range dependencies in sequences, with RNNs for image captioning. Our model yields comparable performance with the state-of-the-art approaches on Flickr30k [39] and

MS COCO [29].

2. Related Works

The problem of generating natural language descriptions for images has become a hot topic in computer vision community. Prior to using neural networks for generating descriptions, the classical approach is to pose the problem as a retrieval and ranking problem [12, 9, 37]. The main weakness of those retrieval-based approaches is that they cannot generate proper captions for a new combination of objects. Inspired by the success of deep neural networks in machine translation [44, 4, 17], researchers have proposed to use the encoder-decoder framework for image caption generation [21, 35, 19, 46, 6, 3, 26]. Instead of translating sentences between two languages, the goal of image captioning is to "*translate*" a query image into a sentence that describes the image. The earliest approach using neural network for image captioning is proposed by Vinyals *et al.* [46] which is an encoder-decoder system trained to maximize the log-likelihood of the target image descriptions. Similarly, Mao *et al.* [35] and Donahue *et al.* [6] use the multimodal fusion layer to fuse the image features and word representation at each time step. In both cases, *i.e.*, the models in [35] and [6], the captions are generated from the full images, while the image captioning model proposed by Karpathy *et al.* [19] generates descriptions based on regions. This work is later followed by Johnson *et al.* [16] whose method is designed to jointly localize regions and describe each with captions.

Rather than representing an image as a single feature vector from the top-layer of CNNs, some researchers have explored the structure of networks to explicitly or implicitly model the correlation between images and descriptions [51, 34, 30]. Xu *et al.* [51] incorporate the spatial attention on convolutional features of an image into the encoder-decoder framework through the "*hard*" and "*soft*" attention mechanisms. Their work is followed by Yang *et al.* [52] whose method introduces a review network to improve the attention mechanism and Liu *et al.* [30] whose approach is designed to improve the correctness of visual attention. Moreover, a variational autoencoder for image captioning is developed by Pu *et al.* [40]. They use a CNN as the image encoder and use a deep generative deconvolutional network as the decoder together with a Gated Recurrent Unit (GRU) [4] to generate image descriptions.

More recently, high-level attributes have been shown to obtain clear improvements on the image captioning task when injected into existing encoder-decoder based models [50, 53, 8]. Specifically, Jia *et al.* [15] use the semantic information as the extra input to guide the model in generating captions. In addition, Fang *et al.* [7] learn a visual attributes detector based on multi-instance learning (MIL) first and then learn a statistical language model for caption generation. Likewise, Wu *et al.* [50] train several visual attribute classifiers and take the outputs of those classifiers as inputs for the LSTM network to predict words.

In general, current recurrent neural network based approaches have shown their powerful capability on modeling word sequences [46, 19]. However, the history-summarizing hidden states of RNNs are updated at each time, which render the long-term memory rather difficult [25, 36]. Besides, we argue that current recurrent networks like LSTM are not efficient on modeling the hierarchical structure in word sequences. All of these prompt us to explore a new language model to extract better sentence representation. Considering ConvNets can be stacked to extract hierarchical features over long-range contexts and have received a lot of attention in many tasks [10], in this paper, we design a language CNN to model words with long-term dependencies through multilayer ConvNets and to model the hierarchical representation through the bottom-up and convolutional architecture.

3. Model Architecture

3.1. Overall Framework

We study the effect of language CNN by combining it with Recurrent Networks. Figure 1 shows a recursive framework. It consists of one deep CNN for image encoding, one CNN for sentence modeling, and a recurrent network for sequence prediction. In order to distinguish these two CNN networks, we name the first CNN for image feature extraction as $\mathrm{CNN}_{\mathcal{I}}$, and the second CNN for language modeling as $\mathrm{CNN}_{\mathcal{L}}$.

Given an image $\mathbf{I}$, we take the widely-used CNN architecture VGGNet (16-layer) [42] pre-trained on ImageNet [22] to extract the image features $\mathrm{V} \in \mathbb{R}^K$. The $\mathrm{CNN}_{\mathcal{L}}$ is designed to represent words and their hierarchical structure in word sequences. It takes a sequence of t generated words (each word is encoded as a one-hot representation) as inputs and generates a bottom-up representation of these words. The outputs of $\mathrm{CNN}_{\mathcal{I}}$ and $\mathrm{CNN}_{\mathcal{L}}$ will be fed into a multimodal fusion layer, and use the recurrent network $f_{\text{recurrent}}(\cdot)$ to predict the next word. The following equations show the main working flow of our model:

$$\mathrm{V} = \mathrm{CNN}_{\mathcal{I}}(\mathbf{I}) \tag{1}$$

$$\mathrm{y}^{[t]} = \mathrm{CNN}_{\mathcal{L}}(\mathrm{S}^{[0]}, \mathrm{S}^{[1]}, \cdots, \mathrm{S}^{[t-1]}) \tag{2}$$

$$\mathrm{m}^{[t]} = f_{\text{multimodal}}(\mathrm{y}^{[t]}, \mathrm{V}) \tag{3}$$

$$\mathrm{r}^{[t]} = f_{\text{recurrent}}(\mathrm{r}^{[t-1]}, \mathrm{x}^{[t-1]}, \mathrm{m}^{[t]}) \tag{4}$$

$$\mathrm{S}^{[t]} \sim \arg\max_{\mathcal{S}} \mathrm{Softmax}(\mathbf{W}_o \mathrm{r}^{[t]} + \mathbf{b}_o) \tag{5}$$

where $t \in [0, N-1]$ is the time step, $\mathrm{y}^{[t]}$ is the output vector of $\mathrm{CNN}_{\mathcal{L}}$, $\mathrm{r}^{[t]}$ is the activation output of recurrent network, $\mathrm{S}^{[t]}$ is the t-th word drawn from the dictionary $\mathcal{S}$ according

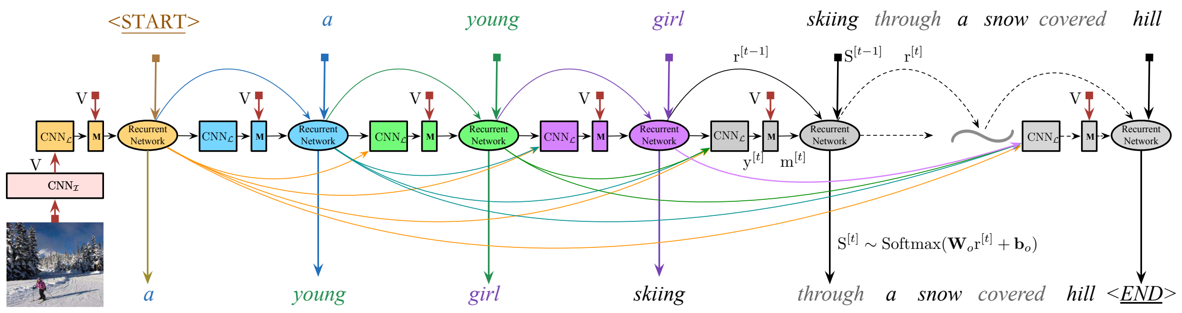


Figure 1. An overview of our framework. The input of our model is a query image. Our model estimates the probability distribution of the next word given previous words and image. It consists of four parts: a CNN$_{\mathcal{I}}$ for image feature extraction, a deep CNN$_{\mathcal{L}}$ for language modeling, a multimodal layer (M) that connects the CNN$_{\mathcal{I}}$ and CNN$_{\mathcal{L}}$, and a Recurrent Network (*e.g.*, RNN, LSTM, *etc.*) for word prediction. The weights are shared among all time frames.

