# Learning to Compose Dynamic Tree Structures for Visual Contexts

Kaihua Tang[1], Hanwang Zhang[1], Baoyuan Wu[2], Wenhan Luo[2], Wei Liu[2]
[1] Nanyang Technological University
[2] Tencent AI Lab
*kaihua001@e.ntu.edu.sg*, *hanwangzhang@ntu.edu.sg*,
{*wubaoyuan1987, whluo.china*}*@gmail.com*, *wl2223@columbia.edu*

## Abstract

*We propose to compose dynamic tree structures that place the objects in an image into a visual context, helping visual reasoning tasks such as scene graph generation and visual Q&A. Our visual context tree model, dubbed* VCTREE, *has two key advantages over existing structured object representations including chains and fully-connected graphs: 1) The efficient and expressive binary tree encodes the inherent parallel/hierarchical relationships among objects,* e.g., *"clothes" and "pants" are usually co-occur and belong to "person"; 2) the dynamic structure varies from image to image and task to task, allowing more content-/task-specific message passing among objects. To construct a* VCTREE, *we design a score function that calculates the task-dependent validity between each object pair, and the tree is the binary version of the maximum spanning tree from the score matrix. Then, visual contexts are encoded by bidirectional TreeLSTM and decoded by task-specific models. We develop a hybrid learning procedure which integrates end-task supervised learning and the tree structure reinforcement learning, where the former's evaluation result serves as a self-critic for the latter's structure exploration. Experimental results on two benchmarks, which require reasoning over contexts: Visual Genome for scene graph generation and VQA2.0 for visual Q&A, show that* VCTREE *outperforms state-of-the-art results while discovering interpretable visual context structures.*

## 1. Introduction

Objects are not alone. They are placed in the visual context: a coherent object configuration attributed to the fact that they co-vary with each other. Extensive studies in cognitive science show that our brains inherently exploit visual contexts to understand cluttered visual scenes comprehensively [4, 6, 37]. For example, even the girl's leg and the horse are not fully observed in Figure 1, we can still infer "girl riding horse". Inspired by this, modeling visual contexts is also indispensable in many modern computer vision systems. For example, state-of-the-art CNN architectures capture the context by convolutions of various receptive fields and encode it into multi-scale feature map pyramid [8, 27, 60]. Such pixel-level visual context (or local context [16]) arguably plays one of the key roles in closing the performance gap of the "mid-level" vision between humans and machines, such as R-CNN based object detection [27, 29, 40], instance segmentation [18, 38], and FCN

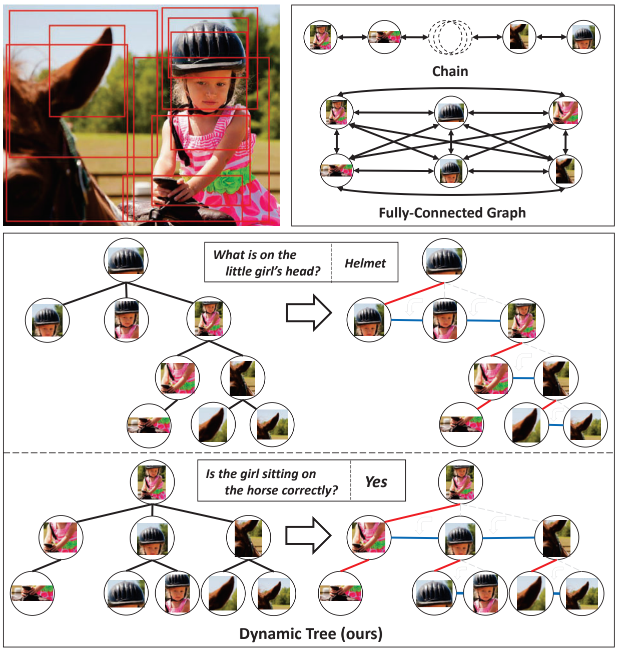
Chain
Fully-Connected Graph
What is on the little girl's head?
Helmet
Is the girl sitting on the horse correctly?
Yes
Dynamic Tree (ours)

Figure 1. Illustrations of different object-level visual context structures: chains [57], fully-connected graphs [50], and dynamic tree structures constructed by the proposed VCTREE. For the purpose of efficient context encoding by using TreeLSTM [44], we transform the multi-branch trees (left) to the equivalent left-child right-sibling binary trees [14], where the left branches (red) indicate the hierarchical relations and right branches (blue) indicate the parallel relations. The key advantages of VCTREE over chains and graphs are hierarchical, dynamic, and efficient.

based semantic segmentation [8, 9, 56].

Modeling visual contexts *explicitly* on the object-level has also been shown effective in "high-level" vision tasks such as image captioning [54] and visual Q&A [46]. In fact, the visual context serves as a powerful inductive bias that connects objects in a particular layout for high-level reasoning [26, 30, 46, 54, 36, 28]. For example, the spatial layout of "person" on "horse" is useful for determining the relationship "ride", which is in turn informative to localize the "person" if we want to answer "who is riding on the horse?". However, those works assume that the context is a scene graph, whose detection *per se* is a high-level task and not yet reliable. Without high-quality scene graphs, we have to use a prior layout structure. As shown in Figure 1, two popular structures are chains [57] and fully-connected graphs [7, 10, 15, 25, 50, 55, 49], where the context is encoded by sequential models such as bidirectional LSTM [19] for chains and CRF-RNN [61] for graphs.

