StructCap: Structured Semantic Embedding for Image Captioning

Fuhai Chen[1], Rongrong Ji[1*], Jinsong Su[2], Yongjian Wu[3], Yunsheng Wu[3]
[1]Fujian Key Laboratory of Sensing and Computing for Smart City, School of Information Science and Engineering, Xiamen University, 361005, China
[2]the School of Software, Xiamen University, 361005, China
[3]Tencent YouTu Lab
cfh3c@stu.xmu.edu.cn, {rrji,jssu}@xmu.edu.cn, {littlekenwu,simonwu}@tencent.com

ABSTRACT

Image captioning has attracted ever-increasing research attention in multimedia and computer vision. To encode the visual content, existing approaches typically utilize the off-the-shelf deep Convolutional Neural Network (CNN) model to extract visual features, which are sent to Recurrent Neural Network (RNN) based textual generators to output word sequence. Some methods encode visual objects and scene information with attention mechanism more recently. Despite the promising progress, one distinct disadvantage lies in distinguishing and modeling key semantic entities and their relations, which are in turn widely regarded as the important cues for us to describe image content. In this paper, we propose a novel image captioning model, termed StructCap. It parses a given image into key entities and their relations organized in a visual parsing tree, which is transformed and embedded under an encoder-decoder framework via visual attention. We give an end-to-end formulation to facilitate joint training of visual tree parser, structured semantic attention and RNN-based captioning modules. Experimental results on two public benchmarks, Microsoft COCO and Flickr30K, show that the proposed StructCap model outperforms the state-of-the-art approaches under various standard evaluation metrics.

CCS CONCEPTS

•**Computing methodologies** → **Computer vision; Scene understanding;**

KEYWORDS

image captioning; structured learning; visual relation; deep learning

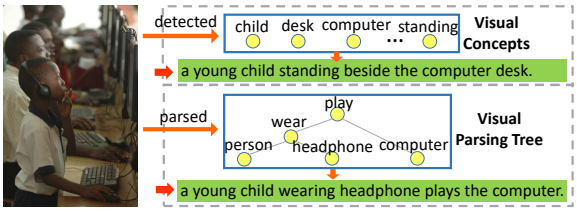

Figure 1: Comparison between two image captioning schemes. The top green row shows an example generated by the traditional end-to-end image captioning. The second green row shows an illustrative example generated by the proposed StructCap scheme. Different from the traditional model, the proposed scheme parses and organizes key visual entities and their relations in a tree structure, which are embedded as visual attention for captioning.

1 INTRODUCTION

Automatic description of an image, *a.k.a.*, image captioning, has recently attracted extensive research attention with broad application prospects. Differing from image classification or object detection, image captioning targets holistical description of objects, scenes, and their relationships presented in a natural sentence. It is a hybrid task that involves the integrated design of cutting-edge techniques in visual scene parsing, content semantic understanding, as well as natural language processing.

Recently there are many new approaches adopted to image captioning [1–8]. Early works are mainly based upon an end-to-end formulation [2, 4]. For instance, the visual features are extracted from the last fully-connected layer of deep Convolutional Neural Networks (CNN), and they are fed into the Recurrent Neural Network (RNN) to output word sequences as captions. However, such a black-box procedure lacks sufficient high-level semantic cues to guide the sentences generation, which is even harder to be interpreted in the high-level semantic space. To address this issue, more recently attention models are introduced into image captioning [6, 8, 9]. For instance, the work in [8] proposed a semantic attention model to link textual generation with visual content using visual entities detected in the image. However, considering the existing works, all identified entities are fed into the attention model in spite of their importance and

*Corresponding Author.

MM '17, October 23–27, 2017, Mountain View, CA, USA

DOI: https://doi.org/10.1145/3123266.3123275

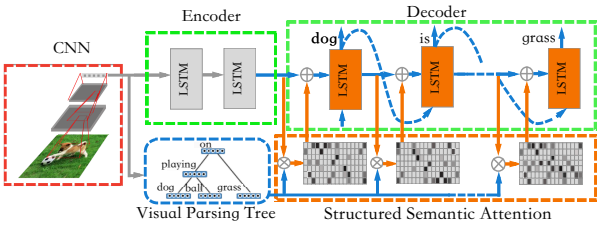

Figure 2: The framework overview of the proposed StructCap model for image captioning. Firstly, the model parses a given image into key entities and their relations organized in a visual parsing tree (VP-Tree). Then the root node of the VP-Tree is embedded into the encoder, and the VP-Tree is transformed to visual attention to participate in forming each state of the LSTM based decoder. Finally, the new word is output conditioned on the previous hidden feature and the current word input.

relations, which is indeed less distinguishable in describing major contents of the image.

In this paper, we argue that only the principled entities are essential for attention embedding during the caption generation. Moreover, their relations, rather than individual entities, also plays an important role in characterizing images. For instance, as shown in Fig. 1, taking no account of mutual relations among entities obtained, the caption generated by equally embedding all entities is less informative than that of proposed scheme. However, discovering and embedding key entities and their relations is still troublesome. On one hand, how to parse key image entities and model their relations remains an open problem. On the other hand, how to organize the parsed entities and relations into a suitable structure is also questionable. In particular, such a structure should be able to transformed into a fixed-dimensional embedding which can be integrated into the attention model for image captioning.

In this paper, we propose a structured semantic embedding model for image captioning, termed *StructCap*. Our main innovation lies in a novel tree structure embedding scheme, which simultaneously models the importance and relations of visual entities into a visual parsing tree that can be easily embedded and transferred to the Long Short-Term Memory (LSTM) based caption generator [2, 10]. In particular, given an image to be captioned, we firstly parse key entities and model/encode their relations as a visual parsing tree, termed *VP-Tree*. This corresponding visual parser is learned by using image-caption correspondences with textual parsing trees in the training set. Secondly, the proposed embedding scheme encodes VP-Tree into LSTM states, which is used as attention in encoder-decoder model [2]. More specially, the correlation score between each node feature of the VP-Tree and the hidden feature of each LSTM state is computed as an attention score to weight the previous hidden feature in this LSTM state. By above scheme, both CNN based visual features and VP-Tree based semantic cues are jointly fed into LSTM state to encode the visual contents for caption generation.