to the maximum Softmax probability controlled by $\mathrm{r}^{[t]}$, $\mathbf{W}_o$ and $\mathbf{b}_o$ are weights and biases used for calculating the distribution over words. Equation 2, 3, 4 and 5 are recursively applied, the design of each function is discussed below.

3.2. CNN$_{\mathcal{L}}$ Layer

Models based on RNNs have dominated recent sequence modeling tasks [23, 31, 32, 44], and most of the recent image captioning models are based on LSTM networks [6, 19, 34]. However, LSTM networks cannot explicitly model the hierarchical representation of words. Even with multi-layer LSTM networks, such hierarchical structure is still hard to be captured due to the more complex model and higher risk of over-fitting.

Inspired by the recent success of CNNs in computer vision [10, 14], we adopt a language CNN with a hierarchical structure to capture the long-range dependencies between the input words, called CNN$_{\mathcal{L}}$. The first layer of CNN$_{\mathcal{L}}$ is a word embedding layer. It embeds the one-hot word encoding from the dictionary into word representation through a lookup table. Suppose we have t input words $\mathbf{S} = \{\mathrm{S}^{[0]}, \mathrm{S}^{[1]}, \cdots, \mathrm{S}^{[t-1]}\}$, and $\mathrm{S}^{[i]}$ is the one-of-V (one-hot) encoding, with V as the size of the vocabulary. We first map each word $\mathrm{S}^{[t]}$ in the sentence into a K-dimensional vector $\mathrm{x}^{[t]} = \mathbf{W}_e\mathrm{S}^{[t]}$, where $\mathbf{W}_e \in \mathbb{R}^{K\times V}$ is a word embedding matrix (to be learned). Next, those embeddings are concatenated to produce a matrix as follows:

$$\mathbf{x} = \left[\mathrm{x}^{[0]}, \mathrm{x}^{[1]}, \cdots, \mathrm{x}^{[t-1]}\right]^T, \mathbf{x} \in \mathbb{R}^{t\times K} \tag{6}$$

The concatenated matrix $\mathbf{x}$ is fed to the convolutional layer. Just like the normal CNN, CNN$_{\mathcal{L}}$ has a fixed architecture with predefined maximum number of input words (denoted as $L_{\mathcal{L}}$). Unlike the toy example in Figure 2, in practice we use a larger and deeper CNN$_{\mathcal{L}}$ with $L_{\mathcal{L}} = 16$.

We use the temporal convolution [21] to model the sentence. Given an input feature map $\mathbf{y}^{(\ell-1)} \in \mathbb{R}^{M_{\ell-1}\times K}$ of Layer-$\ell - 1$, the output feature map $\mathbf{y}^{(\ell)} \in \mathbb{R}^{M_\ell \times K}$ of the temporal convolution layer-ℓ will be:

$$\mathrm{y}_i^{(\ell)}(\mathbf{x}) = \sigma(\mathbf{w}_L^{(l)}\mathbf{y}_i^{(\ell-1)} + \mathrm{b}_L^{(\ell)}) \tag{7}$$

here $\mathrm{y}_i^{(\ell)}(\mathbf{x})$ gives the output of feature map for location i in Layer-ℓ, $\mathbf{w}_L^{(l)}$ denotes the parameters on Layer-ℓ, $\sigma(\cdot)$ is the activation function, *e.g.*, Sigmoid, or ReLU. The input feature map $\mathbf{y}_i^{(l-1)}$ is the segment of Layer-$\ell - 1$ for the convolution at location i, while $\mathbf{y}^{(0)}$ is the concatenation of t word embeddings from the sequence input $\mathrm{S}^{[0:t-1]}$. The definition of $\mathbf{y}^{(0)}$ is as follows:

$$\mathbf{y}^{(0)} \overset{\text{def}}{=} \begin{cases} \left[\mathrm{x}^{[t-L_{\mathcal{L}}]}, \cdots, \mathrm{x}^{[t-1]}\right]^T, & \text{if } t \geq L_{\mathcal{L}} \\ \left[\mathrm{x}^{[0]}, \cdots, \mathrm{x}^{[t-1]}, \tilde{\mathrm{x}}^{[t]}, \cdots, \tilde{\mathrm{x}}^{[L_{\mathcal{L}}-1]}\right]^T & \text{otherwise} \end{cases} \tag{8}$$

Specially, when $t \geq L_{\mathcal{L}}$, the input sentence will be truncated, we only use $L_{\mathcal{L}}$ words before the current time step t. When $t < L_{\mathcal{L}}$, the input sentence will be padded with $\tilde{\mathrm{x}}^{[:]}$. Note that if $t = 0$, $\tilde{\mathrm{x}}^{[:]}$ are the image features V, otherwise $\tilde{\mathrm{x}}^{[:]}$ are the zero vectors that have the same dimension as $\mathrm{x}^{[:]}$.

Previous CNNs, including those adopted for NLP tasks [13, 18], take the classic *convolution-pooling* strategy, which uses *max-pooling* to pick the highest response feature across time. This strategy works well for tasks like text classification [18] and matching [13], but is undesirable for modeling the composition functionality, because it ignores the temporal information in sequence. In our network, we discard the pooling operations. We consider words as the smallest linguistic unit and apply a straightforward stack of convolution layers on top of each other. In practice, we find that deeper CNN$_{\mathcal{L}}$ works better than shallow CNN$_{\mathcal{L}}$, which is consistent with the tradition of CNNs in computer vision [10], where using very deep CNNs is key to having better feature representation.

The output features of the final convolution layer are fed into a fully connected layer that projects the extracted words features into a low-dimensional representation. Next, the projected features will be fed to a highway connection [43] which controls flows of information in the layer and improves the gradient flow. The final output of the highway connection is a K-dimensional vector $\mathrm{y}^{[t]}$.