However, these two prior structures are sub-optimal. First, chains are oversimplified and may only capture simple spatial information or co-occurrence bias; though fully-connected graphs are complete, they lack the discrimination between hierarchical relations, *e.g.*, "helmet affiliated to head", and parallel relations, *e.g.*, "girl on horse"; in addition, dense connections could also lead to message passing saturation in the subsequent context encoding [50]. Second, visual contexts are inherently content-/task-driven, *e.g.*, the object layouts should vary from content to content, question to question. Therefore, fixed chains and graphs are incompatible with the dynamic nature of visual contexts [47].

In this paper, we propose a model dubbed VCTREE, pioneering to compose dynamic tree structures for encoding object-level visual context for high-level visual reasoning tasks, such as scene graph generation (SGG) and visual Q&A (VQA). Given a set of object proposals in an image (*e.g.*, obtained from Faster-RCNN [40]), we maintain a trainable task-specific score matrix of the objects, where each entry indicates the contextual validity of the pairwise objects. Then, a maximum spanning tree can be trimmed from the score matrix, *e.g.*, the multi-branch trees shown in Figure 1. This dynamic structure represents a "hard" hierarchical layout bias of what objects should gain more contextual information from others, *e.g.*, objects on the person's head are most informative given the question "what on the little girl's head?"; while the whole person's body is more important given the question "Is the girl sitting on the horse correctly?". To avoid the saturation issue caused by the densely connected arbitrary number of children, we further morph the multi-branch trees to the equivalent left-child right-sibling binary trees [14], where the left branches (red) indicate the hierarchical relations and right branches (blue) indicate the parallel relations, then use TreeLSTM [44] to encode the context.

As the above VCTREE construction is in a discrete and non-differentiable nature, we develop a hybrid learning strategy using REINFORCE [20, 41, 48] for tree structure exploration and supervised learning for context encoding and its subsequent tasks. In particular, the evaluation result (Recall for SGG and Accuracy for VQA) from supervised task can be exploited as a critic function that guide the "action" of tree construction. We evaluate VCTREE on two benchmarks: Visual Genome [24] for SGG and VQA2.0 [17] for VQA. For SGG, we achieve a new state-of-the-art on all three standard tasks, *i.e.*, Scene Graph Generation, Scene Graph Classification, and Predicate Classification; for VQA, we achieve competitive results on single model performances. In particular, VCTREE helps high-level vision models fight against the dataset bias. For example, we achieve 4.1% absolute gain in proposed Mean Recall@100 metric of Predicate Classification than MOTIFS [57], and observe higher improvement in VQA2.0 balanced pair subset [45] than normal validation set. Qualitative results also show that VCTREE composes interpretable structures.

## 2. Related Work

**Visual Context Structures**. Despite the consensus on the value of visual contexts, existing context models are diversified into a variety of implicit or explicit approaches. Implicit models directly encode surrounding pixels into multi-scale feature maps, *e.g.*, dilated convolution [56] presents a efficient way to increase receptive field, applicable in various dense prediction tasks [8, 9]; feature pyramid structure [27] combines low-resolution contextual features with high-resolution detailed features, facilitating object detection with rich semantics. Explicit models incorporate contextual cues through object connections. However, such methods [25, 50, 57] group objects into fixed layouts, *i.e.*, chains or graphs.

**Learning to Compose Structures**. Learning to compose structures is becoming popular in NLP for sentence representation, *e.g.*, Cho *et al*. [11] applied a gated recursive convolutional neural network (grConv) to control the bottom-up feature flow for a dynamic structure; Choi *et al*. [12] combines TreeLSTM with Gumbel-Softmax, allowing task-specific tree structures automatically learned from plain text. Yet, only few works compose visual structures for images. Conventional approaches construct a statistical dependency graph/tree for the entire dataset based on object categories [13] or exemplars [32]. Those statistical methods cannot put per-image objects in a context as a whole to reason over content-/task-specific fashion. Socher *et al*. [43] constructed a bottom-up tree structure to parse images; however, their tree structure learning is supervised while ours is reinforced, which does not require tree ground-truth.

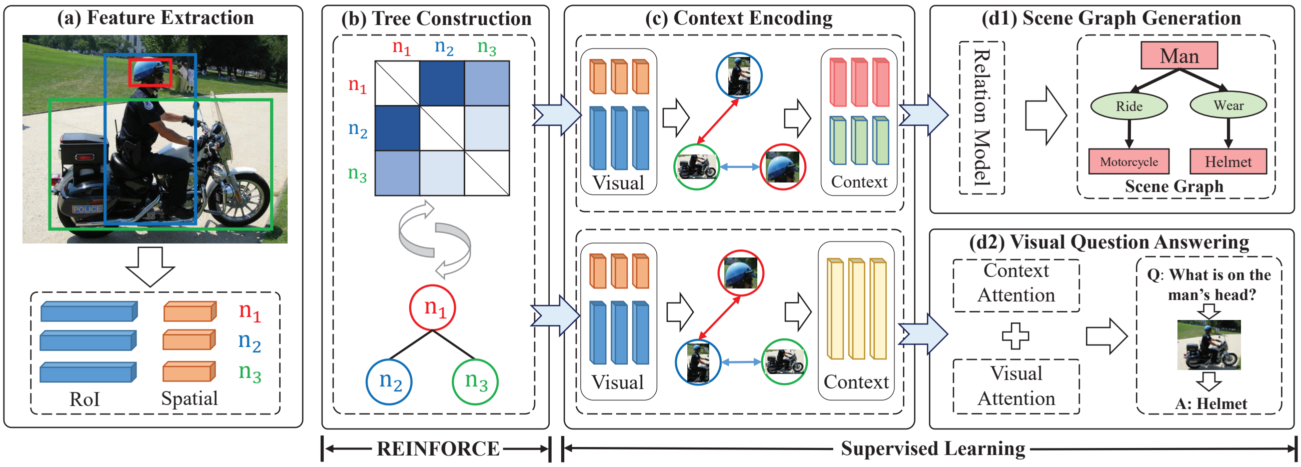

Figure 2. The framework of the proposed VCTREE model. We extract visual features from proposals and construct a dynamic VCTREE using the learnable score matrix. The tree structure is used to encode the object-level visual context, which will be decoded for each specific end-task. Parameters in stages (c)&(d) are trained by supervised learning, while those in stage (b) are using REINFORCE with a self-critic baseline.