In practice, only limited object detectors are reliably available, as demonstrated in ILSVRC recently [11–14]. To this end, we further propose a simplified version of StructCap, which transfers the dynamic VP-Tree into a fixed structure with 3 layers. In particular, this simplified VP-Tree consists of *Semantic mapper*, *Combiner* and *Categorizer*. The *Semantic mapper* maps the CNN visual feature to the semantic space. The *Combiner* combines the visual entities or their relations into a new representation in the VP-Tree. The *Categorizer* classifies the representation of each node as the categories of entity or relation, which can be trained according to the textual parsing tree.

The contribution of this paper is as follows: 1) We propose a novel visual parsing tree that parses and encodes important visual entities together with their relations of a given image. 2) We propose a novel embedding scheme to project this dynamic parsing tree into a fixed-length structure for visual semantic attention. 3) To deal with the limitation of missing or unreliable object detectors, we present a simplified version of parsing tree, which quantitatively shows promising performance and robustness in practice.

Our algorithm outperforms the state-of-the-art competing methods in image captioning. Our quantitative gains are consistent across different evaluation metrics (Bleu-1,2,3,4, Meteor, Rouge-L, and CIDEr) on two widely used benchmarks, *i.e.*, Microsoft COCO and Flickr 30K.

2 RELATED WORK

Image Captioning. Most existing image captioning methods are inspired by the encoder-decoder framework in machine translation [15, 16]. From this perspective, image captioning is analogous to translating images to texts. [2, 4] proposed a CNN-RNN architecture for image captioning, where CNN encodes visual content and RNN transforms the visual feature to sequential output of words. [3, 7] proposed to use multimodal recurrent convolutional networks (m-RNN) to interact CNN with RNN based sub-netwoks in a multimodal layer. [5] proposed a long-term recurrent convolutional network model spatially and temporally which is flexible for variety of vision tasks. [17] proposed a bidirectional LSTM model that generates descriptive sentence for image by taking both history and future context into account. Based on CNN-RNN architecture, [18] proposed a reference based LSTM model, which leveraged training images as references to improve the quality of captions generated. However, all these approaches can be treated as an end-to-end setting, which lacked high-level guidance of the scene and object cues in the image.

Image Captioning with Attention. More Recently, attention mechanisms have been introduced to the encoder-decoder framework for image captioning. [6] proposed a spatial attention model, which used "hard" pooling to select the most probable attention region, or "soft" pooling that averaged the spatial features with attention weights. [19] proposed an alternative extension of LSTM (gLSTM)

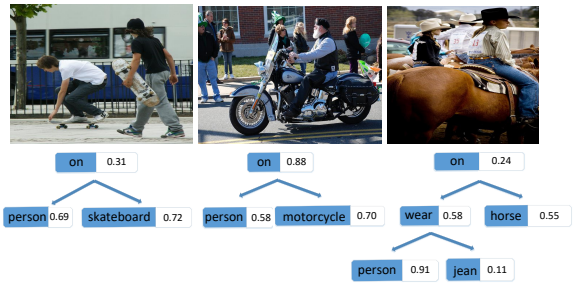

Figure 3: The examples of online parsing results. Taking the images in the upper row as inputs, the proposed model generates corresponding VP-Trees with confidence coefficients of nodes shown in the down row.

by adding semantic information as an extra input to the gate of each LSTM memory cell. [20] proposed a model that aligned the process of generating captions with the attention shifting among the visual regions. [8] proposed a semantic attention model to selectively attend on rich semantic concepts detected from the image. [21] proposed a text-guided attention model, where the visual attention was obtained using guidance captions. [9] proposed a spatial and channel-wise attention-based convolutional neural network model for sequential words generation. However, these methods employed all the entities detected instead of pivotal and related ones, which increases the complexity of the model. Moreover, all these approaches ignored the relations among semantic entities.

Image Parsing. Image parsing is considered to be one of the most fundamental problems in computer vision research. In the literature, most existing works focus on objects and their locations , as well as investigating pixel-level semantic labeling, *i.e.*, semantic segmentation [22–25]. Meanwhile, as a higher-level task, structured scene parsing has also attracted much attention [26–28]. Besides, many recent works exploit the usage of deep neural network models, such as the recurrent neural network model [29] and the fully convolutional network (FCN) [30]. Moreover, the recursive neural networks (RNNs) model are also utilized in [31–33] to conduct hierarchical semantic parsing.

3 THE PROPOSED STRUCTCAP MODEL

In principle, the proposed image caption scheme follows the popular encoder-decoder framework, as shown in Fig. 2. This framework consists of four stages, *i.e.*, deep visual feature extraction, visual parsing tree (VP-Tree) construction, structured semantic attention, and the encoder-decoder based caption generation. In particular, we first feed an image to a pre-trained CNN to obtain deep visual features(s)[1], which is sent to LSTM based encoder and simultaneously used for building the VP-Tree (Subsection 3.1). The built VP-Tree is then transformed into a fixed-length attention to be embedded into LSTM (Subsection 3.2). Finally, both VP-Tree structure and attention-based image captioning model are jointly end-to-end learned (Subsection 3.3).

[1]In the dynamic VP-Tree, we obtain the deep visual features of visual entities detected, while in the simplified VP-Tree, we use the deep global feature of the image.

3.1 Visual Parsing Tree

Given an image, we parse the key entities and their relations into a visual parsing tree (VP-Tree). Its corresponding parser is trained using the image-caption set. In general, this stage consists of both offline parser training and online parsing. In offline training, given image-caption pairs, we first use standard textual parser as introduced in [31] to transform captions into textual parsing trees, upon which visual parser is trained. In online parsing, the trained visual parser transfers a given image into a tree structure, as well as inferring the importances/weights of individual semantic items (including entities and relations) as shown in Fig. 3.

3.1.1 Offline Training. To train the visual parser offline, given an image-caption pair, we employ Faster-RCNN [12] and textual parser [31] to obtain the set of visual entity proposals $\mathcal{S}^p = \{(G_i^p, l_i^p, s_i^p)|i \in \{1, \ldots, K^p\}\}$ and textual parsing tree $\boldsymbol{T}^t$, where G^p, l^p, s^p and K^p denote the deep visual feature, the category label, the confident score of detection, and the number of proposals, respectively. We aim to parse $\mathcal{S}^p$ into a VP-Tree guided by $\boldsymbol{T}^t$. This VP-Tree is denoted by $\boldsymbol{T}^v = \{\boldsymbol{h}_j^n \in \mathbb{R}^{d_n}|j = 1, \ldots, M\}$, where $\boldsymbol{h}_j^n$ denotes the subject, relation or object for the j-th node. $\boldsymbol{h}_{<root>}^n$ denotes the feature of the root node. d_n denotes the dimension of the node feature, and M denotes the number of nodes in the tree. Fig. 4 (left) illustrates an example of the first-layer subtree in our VP-Tree. This subtree consists of five parts: *Proposal Selecter*, *Semantic Mapper*, *Combiner*, *Categorizer* and *Scorer*, which are itemized as follows:

Proposal Selecter. We first select the proposals with scores in top K^{sel}, which outputs a set of selected proposals denoted as $\mathcal{S}^{sel}$.