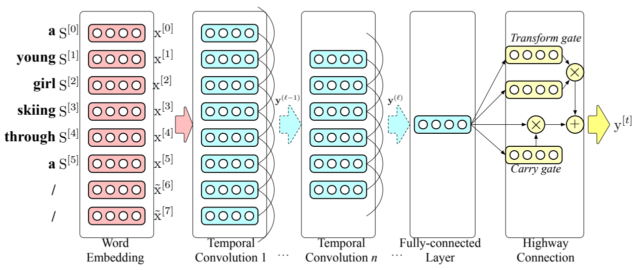


Figure 2. The architecture of language CNN for sentence modeling. Here "/" stands for a zero padding. The $\mathrm{CNN}_{\mathcal{L}}$ builds a hierarchical representation of history words which contains the useful information for next word prediction.

3.3. Multimodal Fusion Layer

Next, we add a multimodal fusion layer after $\mathrm{CNN}_{\mathcal{L}}$, which fuses words representation and image features. This layer has two inputs: the bottom-up words representation $\mathrm{y}^{[t]}$ extracted from $\mathrm{CNN}_{\mathcal{L}}$ and the image representation V extracted from $\mathrm{CNN}_{\mathcal{I}}$. We map these two inputs to the same multimodal feature space and combine them together to obtain the activation of multimodal features:

$$\begin{aligned} \mathrm{m}^{[t]} &= f_{\text{multimodal}}(\mathrm{y}^{[t]}, \mathrm{V}) && (9) \\ &= \sigma\left(f_{\mathrm{y}}(\mathrm{y}^{[t]}; \mathbf{W}_{\mathrm{Y}}, \mathbf{b}_{\mathrm{Y}}) + g_{\mathrm{v}}(\mathrm{V}; \mathbf{W}_{\mathrm{V}}, \mathbf{b}_{\mathrm{V}})\right) && (10) \end{aligned}$$

where "+" denotes element-wise addition, $f_{\mathrm{y}}(\cdot)$ and $g_{\mathrm{v}}(\cdot)$ are linear mapping functions, $\mathrm{m}^{[t]}$ is the multimodal layer output feature vector. $\sigma(\cdot)$ is the activation function, here we use the scaled $\tanh$ function [27] which leads to a faster training process than the basic $\tanh$ function.

3.4. Recurrent Networks

Our $\mathrm{CNN}_{\mathcal{L}}$ may miss the important temporal information because it extracts the holistic features from the whole sequence of words. To overcome this limitation, we combine it with recurrent networks. In our model, the transition equations of the recurrent network can be formulated as:

$$\mathrm{r}^{[t]} = f_{\text{recurrent}}(\mathrm{r}^{[t-1]}, \mathrm{x}^{[t-1]}, \mathrm{m}^{[t]}) \tag{11}$$

$$\mathrm{S}^{[t]} \sim \arg\max_{\mathcal{S}} \mathrm{Softmax}(\mathbf{W}_o \mathrm{r}^{[t]} + \mathbf{b}_o) \tag{12}$$

where $\mathrm{r}^{[t]}$ denotes the recurrent state, $\mathrm{x}^{[t-1]} = \mathbf{W}_e \mathrm{S}^{[t-1]}$ is the previous word embedding, $\mathrm{m}^{[t]}$ is the multimodal fusion output, and $f_{\text{recurrent}}(\cdot)$ is the transition function of recurrent network. $\mathrm{Softmax}(\mathrm{r}^{[t]})$ is the probability of word $\mathrm{S}^{[t]}$ given by the Softmax layer, and $\mathrm{S}^{[t]}$ is the t-th decoded word. In our study, we combine our language CNN with four types of recurrent networks: Simple RNN, LSTM network, GRU [4], and Recurrent Highway Network (RHN) [54].

Traditionally, the simple RNN updates the recurrent state $\mathrm{r}^{[t]}$ of Equation 11 as follows:

$$\mathrm{r}^{[t]} = \tanh(\mathbf{W}_r \mathrm{r}^{[t-1]} + \mathbf{W}_z \mathrm{z}^{[t]} + \mathbf{b}) \tag{13}$$

where $\mathrm{z}^{[t]}$ is the input. However, this type of simple RNN is hard to deal with long-term dependencies [2]. As the vanishing gradient will make gradients in directions that short-term dependencies are large, while the gradients in directions that correspond to long-term dependencies are small.

LSTM network extends the simple RNN with the gating mechanism (*input* gate, *forget* gate, and *output* gate) to control information flow and a *memory cell* to store the history information, thus it can better model the long-term dependencies than simple RNN.

GRU is an architecture similar to the LSTM, but it has a simplified structure. GRU does not has a separate *memory cell* and exposes its hidden state $\mathrm{r}^{[t]}$ without any control. Thus, it is computationally more efficient and outperforms the LSTM network on many tasks due to its simple structure.

Besides, we also consider a fourth type of recurrent network: RHN, which introduces the highway connection to simple RNN. RHN has directly gated connections between previous state $\mathrm{r}^{[t-1]}$ and current input $\mathrm{z}^{[t]}$ to modulate the flow of information. The transition equations of RHN can be formulated as follows:

$$\begin{aligned} \begin{pmatrix} \mathrm{t}^{[t]} \\ \mathrm{c}^{[t]} \\ \mathrm{h}^{[t]} \end{pmatrix} &= \begin{pmatrix} \sigma \\ \sigma \\ \tanh \end{pmatrix} \left(\mathbf{M} \begin{pmatrix} \mathrm{r}^{[t-1]} \\ \mathrm{z}^{[t]} \end{pmatrix} \right) && (14) \\ \mathrm{r}^{[t]} &= \mathrm{h}^{[t]} \odot \mathrm{t}^{[t]} + \mathrm{c}^{[t]} \odot \mathrm{r}^{[t-1]} && (15) \end{aligned}$$

where $\mathrm{c}^{[t]}$ is the *carry* gate, $\mathrm{t}^{[t]}$ is the *transform* gate, $\mathrm{h}^{[t]}$ denotes the modulated input, $\mathbf{M} : \mathbb{R}^{2K+d} \to \mathbb{R}^{3d}$ is an affine transformation. $\mathrm{z}^{[t]} \in \mathbb{R}^{2K}$ denotes the concatenation of two vectors: $\mathrm{m}^{[t]}$ and $\mathrm{x}^{[t-1]}$. According to Equation 3 and 2, $\mathrm{z}^{[t]}$ can be expressed as follows:

$$\mathrm{z}^{[t]} = [f_{\text{multimodal}}(\mathrm{CNN}_{\mathcal{L}}(\mathrm{x}^{[0,\cdots,t-1]}), \mathrm{V}); \mathrm{x}^{[t-1]}] \tag{16}$$

Like GRU, RHN does not have *output* gate to control the exposure of the recurrent state $\mathrm{r}^{[t]}$, but exposes the whole state each time. The RHN, however, does not have *reset* gate to drop information that is irrelevant in the future. As our $\mathrm{CNN}_{\mathcal{L}}$ can extract the relevant information from the sequence of history words at each time step, to some extent, the $\mathrm{CNN}_{\mathcal{L}}$ allows the model to add information that is useful in making a prediction.