**Visual Reasoning Tasks**. Scene Graph Generation (SGG) task [50, 52] is derived from Visual Relationship Detection (VRD) [31, 53]. Early work on VRD [31] treats objects as isolated individuals, while SGG considers each image as a whole. Along with the widely used message passing mechanism [50], a variety of context models [25, 26, 34, 51] have been exploited in SGG to fine-tune local predictions through rich global contexts, making it the best competition field for different contextual models. Visual Question Answering (VQA) as a high-level task bridges the gap between computer vision and natural language processing. State-of-the-art VQA models [1, 3, 45] rely on bag-of-object visual attentions which can be considered as a trivial context structure. However, we propose to learn a tree context structure that is dynamic to visual content and questions.

## 3. Approach

As illustrated in Figure 2, our VCTREE model can be summarized into the following four steps. (a) We adopt Faster-RCNN to detect object proposals [40]. The visual feature of each proposal $i$ is presented as $\boldsymbol{x}_i$, concatenating a RoIAlign feature [18] $\boldsymbol{v}_i \in \mathbb{R}^{2048}$ and spatial feature $\boldsymbol{b}_i \in \mathbb{R}^8$, where 8 elements indicate the bounding box coordinates $(x_1, y_1, x_2, y_2)$, center $(\frac{x_1+x_2}{2}, \frac{y_1+y_2}{2})$, and size $(x_2 - x_1, y_2 - y_1)$, respectively. Note that the visual feature $\boldsymbol{x}_i$ is not limited to bounding box; segment feature from instance segmentations [18] or panoptic segmentations [23] could also be alternatives. (b) In Section 3.1, a learnable matrix will be introduced to construct VCTREE. Moreover, since the VCTREE construction is discrete in nature and the score matrix is non-differentiable from the loss of end-task, we develop a hybrid learning strategy in Section 3.5. (c) In Section 3.2, we employ Bidirectional Tree LSTM (BiTreeLSTM) to encode the contextual cues using the constructed VCTREE. (d) The encoded contexts will be decoded for each specific end-task detailed in Section 3.3 and Section 3.4.

### 3.1. VCTREE Construction

VCTREE construction aims to learn a score matrix $\boldsymbol{S}$, which approximates the task-dependent validity between each object pair. Two principles guide the formulation of this matrix: 1) inherent object correlations should be maintained, *e.g.*, "man wears helmet" in Figure 2; (2) task related object pair has higher score than irrelevant ones, *e.g.*, given question "what is on the man's head?", "man-helmet" pair should be more important than "man-motorcycle" and "helmet-motorcycle" pairs. Therefore, we define each element of $\boldsymbol{S}$ as the product of the object correlation $f(\boldsymbol{x}_i, \boldsymbol{x}_j)$ and the pairwise task-dependency $g(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{q})$:

$$\begin{cases} \boldsymbol{S}_{ij} = f(\boldsymbol{x}_i, \boldsymbol{x}_j) \cdot g(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{q}), \\ f(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma\left(\mathrm{MLP}(\boldsymbol{x}_i, \boldsymbol{x}_j)\right), \\ g(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{q}) = \sigma(h(\boldsymbol{x}_i, \boldsymbol{q})) \cdot \sigma(h(\boldsymbol{x}_j, \boldsymbol{q})), \end{cases} \tag{1}$$

where $\sigma(\cdot)$ is the sigmoid function; $\boldsymbol{q}$ is the task feature, *e.g.*, the question feature encoded by GRU in VQA; MLP is a multi-layer perceptron; $h(\boldsymbol{x}_i, \boldsymbol{q})$ is the object-task correlation in VQA, which will be introduced later in Section 3.4. In SGG, the entire $g(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{q})$ is set to 1, as we assume that each object pair contributes equally without the question prior. We pretrain $f(\boldsymbol{x}_i, \boldsymbol{x}_j)$ on Visual Genome [24] for a reasonable binary prior if two objects are related. Yet, such a pretrained model is not perfect due to the lack of coherent graph-level constraint or question prior, so it will be further fine-tuned in Section 3.5.