Semantic Mapper. The deep feature of each proposal is then mapped onto a semantic space by the *Semantic Mapper*:

$$\boldsymbol{h}_{<e>}^n = F^{sem}(G^p; \boldsymbol{W}^{sem}), \tag{1}$$

where $\boldsymbol{h}_{<e>}^n$ is the mapped feature of the entity (leaf node of VP-Tree), F^{sem} is the network transformation, and $\boldsymbol{W}^{sem}$ is the parameter of semantic mapping for the leaf node.

Combiner. The features of two child nodes are fed to the *Combiner* and generate the feature of their parent node as:

$$\boldsymbol{h}_{<r>}^n = F^{com}([\boldsymbol{h}_{<lc>}^n, \boldsymbol{h}_{<rc>}^n]; \boldsymbol{W}^{com}), \tag{2}$$

where $[\boldsymbol{h}_{<lc>}^n, \boldsymbol{h}_{<rc>}^n]$ indicates the feature concatenation operation for the left child and the right child. $\boldsymbol{h}_{<r>}^n$ denotes the corresponding feature of the relation (parent node of VP-Tree). F^{com} and $\boldsymbol{W}^{com}$ are the network transformation and the parameter of *Combiner*. The feature of the parent node has the same dimensionality as its child nodes, leaving the procedure recursive.

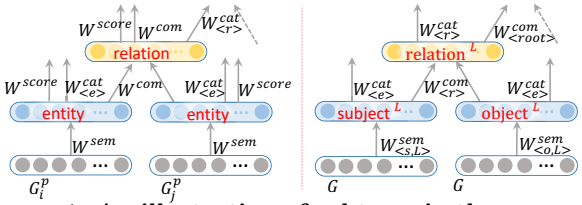

Figure 4: An illustration of subtrees in the proposed VP-Trees. The left is a dynamic VP-Tree and the right is a simplified VP-Tree.

Categorizer. The *Categorizer* determines the entity or relation category of a node, which uses a softmax classifier to map the node feature $\boldsymbol{h}^n$ to the label space:

$$\boldsymbol{y}^n_{<\cdot>} = softmax\big(F^{cat}_{<\cdot>}(\boldsymbol{h}^n_{<\cdot>}; \boldsymbol{W}^{cat}_{<\cdot>})\big), \tag{3}$$

where $\boldsymbol{y}^n_{<\cdot>}$ denotes the predicted probability vectors of entity or relation categories according to a entity dictionary or relation dictionary[2], respectively. $F^{cat}_{<\cdot>}$ denotes the corresponding network transformation and $\boldsymbol{W}^{cat}_{<\cdot>}$ denotes the parameter. We employ cross entropy to define the loss function of *Categorizer*, denoted as J^{cat}.

Scorer. The confidence of the p-th combination operation between the two child nodes can be measured by *Scorer* using the parent feature $\boldsymbol{h}^n_{<r>}$, written as:

$$s^n(p) = F^{score}(\boldsymbol{h}^n_{<r>}; \boldsymbol{W}^{score}). \tag{4}$$

As the objective in training, we keep the correct VP-Tree $\boldsymbol{T}^v_{corr}$ (the combinations in each layer are based on textual parsing tree $\boldsymbol{T}^t$) with higher score than the VP-Tree 2-nd highest score (the combinations in each layer are based on highest-score selection). Firstly, suppose we make the p-th possible combination at each step and in total we have K^{op} combinations according to textual parsing tree. At each step, we collect the subtree set $\mathcal{S}^{all}_p = \{\hat{\tau}_i | i \in \{1, \ldots, K^{all}_p\}\}$ and the subtree set $\mathcal{S}^{corr}_p = \{\tau_i | i \in \{1, \ldots, K^{corr}_p\}\} \in \mathcal{S}^{all}$ of the all and the correct tree $\boldsymbol{T}^v_{corr}$, correspondingly. Max-margin estimation is employed to train the model to select the highest scoring subtree in the correct VP-Tree. As above, the score of the correct subtree needs to climb a certain margin. The loss function for the structure of VP-Tree can be written as follows:

$$\begin{aligned} J^{struct}(\boldsymbol{\theta}^{struct}; \mathcal{S}^{sel}, \boldsymbol{T}^t) = \frac{1}{K^{op}} \sum_{p}^{K^{op}} [\max_{\hat{\tau}_i \in \mathcal{S}^{all}_p} (s^n(\hat{\tau}_i)) \\ + \Delta(p) - \max_{\tau \in \mathcal{S}^{corr}_p} (s^n(\tau_i))] + \frac{\lambda}{2} \|W\|^2, \end{aligned} \tag{5}$$

where $\boldsymbol{\theta}^{struct}$ is the parameters of tree structure including $\boldsymbol{W}^{sem}$, $\boldsymbol{W}^{com}$ and $\boldsymbol{W}^{score}$, and $s^n(p)$ can be computed via Eq. 4, and the Δ is a constant margin, which can be written as:

$$\Delta(p) = \eta \sum_{\hat{\tau}_i \in \mathcal{S}^{all}_p} \mathbf{1}\{\hat{\tau}_i \notin \mathcal{S}^{corr}_p\}. \tag{6}$$

[2]We first select entities and relations with high frequencies and then similar entities or relations are categorized into one class, *e.g.*, *man* and *woman* belong to *person*. Finally we obtain the entity dictionary and the relation dictionary respectively.