3.5. Training

During training, given the ground truth words $\mathbf{S}$ and corresponding image $\mathbf{I}$, the loss function for a single training instance $(\mathbf{S}, \mathbf{I})$ is defined as a sum of the negative log likelihood of the words. The loss can be written as:

$$\mathcal{L}(\mathbf{S}, \mathbf{I}) = -\sum_{t=0}^{N-1} \log P(\mathrm{S}^{[t]} | \mathrm{S}^{[0]}, \cdots, \mathrm{S}^{[t-1]}, \mathbf{I}) \qquad (17)$$

where N is the sequence length, and $\mathrm{S}^{[t]}$ denotes a word in the sentence $\mathbf{S}$.

The training objective is to minimize the cost function, which is equivalent to maximizing the probability of the ground truth context words given the image by using: $\arg\max_\theta \sum_{t=0}^{N-1} \log P(\mathrm{S}^{[t]} | \mathrm{S}^{[0:t-1]}, \mathbf{I})$, where θ are the parameters of our model, and $P(\mathrm{S}^{[t]} | \mathrm{S}^{[0:t-1]}, \mathbf{I})$ corresponds to the activation of Softmax layer.

3.6. Implementation Details

In the following experiments, we use the 16-layer VGGNet [42] model to compute CNN features and map the last fully-connected layer's output features to an embedding space via a linear transformation.

As for preprocessing of captions, we transform all letters in the captions to lowercase and remove all the non-alphabetic characters. Words occur less than five times are replaced with an unknown token <UNK>. We truncate all the captions longer than 16 tokens and set the maximum number of input words for $\mathrm{CNN}_\mathcal{L}$ to be 16.

3.6.1 Training Details

In the training process, each image $\mathbf{I}$ has five corresponding annotations. We first extract the image features V with $\mathrm{CNN}_\mathcal{I}$. The image features V are used in each time step. We map each word representation $\mathrm{S}^{[t]}$ with: $\mathrm{x}^{[t]} = W_e \mathrm{S}^{[t]}, t \in [0, N-1]$. After that, our network is trained to predict the words after it has seen the image and preceding words. Please note that we denote by $\mathrm{S}^{[0]}$ a special <START> token and by $\mathrm{S}^{[N-1]}$ a special <END> token which designate the start and end of the sentence.

For Flickr30K [39] and MS COCO [29] we set the dimensionality of the image features and word embeddings as 512. All the models are trained with Adam [20], which is a stochastic gradient descent method that computes adaptive learning rate for each parameter. The learning rate is initialized with 2e-4 for Flickr30K and 4e-4 for MS COCO, and the restart technique mentioned in [33] is adopted to improve the convergence of training. Dropout and early stopping are used to avoid overfitting. All weights are randomly initialized except for the CNN weights. More specifically, we fine-tune the VGGNet when the validation loss stops decreasing. The termination of training is determined by evaluating the CIDEr [45] score on the validation split after each training epoch.

3.6.2 Testing

During testing, the previous output $\mathrm{S}^{[t-1]}$ is used as input in lieu of $\mathrm{S}^{[t]}$. The sentence generation process is straightforward. Our model starts from the <START> token and calculates the probability distribution of the next word : $P(\mathrm{S}^{[t]} | \mathrm{S}^{[0:t-1]}, \mathbf{I})$. Here we use Beam Search technology proposed in [15], which is a fast and efficient decoding method for recurrent network models. We set a fixed beam search size (k=2) for all models (with RNNs) in our tests.

4. Experiments

4.1. Datasets and Evaluation Metrics

We perform experiments on two popular datasets that are used for image caption generation: MS COCO and Flickr30k. These two datasets contain 123,000 and 31,000 images respectively, and each image has five reference captions. For MS COCO, we reserve 5,000 images for validation and 5,000 images for testing. For Flickr30k, we use 29,000 images for training, 1,000 images for validation, and 1,000 images for testing.

We choose four metrics for evaluating the quality of the generated sentences: **BLEU-n** [38] is a precision-based metric. It measures how many words are shared by the generated captions and ground truth captions. **METEOR** [5] is based on the explicit word to word matches between generated captions and ground-truth captions. **CIDEr** [45] is a metric developed specifically for evaluating image captions. It measures consensus in image caption by performing a Term Frequency-Inverse Document Frequency weighting for each n-gram. **SPICE** [1] is a more recent metric which has been shown to correlate better with the human judgment of semantic quality than previous metrics.

4.2. Models

To gain insight into the effectiveness of $\mathrm{CNN}_\mathcal{L}$, we compare $\mathrm{CNN}_\mathcal{L}$-based models with methods using the recurrent network only. For a fair comparison, the output dimensions of all gates are fixed to 512.

Recurrent Network-based Models. We implement Recurrent Network-based Models based on the framework proposed by Vinyals *et al.* [46], it takes an image as input and predicts words with one-layer Recurrent Network. Here we use the publicly available implementation Neuraltalk2 [1]. We evaluate four baseline models: **Simple RNN**, **RHN**, **LSTM**, and **GRU**.

$\mathrm{CNN}_\mathcal{L}$-based Models. As can be seen in Figure 1. The $\mathrm{CNN}_\mathcal{L}$-based models employ a $\mathrm{CNN}_\mathcal{L}$ to obtain the bottom-up representation from the sequence of words and cooperate with the Recurrent Network to predict the next word. Image features and words representation learned from $\mathrm{CNN}_\mathcal{I}$ and

[1]`https://github.com/karpathy/neuraltalk2`

$\text{CNN}_{\mathcal{L}}$ respectively are fused with the multimodal function. We implement four $\text{CNN}_{\mathcal{L}}$-based models: **$\text{CNN}_{\mathcal{L}}$+Simple RNN**, **$\text{CNN}_{\mathcal{L}}$+RHN**, **$\text{CNN}_{\mathcal{L}}$+LSTM**, and **$\text{CNN}_{\mathcal{L}}$+GRU**.

4.3. Quantitative Results

We first evaluate the importance of language CNN for image captioning, then evaluate the effects of $\text{CNN}_{\mathcal{L}}$ on two datasets (Flickr30K and MS COCO), and also compare with the state-of-the-art methods.

4.3.1 Analysis of $\text{CNN}_{\mathcal{L}}$

It is known that $\text{CNN}_{\mathcal{L}}$-based models have larger capacity than RNNs. To verify that the improved performance is from the developed $\text{CNN}_{\mathcal{L}}$ rather than due to more layers/parameters, we set the hidden and output sizes of RNNs to 512 and 9568 (vocabulary size), and list the parameters of each model as well as their results in Table 1.