Considering $\boldsymbol{S}$ as a symmetric adjacency matrix, we can obtain a maximum spanning tree using the Prim's

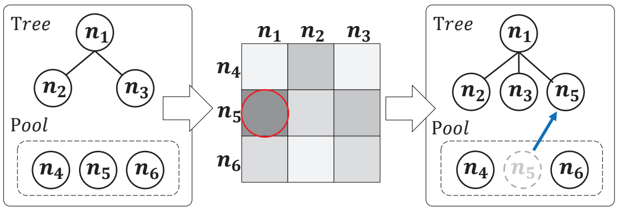

Figure 3. The maximum spanning tree from $\boldsymbol{S}$. In each step, a node in the remaining pool is connected to the current tree, if it has the highest validity score.

algorithm [39], with a root (source node) $i$ satisfying $\arg\max_i \sum_{j \neq i} \boldsymbol{S}_{ij}$. In a nutshell, as illustrated in Figure 3, we construct the tree recursively by connecting the node from the pool to the tree node if it has the most validity. Note that during the tree structure exploration in Section 3.5, each of the $i$-th step $t^{(i)}$ in the above tree construction is sampled from all possible choices in a multinomial distribution with the probability $p(t^{(i)}|t^{(1)}, ..., t^{(i-1)}, \boldsymbol{S})$ in proportion to the validity. The resultant tree is multi-branch and is merely a sparse graph with only one kind of connection, which is still unable to discriminate the hierarchical and parallel relations in the subsequent context encoding. To this end, we convert the multi-branch tree into an equivalent binary tree, *i.e.*, VCTREE by changing non-leftmost edges into right branches as in Figure 1. In this fashion, the right branches (blue) indicate parallel contexts, and left ones (red) indicate hierarchical contexts. Such a binary tree structure achieves significant improvements in our SGG and VQA experiments compared to its multi-branch alternative.

## 3.2. TreeLSTM Context Encoding

Given the above constructed VCTREE, we adopt BiTreeLSTM as our context encoder:

$$D = \text{BiTreeLSTM}(\{\boldsymbol{z}_i\}_{i=1,2,...,n}), \tag{2}$$

where $\boldsymbol{z}_i$ is the input node feature, which will be specified in each task, and $D = [\boldsymbol{d}_1, \boldsymbol{d}_2, ..., \boldsymbol{d}_n]$ is the encoded object-level visual context. Each $\boldsymbol{d}_i = [\vec{\boldsymbol{h}}_i; \overleftarrow{\boldsymbol{h}}_i]$ is the concatenated hidden states from both TreeLSTM [44] directions:

$$\vec{\boldsymbol{h}}_i = \text{TreeLSTM}(\boldsymbol{z}_i, \vec{\boldsymbol{h}}_p), \tag{3}$$

$$\overleftarrow{\boldsymbol{h}}_i = \text{TreeLSTM}(\boldsymbol{z}_i, [\overleftarrow{\boldsymbol{h}}_l; \overleftarrow{\boldsymbol{h}}_r]), \tag{4}$$

where $\rightarrow$and$\leftarrow$ denote the top-down and bottom-up directions, respectively; we slightly abuse the subscripts $p, l, r$ to denote the parent, left child, and right child of node $i$. The order of the concatenation $[\overleftarrow{\boldsymbol{h}}_l; \overleftarrow{\boldsymbol{h}}_r]$ in Eq. (4) indicates the explicit discrimination between the left and right branches in context encoding. We use zero vectors to pad all the missing branches.

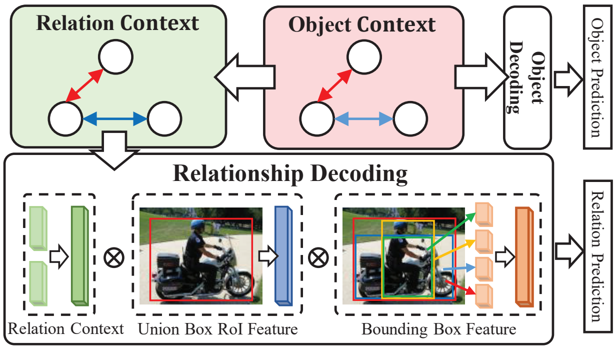

Figure 4. The overview of our SGG Model. The object context feature will be used to decode object categories, and the pairwise relationship decoding jointly fuses the relation context feature, RoIAlign feature of union box, and bounding box feature, before prediction.

## 3.3. Scene Graph Generation Model

Now we detail the implementation of Eq. (2) and how to decode them for the SGG task as illustrated in Figure 4.

**Object Context Encoding**. We employ BiTreeLSTM from Eq. (2) to encode object context representation into $D^o = [\boldsymbol{d}^o_1, \boldsymbol{d}^o_2, ..., \boldsymbol{d}^o_n], \boldsymbol{d}^o_i \in \mathbb{R}^{512}$. We set inputs $\boldsymbol{z}_i$ of Eq. (2) to $[\boldsymbol{x}_i; \boldsymbol{W}_1\hat{\boldsymbol{c}}_i]$, *i.e.*, concatenation of object visual features and embedded N-way original Faster-RCNN class probabilities, where $\boldsymbol{W}_1$ is the embedding matrix that maps each original label distribution $\hat{\boldsymbol{c}}_i$ into $\mathbb{R}^{200}$.

**Relation Context Encoding**. We apply an additional BiTreeLSTM using the above $\boldsymbol{d}^o_i$ as input $\boldsymbol{z}_i$ to further encode the relation context $D^r = [\boldsymbol{d}^r_1, \boldsymbol{d}^r_2, ..., \boldsymbol{d}^r_n], \boldsymbol{d}^r_i \in \mathbb{R}^{512}$.

**Context Decoding**. The goal of SGG is to detect objects and then predict their relationship. Similar to [57], we adopt a dynamic object prediction which can be viewed as a decoding process in a top-down direction using Eq. (3), that is, the object class of a child is dependent on its parent. Specifically, we set the input $\boldsymbol{z}_i$ of Eq. (3) to be $[\boldsymbol{d}^o_i; \boldsymbol{W}_2\boldsymbol{c}_p]$, where $\boldsymbol{c}_p$ is the predicted label distribution of the $i$'s parent, and $\boldsymbol{W}_2$ embeds it into $\mathbb{R}^{200}$, then the output hidden is passed to a softmax classifier to achieve object label distribution $\boldsymbol{c}_i$.