Finally, we formulate the loss function of VP-Tree, which consists of both the category classifying loss and the structure selection loss as follows:

$$\begin{aligned} J(\boldsymbol{\theta}^T; \mathcal{S}^{sel}_i, \boldsymbol{T}^t_i) = \frac{1}{N} \sum_{i=1}^{N} \big(J^{cat}(W^{cat}_{<e>}; \mathcal{S}^{sel}_i, \boldsymbol{T}^t_i) \\ + J^{cat}(W^{cat}_{<r>}; \mathcal{S}^{sel}_i, \boldsymbol{T}^t_i) \\ + J^{struct}(\boldsymbol{\theta}^{struct}; \mathcal{S}^{sel}, \boldsymbol{T}^t) \big), \end{aligned} \tag{7}$$

where the parameter set of VP-Tree is $\boldsymbol{\theta}^T = \{\boldsymbol{\theta}^{struct}; W^{cat}_{<e>}; W^{cat}_{<r>}\}$, and N denotes the number of image-caption pairs. It is noted that the offline training of VP-Tree is integrated with the overall model training, along with the features of nodes attended to the LSTM based decoder for generating sequential words. To this end, we employ the greedy approximation [31] to select an effective tree configuration in each step and use L-BFGS to minimize the objective in each iteration.

3.1.2 Online Parsing. In online parsing, given an image, Fatser-RCNN is also employed to detect proposals $\mathcal{S}^p$. Now we aim to transform the proposal set $\mathcal{S}^p$ into a VP-Tree. We first employ the *Proposal Selecter* to select proposals with high confidence scores. Then the *Semantic Mapper* is used to map the deep visual features to the semantic features of entities in Eq. 1. Each two semantic features are combined by the *Combiner* to obtain semantic feature of relations as Eq. 2. The semantic features of relations are sent to the *Categorizer* and the *Scorer* to obtain the category labels and confidence scores in Eq. 3 and Eq. 4, respectively. The semantic features of relations with high confidence scores will be also subsequently sent to be combined with unused semantic features. We repeat the last step to enlarge the semantic tree structure until the scores reach the upper limit.

3.1.3 Simplified Visual Parsing Tree. As discussed above, the performance of existing object detectors is less effective as demonstrated in the ILSVRC challenge [11]. As a result, we transfer the dynamic VP-Tree into a fixed structure with 3 layers. Then we replace the detected proposals with visual global feature to construct the VP-Tree. This simplified VP-Tree only consists of *Semantic Mapper*, *Combiner* and *Categorizer* as shown in the right subfigure of Fig. 4. These *Semantic Mapper* and *Combiner* are different from the ones of dynamic VP-Tree, and they are itemized as follows:

Semantic Mapper. Visual global feature G extracted from the last fully-connected layer of CNN is mapped onto a semantic space by *Semantic Mappers*. The mappers are divided into subject mapping and object mapping. Here, we take the subject mapping in left subtree as example:

$$\boldsymbol{h}^n_{<s,L>} = F^{sem}(G; \boldsymbol{W}^{sem}_{<s,L>}), \tag{8}$$

where $\boldsymbol{h}^n_{<s,L>}$ is the mapped feature of the subject node in the left subtree, F^{sem} is the network transformation and $\boldsymbol{W}^{sem}_{<s,L>}$ denotes the parameter of the semantic mapping for the subject node in the left subtree.

Combiner. The features of two child nodes are fed to the *Combiner* and generate their parent node feature as follows:

$$\boldsymbol{h}^n_{<r,*>} = F^{com}([\boldsymbol{h}^n_{<s,*>}, \boldsymbol{h}^n_{<o,*>}]; \boldsymbol{W}^{com}_{<r>}), \quad (9)$$

$$\boldsymbol{h}^n_{<r,root>} = F^{com}([\boldsymbol{h}^n_{<r,L>}, \boldsymbol{h}^n_{<r,R>}]; \boldsymbol{W}^{com}_{<root>}), \quad (10)$$

where $[\cdot,\cdot]$ indicate the concatenation operation of two child features. $\boldsymbol{h}^n_{<s,*>}$, $\boldsymbol{h}^n_{<o,*>}$ and $\boldsymbol{h}^n_{<r,*>}$ denotes the subject feature, object feature, and relation feature respectively in the left subtree 'L' or the right subtree 'R'. $\boldsymbol{h}^n_{<r,root>}$ denotes the relation feature of root node. F^{com} is the network transformation. $\boldsymbol{W}^{com}_{<r>}$ and $\boldsymbol{W}^{com}_{<root>}$ are the corresponding parameters of the *Combiner*.

The parameter set of VP-Tree can be denoted as $\boldsymbol{\theta}^T = \{\boldsymbol{W}^{sem}_{<s,*>}; \boldsymbol{W}^{sem}_{o,*}; \boldsymbol{W}^{com}_{<r>}; \boldsymbol{W}^{com}_{<root>}; \boldsymbol{W}^{cat}_{<e>}; \boldsymbol{W}^{cat}_{<r>}\}$, where s, o, r and e mean subject, object, relation and entity (we use 'e' to joint 's' and 'o'), respectively. '*' means left subtree ('L') or right subtree ('R'). In the simplified VP-Tree, we only minimize the loss of the *categorizer* to optimize the whole tree model. The offline training of simplified VP-Tree is also integrated into the overall model training, which attends the features of nodes to the LSTM-based decoder for sequential words generation.

3.2 Structured Semantic Attention

The generated VP-Tree is embedded into each state of the decoder as attention. In the t-th LSTM state of the decoder, we first compute the vector of relevance scores between the feature of each tree node $\boldsymbol{h}^n$ (the feature $\boldsymbol{h}^n_{<e>}$ of the leaf node for predicting entities, or the feature $\boldsymbol{h}^n_{<r>}$ of the non-leaf node for predicting relations) and the previous hidden feature $\boldsymbol{h}^s_{t-1}$:

$$\boldsymbol{A}^{<\cdot>}_{j,t-1} = \frac{exp((\boldsymbol{h}^n_{<\cdot>,j})^T \boldsymbol{U}^{<\cdot>} \boldsymbol{h}^s_{t-1})}{\sum_{k=1}^{M^{<\cdot>}} exp((\boldsymbol{h}^n_{<\cdot>,k})^T \boldsymbol{U}^{<\cdot>} \boldsymbol{h}^s_{t-1})}, \quad (11)$$

where $\boldsymbol{U}^{<\cdot>}$ denotes the bilinear parameter matrices for entities or relations, respectively. $M^{<\cdot>}$ is the number of entity or relation in the VP-Tree. Then the corresponding hidden attention features in the t-th state can be computed by summing the features of tree nodes, weighted by corresponding relevance scores:

$$\boldsymbol{h}^a_{<\cdot>,t-1} = \sum_{j=1}^{M^{<\cdot>}} \boldsymbol{A}^{<\cdot>}_{j,t} \boldsymbol{h}^{n,<\cdot>}_j. \quad (12)$$

Finally, we concatenate $\boldsymbol{h}^a_{<e>,t-1}$, $\boldsymbol{h}^a_{<r>,t-1}$ and $\boldsymbol{h}^s_{t-1}$, and project the concatenation to obtain $\boldsymbol{h}^c_{t-1}$ as the previous hidden feature of the t-th state:

$$\boldsymbol{h}^c_{t-1} = \tanh(\boldsymbol{W}^c[\boldsymbol{h}^a_{<e>,t-1}, \boldsymbol{h}^a_{<r>,t-1}, \boldsymbol{h}^s_{t-1}] + \boldsymbol{b}^c), \quad (13)$$

where $\boldsymbol{W}^c$ and $\boldsymbol{b}_c$ are the weight parameter matrix and the bias parameter vector of the concatenation operation in the structured attention model, respectively. The hidden feature from the $t-1$ state is updated and taken as the input of the current LSTM state.