Approach	Params	B@4	C	Approach	Params	B@4	C
Simple RNN	5.4M	27.0	87.0	LSTM	7.0M	29.2	92.6
$\text{CNN}_{\mathcal{L}}$	6.3M	18.4	56.8	LSTM_2	9.1M	**29.7**	93.2
$\text{CNN}_{\mathcal{L}}$+RNN	**11.7M**	29.5	**95.2**	LSTM_3	11.2M	29.3	92.9

Table 1. Results on MS COCO, where B@n is short for BLEU-n, C is short for CIDEr. All values are reported as percentage (Bold numbers are the best results). $\text{CNN}_{\mathcal{L}}$ contains five temporal convolutional layers, the kernel size of the first two convolutional layers is 5, and the rest kernel size of convolutional layers is 3.

As seen in Table 1, the parameter size of the 3-layer LSTM (LSTM_3) is close to that of the $\text{CNN}_{\mathcal{L}}$+RNN. Adding the 2nd LSTM layer (LSTM_2) improves the performance of LSTM, but it is still lower than $\text{CNN}_{\mathcal{L}}$+RNN. Meanwhile, LSTM_3 does not show improvements as the model experiences overfitting. This issue is even worse on Flickr30K which has relatively small number of training data. Note that $\text{CNN}_{\mathcal{L}}$ (without RNNs) achieves lower performance than $\text{CNN}_{\mathcal{L}}$+RNN. We find that those predicted captions of $\text{CNN}_{\mathcal{L}}$ (without RNNs) only are short, but contain primary attributes, *e.g.*, $\text{CNN}_{\mathcal{L}}$ model generates: "*a person on a wave*", while $\text{CNN}_{\mathcal{L}}$+RNN provides: "*a young man surfing a wave*". This finding shows that the temporal recurrence of RNNs is still crucial for modeling the short-term contextual information across words in the sentence.

We further compare language CNNs with different input words and with *max-pooling* operations, where those language CNNs are combined with RHN instead of RNN. Table 2 shows that larger context windows achieve better performance. This is likely because $\text{CNN}_{\mathcal{L}}$ with larger window size can better utilize contextual information and learn better word embedding representation. In addition, the performance of $\text{CNN}_{\mathcal{L}^*_{16\text{ words}}}$+RHN is inferior to $\text{CNN}_{\mathcal{L}}$+RHN, which experimentally supports our opinion that *max-pooling* operations lose information about the local order of words.

Approach	B@4	C	Approach	B@4	C
$\text{Avg}_{\text{history}}$+RHN	30.1	95.8	$\text{CNN}_{\mathcal{L}_{2\text{ words}}}$+RHN	29.2	93.8
$\text{CNN}_{\mathcal{L}^*_{16\text{ words}}}$+RHN	28.9	91.9	$\text{CNN}_{\mathcal{L}_{4\text{ words}}}$+RHN	29.5	95.8
$\text{CNN}_{\mathcal{L}}$+RHN	**30.6**	**98.9**	$\text{CNN}_{\mathcal{L}_{8\text{ words}}}$+RHN	30.0	95.9

Table 2. Results of different history information encoding approaches on MS COCO. $\text{CNN}_{\mathcal{L}_{N\text{words}}}$ takes N previous words as inputs, where we set N to 2, 4, and 8. $\text{Avg}_{\text{history}}$ computes an average over history word embeddings. $\text{CNN}_{\mathcal{L}^*_{16\text{ words}}}$ replaces the 2nd and 4th convolutional layers in $\text{CNN}_{\mathcal{L}}$ with the *max-pooling* layer.

4.3.2 Results Using $\text{CNN}_{\mathcal{L}}$ on MS COCO

Table 3 shows the generation performance on MS COCO. By combine $\text{CNN}_{\mathcal{L}}$, our methods clearly outperforms the recurrent network counterpart in all metrics.

Approach	B@1	B@2	B@3	B@4	M	C	S
Simple RNN	70.1	52.1	37.6	27.0	23.2	87.0	16.0
$\text{CNN}_{\mathcal{L}}$+RNN	72.2	55.0	40.7	29.5	24.5	95.2	17.6
RHN	70.5	52.7	37.8	27.0	24.0	90.6	17.2
$\text{CNN}_{\mathcal{L}}$+RHN	72.3	55.3	**41.3**	**30.6**	**25.2**	98.9	**18.3**
LSTM	70.8	53.6	39.5	29.2	24.5	92.6	17.1
$\text{CNN}_{\mathcal{L}}$+LSTM	72.1	54.6	40.9	30.4	25.1	**99.1**	18.0
GRU	71.6	54.1	39.7	28.9	24.3	93.3	17.2
$\text{CNN}_{\mathcal{L}}$+GRU	**72.6**	**55.4**	41.1	30.3	24.6	96.1	17.6

Table 3. Performance comparison on MS COCO, where M is short for METEOR, and S is short for SPICE.

Approach	B@1	B@2	B@3	B@4	M	C	S
Simple RNN	60.5	41.3	28.0	19.1	17.1	32.5	10.5
$\text{CNN}_{\mathcal{L}}$+RNN	71.3	53.8	39.6	28.7	**22.6**	**65.4**	**15.6**
RHN	62.1	43.1	29.4	20.0	17.7	38.4	11.4
$\text{CNN}_{\mathcal{L}}$+RHN	**73.8**	**56.3**	**41.9**	**30.7**	21.6	61.8	15.0
LSTM	60.9	41.8	28.3	19.3	17.6	35.0	11.1
$\text{CNN}_{\mathcal{L}}$+LSTM	64.5	45.8	32.2	22.4	19.0	45.0	12.5
GRU	61.4	42.5	29.1	20.0	18.1	39.5	11.4
$\text{CNN}_{\mathcal{L}}$+GRU	71.4	54.0	39.5	28.2	21.1	57.9	14.5

Table 4. Performance comparison on Flickr30k.