The relationship prediction is in a pairwise fashion. First, we collect three pairwise features for each object pair: (1) $\boldsymbol{d}_{ij} = \text{MLP}([\boldsymbol{d}^r_i; \boldsymbol{d}^r_j])$ as the context feature, (2) $\boldsymbol{b}_{ij} = \text{MLP}([\boldsymbol{b}_i; \boldsymbol{b}_j; \boldsymbol{b}_{i\cup j}; \boldsymbol{b}_{i\cap j}])$ as the bounding box pair feature, with $i \cup j, i \cap j$ being union box and intersection box, (3) $\boldsymbol{v}_{ij}$ as the RoIAlign feature [18] from the union bounding box of the object pair. All $\boldsymbol{d}_{ij}, \boldsymbol{v}_{ij}, \boldsymbol{b}_{ij}$ are under the same dimension $\mathbb{R}^{2048}$. Then, we fuse them into a final pairwise feature: $\boldsymbol{g}_{ij} = \boldsymbol{d}_{ij} \cdot \boldsymbol{v}_{ij} \cdot \boldsymbol{b}_{ij}$, before feed it into the softmax predicate classifier, where $\cdot$ is element-wise product.

## 3.4. Visual Question Answering Model

Now we detail the implementation of Eq. (2) for VQA, and illustrate our VQA model in Figure 5.

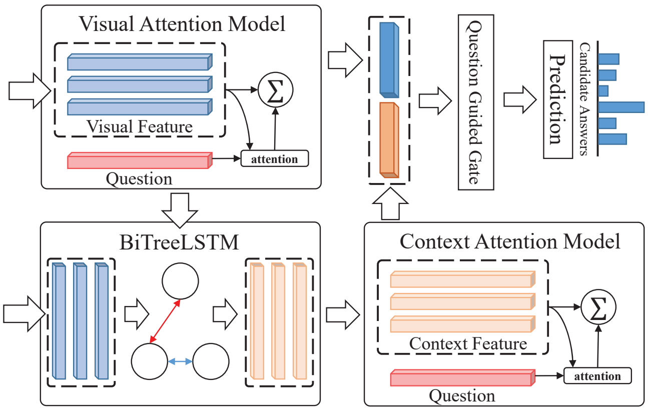

Figure 5. The overview of our VQA framework. It contains two multimodal attention models for visual feature and context feature. Outputs from both models will be concatenated and passed to a question-guided gate before answer prediction.

**Context Encoding**. The context feature in VQA: $D^q = [\boldsymbol{d}^q_1, \boldsymbol{d}^q_2, ..., \boldsymbol{d}^q_n]$, $\boldsymbol{d}^q_i \in \mathbb{R}^{1024}$ is directly encoded from the bounding box visual feature $\boldsymbol{x}_i$ by Eq. (2).

**Multimodal Attention Feature**. We adopt a popular attention model from previous work [1, 45] to calculate the multimodal joint feature $\boldsymbol{m} \in \mathbb{R}^{1024}$ for each question and image pair:

$$\boldsymbol{m} = f_d(\hat{\boldsymbol{z}}, \boldsymbol{q}), \tag{5}$$

where $\boldsymbol{q} \in \mathbb{R}^{1024}$ is the question feature from a one-layer GRU encoding the sentence; $\hat{\boldsymbol{z}} = \sum_{i=1}^{N} \alpha_i \boldsymbol{z}_i$ is the attentive image feature calculated from the input feature set $\{\boldsymbol{z}_i\}$, $\alpha_i = \exp(u_i)/\sum_k \exp(u_k)$ is the attention weight from object-task correlation $u_i = h(\boldsymbol{z}_i, \boldsymbol{q}) = \text{MLP}\big(f_d(\boldsymbol{z}_i, \boldsymbol{q})\big)$, with the output of MLP being a scalar; $f_d$ can be any multi-modal feature fusion function, in particular, we adopt $f_d(\boldsymbol{x}, \boldsymbol{y}) = \text{ReLU}(\boldsymbol{W}_3\boldsymbol{x} + \boldsymbol{W}_4\boldsymbol{y}) - (\boldsymbol{W}_3\boldsymbol{x} - \boldsymbol{W}_4\boldsymbol{y})^2$ as in [59], with $\boldsymbol{W}_3$ and $\boldsymbol{W}_4$ projecting $\boldsymbol{x}, \boldsymbol{y}$ into the same dimension. Therefore, we can use Eq. (5) to obtain both the multimodal visual attention feature $\boldsymbol{m}_x$ by setting input $\boldsymbol{z}_i$ to $\boldsymbol{x}_i$ and multimodal contextual attention feature $\boldsymbol{m}_d$ by setting $\boldsymbol{z}_i$ to $\boldsymbol{d}^q_i$.