3.3 Model Learning

Following [18], we integrate the structure similarity for word weighting by using the textual parsing tree. Formally, the log-likelihood consisting of sequential output and semantic tree construction can be written as follows:

$$\begin{aligned} L(\boldsymbol{\theta}) = &\sum_i^N \sum_t^T w(y^i_t, G^i, \boldsymbol{T}^{t,i}) \log P(\boldsymbol{y}^i_t | G^i, \boldsymbol{y}^i_{0:t-1}; \boldsymbol{\theta}) \\ &+ \lambda \sum_i^N \sum_j^M \log P(\boldsymbol{y}^{n,i}_j | G^i; \boldsymbol{\theta}_T), \end{aligned} \quad (14)$$

where G and $\boldsymbol{T}^t$ denote the visual global feature and the textual parsing tree, respectively. $w(y_t, G, \boldsymbol{T}^t)$ denotes the weight of the t-th word to G and $\boldsymbol{T}^t$. Following the tag ranking approach [34], $w(y_t, G, \boldsymbol{T}^t)$ can be calculated as:

$$w(y_t, G, \boldsymbol{T}^t) = \frac{\beta p(y_t|G, \boldsymbol{T}^t)}{p(y_t)}, \quad (15)$$

where β is a normalization parameter to ensure the average of all weights is 1. $p(y_t|G, \boldsymbol{T}^t)$ is divided by $p(y_t)$ due to the existence of frequent but uninformative words, such as "a" and "the". Based on Bayesian rule, we have

$$w(y_t, I, \boldsymbol{T}^t) = \frac{\beta P(G, \boldsymbol{T}^t|y_t) P(y_t)}{P(G, \boldsymbol{T}^t) p(y_t)} = \beta P(G|y_t) P(\boldsymbol{T}^t|y_t). \quad (16)$$

Then we adopt kernel density estimation (KDE) [35] to have

$$P(G|y_t) = \frac{1}{|S_{y_t}|} \sum_{G_j \in S_{y_t}} K_\sigma(\boldsymbol{G} - \boldsymbol{G}_j), \quad (17)$$

where S_{y_t} denotes the set of samples whose captions contain word y_t. K_σ denotes the Gaussian kernel function. We also employ $Bleu_3$ score [36] to compute the similarity of textual parsing trees for $P(\boldsymbol{T}^t|y_t)$ in the similar way to Eq. 17 after flattening the trees to sentences[3].

We minimize the log-likelihood and back propagate the gradient over the decoder and encoder model. The stochastic gradient descent algorithm is used with an adaptive learning rate. We also use dropout and early stopping to avoid overfitting. The iteration ends until the cost of final word prediction converges.

4 EXPERIMENTS

4.1 Datasets and settings

Datasets and Evaluation Protocols. We choose the popular Flickr30k and MS-COCO to evaluate the performance of our models. We use publicly available splits[4] of training, testing and validating sets for both Flickr30K and MS-COCO. There are over 31,000 images in Flickr30K with 29,000 images for training, 1,000 images for testing, and 1,014 images for valuating. There are over 123,000 images in MS-COCO with 82,738 images for training, 5,000 images for testing, and 5,000 images for valuating. Each image in these two datasets is given at least five captions by different AMT workers.

[3]We use $Bleu_3$ due to the structure of complete binary tree. Note that we don't directly use captions for textual similarity due to the existence of uninformative words and various syntaxes.

[4]https://github.com/karpathy/neuraltalk

Table 2: Performance comparison to the state-of-the-art methods on Flickr30k and MS-COCO. The numbers in bold face are the best known results and (-) indicates unknown scores. We choose StructCap-T-A-W as the default of StructCap model.

	Flickr30K					MS-COCO				
Model	B-1	B-2	B-3	B-4	METEOR	B-1	B-2	B-3	B-4	METEOR
BRNN [4]	0.573	0.369	0.24	0.157	-	0.625	0.45	0.321	0.23	0.195
LRCN [5]	0.587	0.39	0.25	0.165	-	0.628	0.442	0.304	0.21	-
Google NIC [2]	0.663	0.423	0.277	0.183	-	0.666	0.451	0.304	0.203	-
m-RNN [3]	0.60	0.41	0.28	0.19	-	0.67	0.49	0.35	0.25	-
Log Bilinear [37]	0.600	0.380	0.254	0.171	0.169	0.708	0.489	0.344	0.243	0.239
Toronto [6]	0.669	0.439	0.296	0.199	0.185	0.718	0.504	0.357	0.250	0.230
ATT [8]	0.647	0.460	0.324	0.230	0.189	0.709	0.537	0.402	0.304	0.243
StructCap(w/o ensemble)	0.704	0.509	0.377	0.279	0.206	0.726	0.563	0.430	0.329	0.254
StructCap(w/ ensemble)	**0.720**	**0.513**	**0.386**	**0.285**	**0.213**	**0.738**	**0.574**	**0.439**	**0.335**	**0.261**

Table 3: Quantitative comparisons to the state-of-the-art works in image captioning on dataset c5 and c40 evaluated on the online MS-COCO server. The numbers in bold face are the best known results.