Among these models, $\text{CNN}_{\mathcal{L}}$+RHN achieves the best performances in terms of B@(3,4), METEOR, and SPICE metrics, $\text{CNN}_{\mathcal{L}}$+LSTM achieves the best performance in CIDEr metric (99.1), and $\text{CNN}_{\mathcal{L}}$+GRU achieves the best performance in B@(1,2) metrics. Although the absolute gains across different B@n metrics are similar, the percentage of the relative performance improvement is increasing from B@1 to B@4. It does show the advantage of our method in terms of better capturing long-term dependency. Note that the $\text{CNN}_{\mathcal{L}}$+RNN model achieves better performance than simple RNN model and outperforms LSTM model. As mentioned in Section 3.4, LSTM networks model the word dependencies with multi-gates and the internal memory cell. However, our $\text{CNN}_{\mathcal{L}}$+RNN without memory cell works better than LSTM model. We think the reason is that our language CNN takes all history words as input and explicitly model the long-term dependencies in history words, this could be regarded as an external "*memory cell*". Thus, the $\text{CNN}_{\mathcal{L}}$'s ability to model long-term de-

	Flickr30k					*MS COCO*					
Approach	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	CIDEr
BRNN [19]	57.3	36.9	24.0	15.7	—	62.5	45.0	32.1	23.0	19.5	66.0
Google NIC [46]	—	—	—	—	—	—	—	—	27.7	23.7	85.5
LRCN [6]	58.8	39.1	25.1	16.5	—	66.9	48.9	34.9	24.9	—	—
MSR [7]	—	—	—	—	—	—	—	—	25.7	23.6	—
m-RNN [35]	60.0	41.0	28.0	19.0	—	67.0	49.0	35.0	25.0	—	—
Hard-Attention [51]	66.9	43.9	29.6	19.9	18.5	70.7	49.2	34.4	24.3	23.9	—
Soft-Attention [51]	66.7	43.4	28.8	19.1	18.5	71.8	50.4	35.7	25.0	23.0	—
ATT-FCN [53]	64.7	46.0	32.4	23.0	18.9	70.9	53.7	40.2	30.4	24.3	—
ERD+GoogLeNet [52]	—	—	—	—	—	—	—	—	29.8	24.0	88.6
emb-gLSTM [15]	64.6	44.6	30.5	20.6	17.9	67.0	49.1	35.8	26.4	22.7	81.3
VAE [40]	72.0	53.0	38.0	25.0	—	72.0	52.0	37.0	28.0	24.0	90.0
	State-of-the-art results using model assembling or extra information										
Google NICv2 [47]	—	—	—	—	—	—	—	—	32.1	25.7	99.8
Attributes-CNN+RNN [50]	73.0	55.0	40.0	28.0	—	74.0	56.0	42.0	31.0	26.0	94.0
	Our results										
CNN$_{\mathcal{L}}$+RNN	71.3	53.8	39.6	28.7	22.6	72.2	55.0	40.7	29.5	24.5	95.2
CNN$_{\mathcal{L}}$+RHN	73.8	56.3	41.9	30.7	21.6	72.3	55.3	41.3	30.6	25.2	98.9
CNN$_{\mathcal{L}}$+LSTM	64.5	45.8	32.2	22.4	19.0	72.1	54.6	40.9	30.4	25.1	99.1
CNN$_{\mathcal{L}}$+GRU	71.4	54.0	39.5	28.2	21.1	72.6	55.4	41.1	30.3	24.6	96.1

Table 5. Performance in terms of BLEU-n, METEOR, and CIDEr compared with other state-of-the-art methods on the MS COCO and Flickr30k datasets. For those competing methods, we extract their performance from their latest version of papers.

pendencies can be taken as enhancement of simple RNNs, which can solve the difficulty of learning long-term dependencies.

4.3.3 Results Using CNN$_{\mathcal{L}}$ on Flickr30K

We also evaluate the effectiveness of language CNN on the smaller dataset Flickr30K. The results in Table 4 clearly indicate the advantage of exploiting the language CNN to model the long-term dependencies in words for image captioning. Among all models, CNN$_{\mathcal{L}}$+RHN achieves the best performances in B@(1,2,3,4) metrics, and CNN$_{\mathcal{L}}$+RNN achieves the best performances in METEOR, CIDEr, and SPICE metrics.

As for the low results (without CNN$_{\mathcal{L}}$) on Flickr30k, we think that it is due to lack of enough training data to avoid overfitting. In contrast, our CNN$_{\mathcal{L}}$ can help learn better word embedding and better representation of history words for word prediction, and it is much easier to be trained compared with LSTM due to its simplicity and efficiency. Note that the performance of LSTM and CNN$_{\mathcal{L}}$+LSTM models are lower than RHN/GRU and CNN$_{\mathcal{L}}$+RHN/GRU. This illustrates that the LSTM networks are easily overfitting on this smaller dataset.

4.3.4 Comparison with State-of-the-art Methods

To empirically verify the merit of our models, we compare our methods with other state-of-the-art approaches.

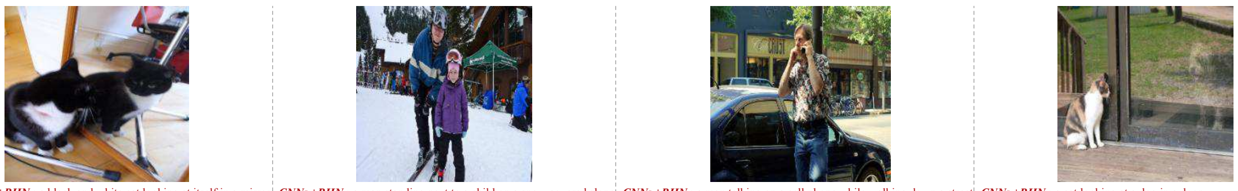
Performance on MS COCO. The right-hand side of Table 5 shows the results of different models on MS COCO dataset. CNN$_{\mathcal{L}}$-based models perform better than most image captioning models. The only two methods with better performance (for some metrics) than ours are Attributes-CNN+RNN [50] and Google NICv2 [47]. However, Wu *et al.* [50] employ an attribute prediction layer, which requires determining an extra attribute vocabulary. While we generate the image descriptions only based on the image features. Google NICv2 [47] is based on Google NIC [46], the results of Google NICv2 are achieved by model ensembling. All our models are based on VGG-16 for a fair comparison with [6, 7, 15, 35, 50, 51]. Indeed, better image CNN (*e.g.* Resnet [11]) leads to higher performance[2]. Despite all this, the CIDEr score of our CNN$_{\mathcal{L}}$+LSTM model can still achieve 99.1, which is comparable to their best performance even with a single VGG-16 model.

Performance on Flickr30K. The results on Flickr30K are reported on the left-hand side of Table 5. Interestingly, CNN$_{\mathcal{L}}$+RHN performs the best on this smaller dataset and even outperforms the Attributes-CNN+RNN [50]. Obviously, there is a significant performance gap between CNN$_{\mathcal{L}}$+RNN/RHN/GRU and RNN/RHN/GRU/LSTM models. This demonstrates the effectiveness of our language CNN on the one hand, and also shows that our CNN$_{\mathcal{L}}$+RNN/RHN/GRU models are more robust and easier to train than LSTM networks when less training data is available.

4.4. Qualitative Results

Figure 3 shows some examples generated by our models. It is easy to see that all of these caption generation models can generate somewhat relevant sentences, while

[2]We uploaded the results based on Resnet-101+CNN$_{\mathcal{L}}$+LSTM (named jxgu_LCNN_NTU) to the official MS COCO evaluation server (`https://competitions.codalab.org/competitions/3221`), and achieved competitive ranking across different metrics.