**Question Guided Gate Decoding**. However, the importance of $\boldsymbol{m}_x$ and $\boldsymbol{m}_d$ varies from question to question, *e.g.*, "is there a dog?" only requires visual features for detection, while "is the man dressed formally?" is highly context dependent. Inspired by [42], we adopt a question guided gate to select the most related channels from $[\boldsymbol{m}_x; \boldsymbol{m}_d]$. The gate vector $\boldsymbol{g} \in \mathbb{R}^{2048}$ is defined as:

$$\boldsymbol{g} = \sigma\big(\text{MLP}([\boldsymbol{q}; \boldsymbol{W}_5\boldsymbol{l}_q])\big), \tag{6}$$

where $\boldsymbol{l}_q \in \mathbb{R}^{65}$ is a one-hot question type vector defined by prefixed words of questions, which is embedded into $\mathbb{R}^{256}$ by matrix $\boldsymbol{W}_5$, and $\sigma(\cdot)$ denotes the sigmoid function.

Finally, we fuse $\boldsymbol{g} \cdot [\boldsymbol{m}_x; \boldsymbol{m}_d]$ as the final VQA feature and feed it into the softmax classifier.

### 3.5. Hybrid Learning

Due to the discrete nature of VCTREE construction, the score matrix $\boldsymbol{S}$ is not fully differentiable from the loss back-propagated from the end-task loss. Inspired by [20], we use a hybrid learning strategy that combines reinforcement learning, *i.e.*, policy gradient [48] for the parameters $\theta$ of $\boldsymbol{S}$ in the tree construction and supervised learning for the rest parameters. Suppose a layout $l$, *i.e.*, a constructed VCTREE, is sampled from $\pi(l|I, q; \theta)$, *i.e.*, the construction procedure in Section 3.1, where $I$ is the given image, $q$ is the task, *e.g.*, questions in VQA. To avoid clutter, we drop $I$ and $q$. Then, we define the reinforcement learning loss $L_r(\theta)$ as:

$$L_r(\theta) = -E_{l\sim\pi(l|\theta)}[r(l)], \tag{7}$$

where $L_r(\theta)$ aims to minimize the negative expected reward $r(l)$, which can be the end-task evaluation metrics such as Recall@100 for SGG and Accuracy for VQA. Then, the above gradient will be $\nabla_\theta L_r(\theta) = -E_{l\sim\pi(l|\theta)}[r(l)\nabla_\theta log\pi(l|\theta)]$. Since it is impractical to estimate all possible layouts, we use the Monte-Carlo sampling to estimate the gradient:

$$\nabla_\theta L_r(\theta) \approx -\frac{1}{M}\sum_{m=1}^{M}\Big(r(l_m)\nabla_\theta log\pi(l_m|\theta)\Big), \tag{8}$$

where we set M to 1 in our implementation.

To reduce the gradient variance, we apply a self-critic baseline [41] $b = r(\hat{l})$, where $\hat{l}$ is the greedy constructed tree without sampling. So the original reward $r(l_m)$ can be replaced by $r(l_m) - b$ in Eq. (8). We observe faster convergence than using a traditional moving baseline [33].

The overall hybrid learning will be alternatively conducted between supervised learning and reinforcement learning, where we first train the supervised end-task on pretrained $\pi(l|\theta)$, then fix the end-task as reward function to learn our reinforcement policy network, after that, we update the supervised end-task by new $\pi(l|\theta)$. The latter two stages are running alternatively 2 times in our model.

## 4. Experiments on Scene Graph Generation

### 4.1. Settings

**Dataset.** Visual Genome (VG) [24] is a popular benchmark for SGG. It contains 108,077 images with tens of thousands of unique object and predicate relation categories, yet most of categories have very limited instances. Therefore, previous works [26, 50, 58] proposed various VG splits that remove rare categories. We adopted the most popular one from [50], which selects top-150 object categories and top-50 predicate categories by frequency. The entire dataset is divided into the training set and test set by 70%, 30%, respectively. We further picked 5,000 images from training set as the validation set for hyper-parameter tuning.

|  | Scene Graph Generation |  |  | Scene Graph Classification |  |  | Predicate Classification |  |  |
|---|---|---|---|---|---|---|---|---|---|
| Model | R@20 | R@50 | R@100 | R@20 | R@50 | R@100 | R@20 | R@50 | R@100 |
| VRD [31] | - | 0.3 | 0.5 | - | 11.8 | 14.1 | - | 27.9 | 35.0 |
| AsscEmbed [34] | 6.5 | 8.1 | 8.2 | 18.2 | 21.8 | 22.6 | 47.9 | 54.1 | 55.4 |
| IMP$^\diamond$ [50] | 14.6 | 20.7 | 24.5 | 31.7 | 34.6 | 35.4 | 52.7 | 59.3 | 61.3 |
| TFR [21] | 3.4 | 4.8 | 6.0 | 19.6 | 24.3 | 26.6 | 40.1 | 51.9 | 58.3 |
| FREQ$^\diamond$ [57] | 20.1 | 26.2 | 30.1 | 29.3 | 32.3 | 32.9 | 53.6 | 60.6 | 62.2 |
| MOTIFS$^\diamond$ [57] | 21.4 | 27.2 | 30.3 | 32.9 | 35.8 | 36.5 | 58.5 | 65.2 | 67.1 |
| Graph-RCNN [51] | - | 11.4 | 13.7 | - | 29.6 | 31.6 | - | 54.2 | 59.1 |
| Chain | 21.2 | 27.1 | 30.3 | 33.3 | 36.1 | 36.8 | 59.4 | 66.0 | 67.7 |
| Overlap | 21.4 | 27.3 | 30.4 | 33.7 | 36.5 | 37.1 | 59.5 | 66.0 | 67.8 |
| Multi-Branch | 21.5 | 27.3 | 30.6 | 34.3 | 37.1 | 37.8 | 59.5 | 66.1 | 67.8 |
| VCTREE-SL | 21.7 | 27.7 | 31.1 | 35.0 | 37.9 | 38.6 | 59.8 | 66.2 | 67.9 |
| VCTREE-HL | **22.0** | **27.9** | **31.3** | **35.2** | **38.1** | **38.8** | **60.1** | **66.4** | **68.1** |