	B-1		B-2		B-3		B-4		METEOR		ROUGE-L		CIDEr	
Model	c5	c40	c5	c40	c5	c40	c5	c40	c5	c40	c5	c40	c5	c40
StructCap	0.724	**0.907**	0.558	**0.822**	0.423	**0.717**	**0.320**	**0.607**	0.251	0.340	0.534	**0.684**	**0.953**	0.948
Google NIC	0.713	0.895	0.542	0.802	0.407	0.694	0.309	0.587	**0.254**	**0.346**	0.530	0.682	0.943	0.946
m-RNN	0.716	0.890	0.545	0.798	0.404	0.687	0.299	0.575	0.242	0.325	0.521	0.666	0.917	0.935
LRCN	0.718	0.895	0.548	0.804	0.409	0.695	0.306	0.585	0.247	0.335	0.528	0.678	0.921	0.934
ATT	**0.731**	0.901	**0.565**	0.816	**0.424**	0.710	0.316	0.600	0.251	0.336	**0.535**	0.683	0.944	**0.959**
SCA-CNN	0.712	0.894	0.542	0.802	0.404	0.691	0.302	0.579	0.244	0.331	0.524	0.674	0.912	0.921

Table 1: Performance comparison of alternative approaches. "-T", "-A" and "-W" represent the proposed StructCap model with VP-Tree, with structured semantic attention, and with structured word weighting, respectively.

	MS-COCO				
Model	B-1	B-2	B-3	B-4	METEOR
original	0.666	0.451	0.304	0.203	-
StructCap-T	0.681	0.480	0.339	0.242	0.245
StructCap-T-A	0.712	0.552	0.421	0.323	0.252
StructCap-T-A-W	0.726	0.563	0.430	0.329	0.254

Quantitative performance of all methods are evaluated by using microsoft COCO caption evaluation tool[5], including BLEU, Meteor, Rouge-L and CIDEr [38].

Parameter Settings. To build our VP-Tree, we set $d = 300$ for the hidden features of nodes. We employ Long-Short Term Memory (LSTM) network [2] as the basic unit of encoder-decoder. The dimensionality of the input and hidden vector is set to 300 and 512, respectively. We use *tanh()* as the nonlinear activation function. We adopt Adam algorithm [39] to optimize our model with the learning rate 0.0001, and the minibatch size is set to 64. We adopt the widely-used ResNet-152 [40] as the CNN architecture. In testing, a caption is formed by drawing words from RNN until a special end token is reached. We follow the ensemble way [8] and BeamSearch [2] to generate a sentence given an image. The ensemble number and the beam size are both set to 5.

Preprocessing on Textual Parsing Trees. Due to the irrelevant words and noise configurations in the parsing trees generated by Stanford Parser [31], we whiten the source sentences simultaneously by using the pos-tag tool and the lemmatizer tool in NTLK [41]. After pruning, we convert the dynamic parsing trees to a fixed-structured, three-layer, complete binary tree, which only contains nouns, verbs, coverbs, prepositions, and conjunctions. Only nouns are regarded as entities and used as leaf nodes in the semantic tree. We select the frequency words automatically and get the original entity dictionary and the relation dictionary. We also manually merge words with similar meaning to obtain the entity dictionary and the relation dictionary with size 748 and 246, respectively.

Baselines and State-of-the-Art. We compare the proposed StructCap with three baselines for caption generation. 1) StructCap-T : Our model only with VP-Tree. 2) StructCap-T-A : Our model with VP-Tree and structured semantic attention. 3) StructCap-T-A-W : Our model with VP-Tree and structured semantic attention training by structured word weighting. We also compare the state-of-the-art method, i.e., ATT [8], which utilizes visual detected concepts as the semantic attention.

4.2 Alternative Approaches

To evaluate the effectiveness of our structured semantic attention, we compare the original model (Google NIC [2]) to

[5]https://github.com/tylin/coco-caption

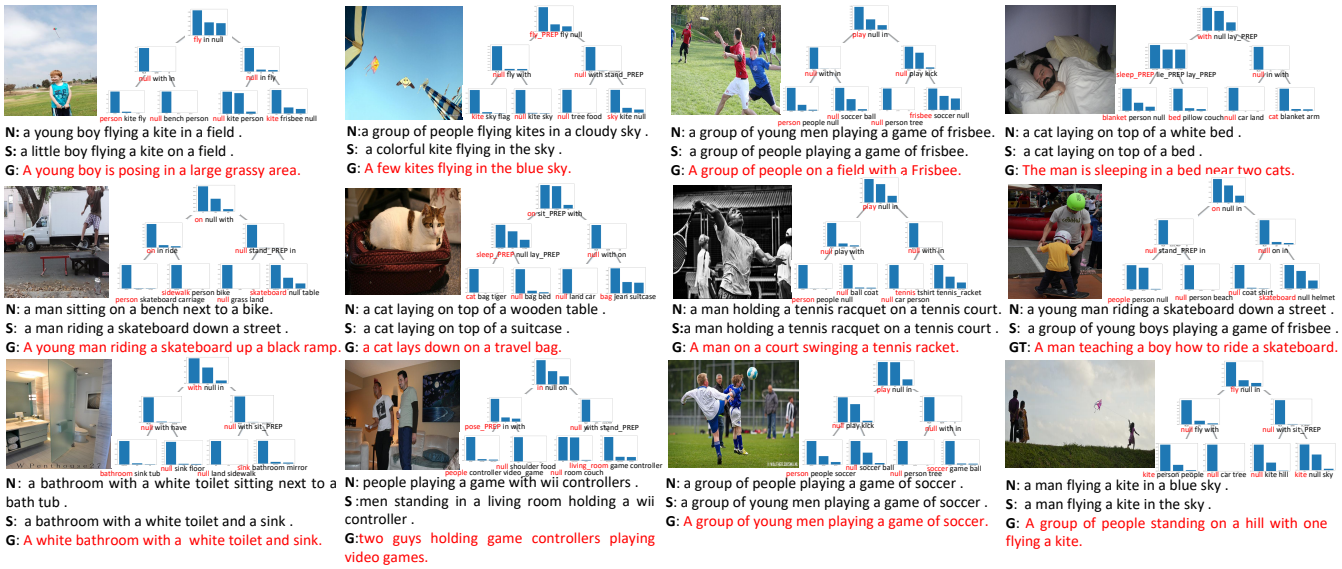

Figure 5: Visualization of the generated results on MS-COCO dataset. "N", "S" and "G" denote the original model (Google NIC [2]), the proposed scheme StructCap, and the ground truth, respectively. Each VP-Tree is constructed on the right of the corresponding image (bad VP-Trees are listed in the last column). Each node of the VP-Tree is presented by a histogram, which shows top-3 probability values of categories. The category with maximum probability value is on each histogram.

deploy our structured attention upon. The quantitative internal comparison results on MS-COCO are shown in Tab. 1. We can see that our proposed models outperformed Google NIC model, which indicates the structured tree and semantic structured attention can both improve the performance of image captioning. Moreover, the "StructCap-T-A-W" model obtains the best performance, which indicates the effectiveness of weighted training with structured references. Specifically, our proposed models gradually widen the advantages from BLEU-1 (1-gram) metric to BLEU-4 (4-gram) metric, which indicates that our models focus more on the semantic continuity of the generated sentence.