***CNN$_{\mathcal{L}}$+RHN* :** a black and white cat looking at itself in a mirror
***CNN$_{\mathcal{L}}$+RNN* :** a black and white cat sitting in front of a mirror
***GRU* :** a black and white cat standing next to a mirror
***LSTM* :** a black and white cat sitting in a bathroom sink
***RNN* :** a cat sitting on the floor in a bathroom

- there is a black tuxedo cat looking in the mirror
- two cats sitting on top of a wooden floor
- a cat looking at itself in the mirror next to a tripod
- a cat and a tripod sitting in front of a mirror
- a close up of a cat in a mirror

***CNN$_{\mathcal{L}}$+RHN* :** a man standing next to a child on a snow covered slope
***CNN$_{\mathcal{L}}$+RNN* :** a man and a woman standing on a snow covered slope
***GRU* :** a man and a child standing on a snow covered slope
***LSTM* :** a man and a child are standing in the snow
***RNN* :** a man and a woman are skiing on the snow

- a woman and child in ski gear next to a lodge
- a man and a child are smiling while standing on skiis
- a young man poses with a little kid in the snow
- an adult and a small child dressed for skiing
- a man and a little girl in skis stand in front of a mountain lodge

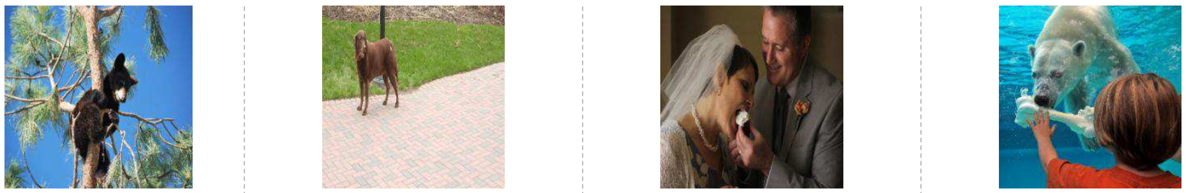
***CNN$_{\mathcal{L}}$+RHN* :** a man talking on a cell phone while walking down a street
***CNN$_{\mathcal{L}}$+RNN* :** a man is talking on a cell phone
***GRU* :** a man is talking on a cell phone in the street
***LSTM* :** a man is talking on his cell phone
***RNN* :** a man standing next to a woman talking on a cell phone

- a man talking on the phone in front of a blue car
- a man on a telephone holds his hand up to his other ear as he walks
- a man standing next to a car with a cellphone
- a man is talking on a cell phone next to a city street
- a man standing on the side of the street with a cell phone up to his

***CNN$_{\mathcal{L}}$+RHN* :** a cat looking at a dog in a door
***CNN$_{\mathcal{L}}$+RNN* :** a cat is looking at a dog in front of a window
***GRU* :** a cat standing next to a door looking out a window
***LSTM* :** a dog and a cat are standing in front of a window
***RNN* :** a cat sitting on the side of the road

- a dog looking at a cat through a glass window
- a cat is outside looking through in at a dog
- the dog wants to go outside with the cat
- a cat sitting outside of a door next to a dog
- a cat sitting at a sliding glass door

Figure 3. Qualitative results for images on MS COCO. Ground-truth annotations (under each dashed line) and the generated descriptions are shown for each image.



***CNN$_{\mathcal{L}}$+RHN* :** a large **bird** perched on top of a tree

- a bear that is hanging in a tree
- a young bear holding onto a pine tree
- a bear cub in the branches of a pine tree
- a black bear cub climbing a pine tree
- the bear cub UNK high up into the tree

***CNN$_{\mathcal{L}}$+RNN* :** a **black and white** dog standing on a sidewalk

- a tan dog standing on a sidewalk next to a UNK and grass
- the dog is standing outside all alone in the backyard
- a dog standing on a brick walk way
- a brown dog is standing on the side of a walk way
- a brown dog standing on a brick path

***CNN$_{\mathcal{L}}$+LSTM* :** a man and a woman holding a glass of **wine**

- a couple that is eating some food together
- the groom is feeding the bride a slice of cake
- a man feeding a piece of cake to his bride
- a husband feeds his wife a piece of cake
- a groom feeding wedding cake to his bride

***CNN$_{\mathcal{L}}$+GRU* :** a polar bear in the water with a **ball** in its mouth

- a child is looking a white bear in a water aquarium
- child stands viewing a polar bear as it dives under water to retrieve a bone
- a boy reaching towards an aquarium in which a polar bear chews on a bone
- a boy watches a polar bear chew on a bone
- a young boy touching the glass of a polar bear

Figure 4. Some failure descriptions for images on MS COCO. Ground-truth descriptions are under each dashed line.

the CNN$_{\mathcal{L}}$-based models can predict more high-level words by jointly exploiting history words and image representations. Take the last image as an example, compared with the sentences generated by RNN/LSTM/GRU model, "*a cat is looking at a dog in front of a window*" generated by CNN$_{\mathcal{L}}$+RNN is more precise to describe their relationship in the image.

Besides, our CNN$_{\mathcal{L}}$-based models can generate more descriptive sentences. For instance, with the detected object "*cat*" in the first image, the generated sentence "*a black and white cat looking at itself in a mirror*" by CNN$_{\mathcal{L}}$+RHN depicts the image content more comprehensively. The results demonstrate that our model with language CNN can generate more humanlike sentences by modeling the hierarchical structure and long-term information of words.

Figure 4 shows some failure samples of our CNN$_{\mathcal{L}}$-based models. Although most of the generated captions are complete sentences. However, the biggest problem is that those predicted visual attributes are wrong. For example, "*bear*" in the first image is detected as "*bird*", and "*brown*" in the second image is detected as "*black and white*". This will decrease the precision-based evaluation score (*e.g.*, B@n). We can improve our model by further taking high-level attributes into account.

5. Conclusion

In this work, we present an image captioning model with language CNN to explore both hierarchical and temporal information in sequence for image caption generation. Experiments conducted on MS COCO and Flickr30K image captioning datasets validate our proposal and analysis. Performance improvements are clearly observed when compared with other image captioning methods. Future research directions will go towards integrating extra attributes learning into image captioning, and how to apply a single language CNN for image caption generation is worth trying.

Acknowledgements

This work is supported by the National Research Foundation, Prime Ministers Office, Singapore, under its IDM Futures Funding Initiative, and NTU CoE Grant. This research was carried out at ROSE Lab at Nanyang Technological University, Singapore. ROSE Lab is supported by the National Research Foundation, Prime Ministers Office, Singapore, under its IDM Futures Funding Initiative and administered by the Interactive and Digital Media Programme Office. We gratefully acknowledge the support of NVAITC (NVIDIA AI Tech Centre) for our research at NTU ROSE Lab, Singapore.

References

[1] P. Anderson, B. Fernando, M. Johnson, and S. Gould. Spice: Semantic propositional image caption evaluation. In *ECCV*, 2016.