Table 1. SGG performances (%) of various methods. $^\diamond$ denotes the methods using the same Faster-RCNN detector as ours. IMP$^\diamond$ is reported from the re-implemented version [57].

|  | SGGen | SGCls | PredCls |
|---|---|---|---|
| Model | mR@100 | mR@100 | mR@100 |
| MOTIFS$^\diamond$ [57] | 6.6 | 8.2 | 15.3 |
| FREQ$^\diamond$ [57] | 7.1 | 8.5 | 16.0 |
| VCTREE-HL | **8.0** | **10.8** | **19.4** |

Table 2. Mean recall (%) of various methods across all the 50 predicate categories.

**Protocols.** We followed three conventional protocols to evaluate our SGG model: (1) **Scene Graph Generation (SGGen)**: given an image, detect object bounding boxes and their categories, and predict their relationships; (2) **Scene Graph Classification (SGCls)**: given ground-truth object bounding boxes in an image, predict the object categories and their relationships; (3) **Predicate Classification (PredCls)**: given the object categories and their bounding boxes in the image, predict their relationships.

**Metrics.** Since the annotation in VG is incomplete and biased, we followed the conventional Recall@K (R@K = 20,50,100) as the evaluation metrics [31, 50, 57]. However, it is well-known that SGG models trained on biased datasets such as VG have low performances for less frequent categories. To this end, we introduced a balanced metric called: **Mean Recall (mR@K)**. It calculates the recall on each predicate category independently, and then averages the results. So, each category contributes equally. Such a metric reduces the influence of some common yet meaningless predicates, *e.g*., "on", "of", and gives equal attention to those infrequent predicates, *e.g*., "riding", "carrying", which are more valuable to high-level reasoning.

## 4.2. Implementation Details

We adopted Faster-RCNN [40] with VGG backbone to detect object bounding boxes and extract RoI features. Since the performance of SGG highly depends on the underlying detector, we used the same set of parameters as [57] for fair comparison. Object correlations $f(\boldsymbol{x}_i, \boldsymbol{x}_j)$ in Eq. (1) will be pretrained on ground-truth bounding boxes with class-agnostic relationships (*i.e*., foreground/background relationships), using all possible symmetric pairs without sampling. In SGGen, top-64 object proposals were selected after non-maximal suppression (NMS) with 0.3 IoU. We set background/foreground ratio for predicate classification to 3, and capped the number of training samples at 64 (retained all foreground pairs if possible). Our model is optimized by SGD with momentum, using learning rate $lr = 6 \cdot 10^{-3}$ and batch size $b = 5$ for supervised learning, and $lr = 6 \cdot 10^{-4}, b = 1$ for reinforcement learning.

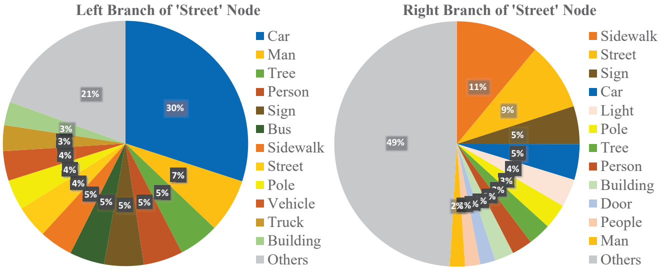

Figure 6. The statistics of left-branch (hierarchical) nodes and right-branch (parallel) nodes of the "street" category.

## 4.3. Ablation Studies

We investigated the influence of different structure construction policies. They are reported on the bottom half of Table 1. The ablative methods are (1) **Chain**: sorting all the objects by $\sum_{j:j\neq i} \boldsymbol{S}_{ij}$, then constructing a chain, which is different from the left-to-right ordered chain in MOTIFS [57]; (2) **Overlap**: iteratively constructing a binary tree by selecting the node with largest number of overlapped objects as parent, and dividing the rest nodes into left/right sub-trees by relatively positions of their bounding boxes; (3) **Multi-Branch**: the maximum spanning tree generated from score matrix $\boldsymbol{S}$, using Child-Sum TreeLSTM [44] to incorporate context; (4) **VCTREE-SL**: the proposed VCTREE trained by supervised learning; (5) **VCTREE-HL**: the complete version of VCTREE, trained by hybrid learning for structure exploration in Section 3.5. As we will show that Multi-Branch is significantly worse than

| VQA2.0 Validation Accuracy | | | | | |
|---|---|---|---|---|---|
| Model | Yes/No | Number | Other | All | Balanced Pairs |
| Graph | 81.8 | 44.9 | 56.6 | 64.5 | 36.3 |
| Chain | 81.8 | 44.5 | 56.9 | 64.6 | 36.3 |
| Overlap | 81.8 | 44.8 | 57.0 | 64.7 | 36.4 |
| Multi-Branch | 82.1 | 44.3 | 56.9 | 64.7 | 36.6 |
| VCTree-SL | 82.3 | 45.0 | 57.0 | 64.9 | 36.9 |
| VCTree-HL | **82.6** | **45.1** | **57.1** | **65.1** | **37.2** |

Table 3. Accuracies (%) of various context structures on the VQA2.0 validation set.