4.3 Performance on MS-COCO and Flickr30k

On two public benchmarks, Microsoft COCO and Flickr30K, we evaluate our model compared with the recent methods in Tab. 2. We can find that our StructCap model achieves best performance under all metrics over all benchmarks. Specially, the StructCap model significantly outperforms the state-of-the-art (ATT) under all metrics. We can also find that our model with ensemble way can achieve better performance, which can be due to more randomness and possibility.

We also compare our model to the recent results on the COCO evaluation server[6] in Tab. 3. We can see that our approach achieves the best performance on most metrics among the published systems. Note that the reason why we cannot surpass ATT under some metrics is that it well integrates the external detection models (trained on external datasets like ImageNet). However, as a holistic model, StructCap can still achieve comparative results under these metrics.

4.4 Visualization of Qualitative Results

Our proposed model is based on VP-Tree, which plays an important role for generating captions. We visualize the VP-Tree and the generated caption for each image in Fig. 5, where we can find that the VP-Tree is almost consistent with the image and the generated caption. Moreover, The results show that StructCap outperforms the Google NIC in most examples, which indicates the VP-Tree contributes to the final caption generation. However, there are also some bad VP-Trees, which are listed in the last column. In the first VP-Tree of the last column, the first subject is recognized as *blanket* instead of *person*. This may be because the most part of the visual object is shielded by the other. In the second VP-Tree of the last column, most nodes are recognized to be *null*[7] instead of specific categories, which indicates that the capacity of StructCap is limited due to the lack of some categories with low frequencies in the dictionaries, *e.g.*, *teaching*. In the third VP-Tree of the last column, *kite* is in front of *fly* in the semantic sequence, which indicates the shortage of StructCap to learn the semantic order.

4.5 Analysis of Structured Attention

We visualize the attention weights (relevance scores of attention as Eq. 11) of some examples to validate the effect of the proposed structured attention on the caption generation. As showed in Fig. 6, attention weights are vary properly

[6] http://mscoco.org/dataset/#captions-leaderboard

[7] *Null* in the VP-Tree means the ignorable node or the unknown category (not in the entity or relation dictionaries).

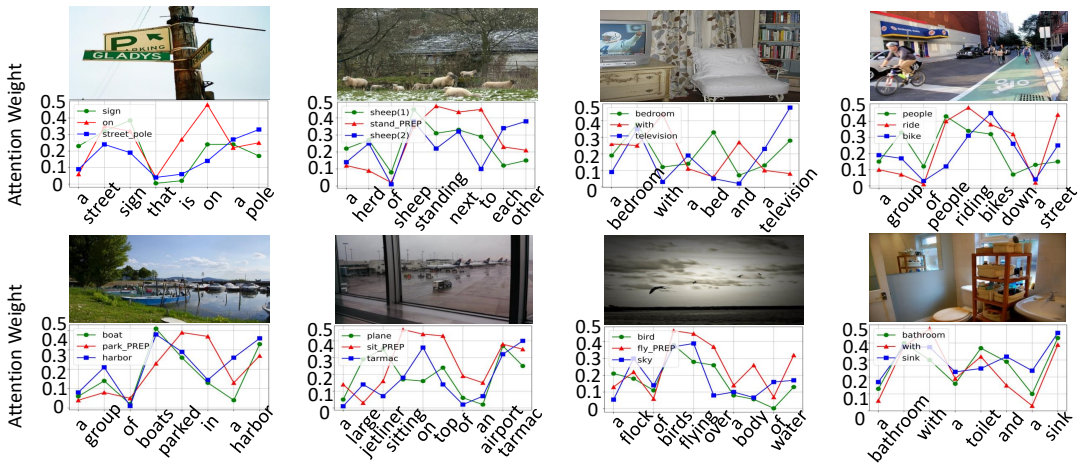

Figure 6: Visualizations of attention weights changing along with the generation of captions. The generated captions are shown under the horizontal time axis of the curve plots, and each word is positioned at time step it is generated. For simplicity, we only show the attention weights of top semantic items (entities or relations) from the generated caption.

as sentence context changes, while the coverages of different semantic items (entities or relations) properly reflect the semantic structure of captions. For instance, in the first example of Fig. 6, the attention on the semantic item *on* peaks after word *sign* is generated. Not only word *sign* but word *street* attracts strong attention on the semantic item *sign*, since the street is high relevant to the sign in the dataset.

4.6 Analysis of Structured Weighting

In our training phase, the words are weighted according to their relevance to images (termed *I-Ref*) and structured textual parsing trees (termed *S-Ref*), which enable the model to focus on the key information of the captions. In this part, we present some examples of word weighting results in Fig. 7. Each example contains a ground true caption, an image, a *I-Ref* histogram, and a *I-S-Ref* (*I-Ref* and *S-Ref*) histogram. Each histogram shows weight values of the words selected based on the weight values within top-5. We can find that the selected words by *I-Ref* and *S-Ref* are both highly relevant to main semantic contents in ground true captions. Especially, word weighting in our *I-S-Ref* way assigns words more reasonable weights than in *I-Ref* way.

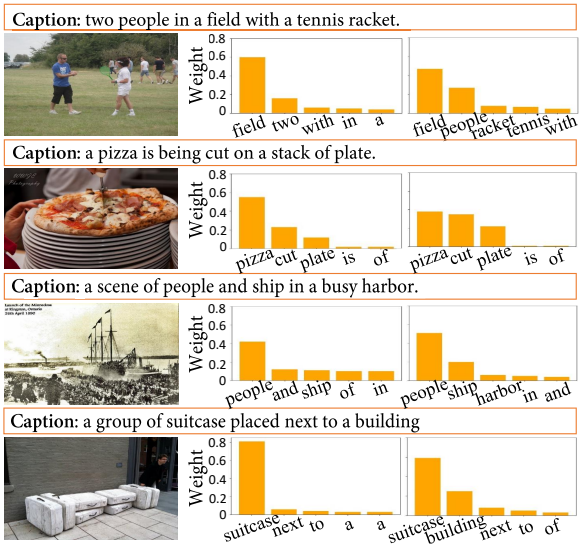

Figure 7: Some word weighting results. Each histogram shows the words and their weights in top-5. The histograms in the middle column show the results of *I-Ref*. The histograms in the right column show the results of *I-S-Ref*.