[2] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 1994.

[3] X. Chen and C. Lawrence Zitnick. Mind's eye: A recurrent visual representation for image caption generation. In *CVPR*, 2015.

[4] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *EMNLP*, 2014.

[5] M. Denkowski and A. Lavie. Meteor universal: Language specific translation evaluation for any target language. In *ACL*, 2014.

[6] 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 *CVPR*, 2015.

[7] H. Fang, S. Gupta, F. Iandola, R. K. Srivastava, L. Deng, P. Dollár, J. Gao, X. He, M. Mitchell, J. C. Platt, et al. From captions to visual concepts and back. In *CVPR*, 2015.

[8] C. Gan, T. Yang, and B. Gong. Learning attributes equals multi-source domain generalization. In *CVPR*, 2016.

[9] Y. Gong, L. Wang, M. Hodosh, J. Hockenmaier, and S. Lazebnik. Improving image-sentence embeddings using large weakly annotated photo collections. In *ECCV*, 2014.

[10] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, and G. Wang. Recent advances in convolutional neural networks. *arXiv preprint arXiv:1512.07108*, 2015.

[11] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In *ECCV*, 2016.

[12] M. Hodosh, P. Young, and J. Hockenmaier. Framing image description as a ranking task: Data, models and evaluation metrics. *JAIR*, 2013.

[13] B. Hu, Z. Lu, H. Li, and Q. Chen. Convolutional neural network architectures for matching natural language sentences. In *NIPS*, 2014.

[14] P. Hu, B. Shuai, J. Liu, and G. Wang. Deep level sets for salient object detection. 2017.

[15] X. Jia, E. Gavves, B. Fernando, and T. Tuytelaars. Guiding long-short term memory for image caption generation. *ICCV*, 2015.

[16] J. Johnson, A. Karpathy, and L. Fei-Fei. Densecap: Fully convolutional localization networks for dense captioning. In *CVPR*, 2016.

[17] N. Kalchbrenner and P. Blunsom. Recurrent continuous translation models. In *EMNLP*, 2013.

[18] N. Kalchbrenner, E. Grefenstette, and P. Blunsom. A convolutional neural network for modelling sentences. *ACL*, 2014.

[19] A. Karpathy and L. Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *CVPR*, 2015.

[20] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *ICLR*, 2015.

[21] R. Kiros, R. Salakhutdinov, and R. S. Zemel. Multimodal neural language models. In *ICML*, 2014.

[22] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[23] J. Kuen, Z. Wang, and G. Wang. Recurrent attentional networks for saliency detection. In *CVPR*, 2016.

[24] G. Kulkarni, V. Premraj, S. Dhar, S. Li, Y. Choi, A. C. Berg, and T. L. Berg. Baby talk: Understanding and generating image descriptions. In *CVPR*, 2011.

[25] Q. V. Le, N. Jaitly, and G. E. Hinton. A simple way to initialize recurrent networks of rectified linear units. *arXiv preprint arXiv:1504.00941*, 2015.

[26] R. Lebret, P. O. Pinheiro, and R. Collobert. Phrase-based image captioning. *ICML*, 2015.

[27] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller. Efficient backprop. In *Neural networks: Tricks of the trade*, pages 9–48. Springer, 2012.

[28] J. Li, M.-T. Luong, and D. Jurafsky. A hierarchical neural autoencoder for paragraphs and documents. *ACL*, 2015.

[29] 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 *ECCV*, 2014.

[30] C. Liu, J. Mao, F. Sha, and A. L. Yuille. Attention correctness in neural image captioning. In *AAAI*, 2017.

[31] J. Liu, A. Shahroudy, D. Xu, and G. Wang. Spatio-temporal lstm with trust gates for 3d human action recognition. In *ECCV*, 2016.

[32] J. Liu, G. Wang, P. Hu, L.-Y. Duan, and A. C. Kot. Global context-aware attention lstm networks for 3d action recognition. CVPR, 2017.

[33] I. Loshchilov and F. Hutter. Sgdr: Stochastic gradient descent with restarts. *ICLR*, 2016.

[34] J. Lu, C. Xiong, D. Parikh, and R. Socher. Knowing when to look: Adaptive attention via a visual sentinel for image captioning. *CVPR*, 2017.

[35] J. Mao, W. Xu, Y. Yang, J. Wang, Z. Huang, and A. Yuille. Deep captioning with multimodal recurrent neural networks (m-rnn). *ICLR*, 2014.

[36] J. B. Oliva, B. Poczos, and J. Schneider. The statistical recurrent unit. *ICML*, 2017.

[37] V. Ordonez, G. Kulkarni, and T. L. Berg. Im2text: Describing images using 1 million captioned photographs. In *NIPS*, 2011.

[38] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In *ACL*, 2002.

[39] B. A. Plummer, L. Wang, C. M. Cervantes, J. C. Caicedo, J. Hockenmaier, and S. Lazebnik. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *ICCV*, 2015.

[40] Y. Pu, Z. Gan, R. Henao, X. Yuan, C. Li, A. Stevens, and L. Carin. Variational autoencoder for deep learning of images, labels and captions. *NIPS*, 2016.

[41] S. J. Rennie, E. Marcheret, Y. Mroueh, J. Ross, and V. Goel. Self-critical sequence training for image captioning. *CVPR*, 2017.

[42] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*, 2014.

[43] R. K. Srivastava, K. Greff, and J. Schmidhuber. Training very deep networks. In *NIPS*, 2015.

[44] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *NIPS*, 2014.

[45] R. Vedantam, C. Lawrence Zitnick, and D. Parikh. Cider: Consensus-based image description evaluation. In *CVPR*, 2015.

[46] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. In *CVPR*, 2015.

[47] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: Lessons learned from the 2015 mscoco image captioning challenge. *PAMI*, 2016.

[48] M. Wang, Z. Lu, H. Li, W. Jiang, and Q. Liu. gen cnn: A convolutional architecture for word sequence prediction. *ACL*, 2015.

[49] J. Weston, S. Chopra, and A. Bordes. Memory networks. *arXiv preprint arXiv:1410.3916*, 2014.

[50] Q. Wu, C. Shen, L. Liu, A. Dick, and A. v. d. Hengel. What value do explicit high level concepts have in vision to language problems? *CVPR*, 2016.

[51] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. S. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. *ICML*, 2015.

[52] Z. Yang, Y. Yuan, Y. Wu, R. Salakhutdinov, and W. W. Cohen. Review networks for caption generation. *NIPS*, 2016.

[53] Q. You, H. Jin, Z. Wang, C. Fang, and J. Luo. Image captioning with semantic attention. *CVPR*, 2016.

[54] J. G. Zilly, R. K. Srivastava, J. Koutník, and J. Schmidhuber. Recurrent highway networks. *arXiv preprint arXiv:1607.03474*, 2016.