5 CONCLUSION

In this paper, we propose a novel structured semantic embedding model for image captioning (StructCap). Different from previous works, our model takes key visual entities and their relations into consideration for characterizing images. In our tree structure embedding scheme, the semantic entities and their relations are modeled into a holistic visual parsing tree (VP-Tree), which can be embedded into LSTM based caption generator. In particular, given an image, we firstly parse key entities and model/encode their relations as a VP-Tree. This corresponding visual parser is learned by using image-caption correspondences with textual parsing trees in the training set. Secondly, the proposed embedding scheme encodes VP-Tree into LSTM states, which is used as attention in the encoder-decoder model. We give an end-to-end formulation to facilitate joint training of visual tree parser, structured semantic attention and RNN-based captioning modules. Abundant experimental evaluations show that our model achieves state-of-the-art performance across popular standard benchmarks under various standard evaluation metrics.

6 ACKNOWLEDGEMENTS

This work is supported by the National Key R&D Program (No. 2016YFB1001503), and the Nature Science Foundation of China (No. 61422210, No. 61373076, No. 61402388, and No. 61572410).

REFERENCES

[1] Xinlei Chen and C Lawrence Zitnick. Mind's eye: A recurrent visual representation for image caption generation. In *CVPR*, pages 2422–2431, 2015.

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

[3] Junhua Mao, Wei Xu, Yi Yang, Jiang Wang, Zhiheng Huang, and Alan Yuille. Deep captioning with multimodal recurrent neural networks (m-rnn). *arXiv preprint arXiv:1412.6632*, 2014.

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

[5] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *CVPR*, pages 2625–2634, 2015.

[6] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C Courville, Ruslan Salakhutdinov, Richard S Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*, volume 14, pages 77–81, 2015.

[7] Junhua Mao, Xu Wei, Yi Yang, Jiang Wang, Zhiheng Huang, and Alan L Yuille. Learning like a child: Fast novel visual concept learning from sentence descriptions of images. In *ICCV*, pages 2533–2541, 2015.

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

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

[10] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[11] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[12] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, pages 91–99, 2015.

[13] Xingyu Zeng, Wanli Ouyang, Bin Yang, Junjie Yan, and Xiaogang Wang. Gated bi-directional cnn for object detection. In *ECCV*, pages 354–369. Springer, 2016.

[14] Bin Yang, Junjie Yan, Zhen Lei, and Stan Z Li. Craft objects from images. In *CVPR*, pages 6043–6051, 2016.

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

[16] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *NIPS*, pages 3104–3112, 2014.

[17] Cheng Wang, Haojin Yang, Christian Bartz, and Christoph Meinel. Image captioning with deep bidirectional lstms. In *ACM MM*, pages 988–997. ACM, 2016.

[18] Minghai Chen, Guiguang Ding, Sicheng Zhao, Hui Chen, Jungong Han, and Qiang Liu. Reference based lstm for image captioning. 2017.

[19] Xu Jia, Efstratios Gavves, Basura Fernando, and Tinne Tuytelaars. Guiding the long-short term memory model for image caption generation. In *ICCV*, pages 2407–2415, 2015.

[20] Junqi Jin, Kun Fu, Runpeng Cui, Fei Sha, and Changshui Zhang. Aligning where to see and what to tell: image caption with region-based attention and scene factorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPMI)*, 2016.

[21] Jonghwan Mun, Minsu Cho, and Bohyung Han. Text-guided attention model for image captioning. 2017.

[22] Xiaofeng Ren and Jitendra Malik. Learning a classification model for segmentation. In *ICCV*, volume 1, pages 10–17, 2003.

[23] Brian Fulkerson, Andrea Vedaldi, and Stefano Soatto. Class segmentation and object localization with superpixel neighborhoods. In *ICCV*, pages 670–677. IEEE, 2009.

[24] Mohammadreza Mostajabi, Payman Yadollahpour, and Gregory Shakhnarovich. Feedforward semantic segmentation with zoom-out features. In *CVPR*, pages 3376–3385, 2015.

[25] Joseph Tighe, Marc Niethammer, and Svetlana Lazebnik. Scene parsing with object instances and occlusion ordering. In *CVPR*, pages 3748–3755, 2014.

[26] Zhuowen Tu, Xiangrong Chen, et al. Image parsing: Unifying segmentation, detection, and recognition. In *ICCV*, pages 18–25. IEEE, 2003.

[27] Feng Han and Song-Chun Zhu. Bottom-up/top-down image parsing with attribute grammar. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPMI)*, 31(1):59–73, 2009.

[28] Long Zhu, Yuanhao Chen, Yuan Lin, Chenxi Lin, and Alan Yuille. Recursive segmentation and recognition templates for image parsing. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPMI)*, 34(2):359–371, 2012.

[29] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip HS Torr. Conditional random fields as recurrent neural networks. In *ICCV*, pages 1529–1537, 2015.

[30] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, pages 3431–3440, 2015.

[31] Richard Socher, Cliff C Lin, Chris Manning, and Andrew Y Ng. Parsing natural scenes and natural language with recursive neural networks. In *ICML*, pages 129–136, 2011.

[32] Abhishek Sharma, Oncel Tuzel, and David W Jacobs. Deep hierarchical parsing for semantic segmentation. In *CVPR*, pages 530–538, 2015.

[33] Liang Lin, Guangrun Wang, Rui Zhang, Ruimao Zhang, Xiaodan Liang, and Wangmeng Zuo. Deep structured scene parsing by learning with image descriptions. In *CVPR*, pages 2276–2284, 2016.

[34] Dong Liu, Xian-Sheng Hua, Linjun Yang, Meng Wang, and Hong-Jiang Zhang. Tag ranking. In *WWW*, pages 351–360. ACM, 2009.

[35] Emanuel Parzen. On estimation of a probability density function and mode. *The annals of mathematical statistics*, 33(3):1065–1076, 1962.

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

[37] Ryan Kiros, Ruslan Salakhutdinov, and Richard S Zemel. Multimodal neural language models. In *ICML*, volume 14, pages 595–603, 2014.

[38] Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*, 2015.

[39] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

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

[41] Steven Bird. Nltk: the natural language toolkit. In *COLING/ACL*, pages 69–72. Association for Computational Linguistics, 2006.