VCTree, so there is no need to conduct hybrid learning experiment on Multi-Branch. We observe that VCTree performs better than other structures, and it is further improved by hybrid learning for structure exploration.

### 4.4. Comparisons with State-of-the-Arts

**Comparing Methods.** We compared VCTree with state-of-the-art methods in Table 1: (1) **VRD** [31], **FREQ** [57] are methods without using visual contexts. (2) **AssocEmbed** [34] assembles implicit contextual features by stacked hourglass backbone [35]. (3) **IMP** [50], **TFR** [21], **MOTIFS** [57], **Graph-RCNN** [51] are explicit context models with a variety of structures.

**Quantitative Analysis.** From Table 1, compared with the previous state-of-the-art MOTIFS [57], the proposed VCTree has the best performances. Interestingly, Overlap tree and Multi-Branch tree are better than other non-tree context models. From Table 2, the proposed VCTree-HL shows larger absolute gains of PredCls under mR@100, which indicates that our model learns non-trivial visual context, *i.e.*, not merely class distribution bias as in FREQ and partially in MOTIFS. Note that MOTIFS [57] is even worse than its FREQ [57] baseline under mR@100.

**Qualitative Analysis.** To better understand what context is learned by VCTree, we visualized a statistics of left-/right-branch nodes for nodes classified as “street” in Figure 6. From the left pie, the hierarchical relations, we can see the node categories are long-tailed, *i.e.*, top-10 categories cover the 73% of the instances; while the right pie, the parallel relations, are more uniformly distributed. This demonstrates that VCTree captures the two types of context successfully. More qualitative examples of VCTrees and their generated scene graph can be viewed in Figure 7. The common errors are generally synonymous labels, *e.g.*, “jeans” vs. “pants”, “man” vs. “person”, and over-interpretation, *e.g.*, the “tail” of bottom left “dog” is considered as “leg”, as it appears at the place where “leg” should be.

## 5. Experiments on Visual Q&A

### 5.1. Settings

**Datasets.** We evaluated the proposed VQA model on VQA2.0 [17]. Compared with VQA1.0 [2], VQA2.0 has more question-image pairs for training (443,757) and validation (214,354), and all the question-answer pairs are balanced by making sure the same question can have different answers. In VQA2.0, the ground-truth accuracy of a candidate answer is considered as the average of $\min(\frac{\#\text{Humans votes}}{3}, 1)$ over all 10 select 9 sets. Question-answer pairs are organized in three answer types: *i.e.* “Yes/No”, “Number”, “Other”. There are also 65 question types determined by prefixed words, which we used to generate question-guided gates. We also tested our models on a balanced subset of validation set, called Balanced Pairs [45], which requires the same question on different images with two different yet perfect (with 1.0 ground-truth score) answers. Since Balanced Pairs strictly removes question-related bias, it reflects the ability of a context model to distinguish subtle differences between images.

| VQA2.0 test-dev | | | | |
|---|---|---|---|---|
| Model | Yes/No | Number | Other | All |
| Teney [45] | 81.82 | 44.21 | 56.05 | 65.32 |
| MUTAN [5] | 82.88 | 44.54 | 56.50 | 66.01 |
| MLB [22] | 83.58 | 44.92 | 56.34 | 66.27 |
| DA-NTN [3] | **84.29** | 47.14 | 57.92 | 67.56 |
| Count [59] | 83.14 | **51.62** | 58.97 | 68.09 |
| Chain | 82.74 | 47.31 | 58.93 | 67.42 |
| Graph | 83.53 | 47.09 | 58.6 | 67.56 |
| VCTree-HL | 84.28 | 47.78 | **59.11** | **68.19** |

Table 4. Single-model accuracies (%) on VQA2.0 test-dev, where MUTAN and MLB are re-implemented versions from [3].

| VQA2.0 test-standard | | | | |
|---|---|---|---|---|
| Model | Yes/No | Number | Other | All |
| Teney [45] | 82.20 | 43.90 | 56.26 | 65.67 |
| MUTAN [5] | 83.06 | 44.28 | 56.91 | 66.38 |
| MLB [22] | 83.96 | 44.77 | 56.52 | 66.62 |
| DA-NTN [3] | **84.60** | 47.13 | 58.20 | 67.94 |
| Count [59] | 83.56 | **51.39** | 59.11 | 68.41 |
| Chain | 83.06 | 47.38 | 58.95 | 67.68 |
| Graph | 84.03 | 47.08 | 58.82 | 68.0 |
| VCTree-HL | 84.55 | 47.36 | **59.34** | **68.49** |

Table 5. Single-model accuracies (%) on VQA2.0 test-standard, where MUTAN and MLB are re-implemented versions from [3].

### 5.2. Implementation Details

We employed a simple text preprocessing for questions and answers, which changes all characters into lower-case and removes special characters. Questions were encoded into a vocabulary of the size 13,758 without trimming. Answers used a 3,000 vocabulary selected by frequency. For fair comparison, we used the same bottom-up feature [1] as previous methods [1, 3, 45, 59], which contains 10 to 100 object proposals per image extracted by Faster-RCNN [40]. We used the same Faster-RCNN detector to pretrain the $f(\boldsymbol{x}_i, \boldsymbol{x}_j)$. Since candidate answers were represented by probabilities rather than one-hot vectors in VQA2.0, we allowed the cross-entropy loss calculating soft categories, *i.e.*, probabilities of ground-truth candidate answers. We used Adam optimizer with learning rate $lr = 0.0015$ and batch size $b = 256$, $lr$ decayed at ratio of 0.5 every 20 epochs.

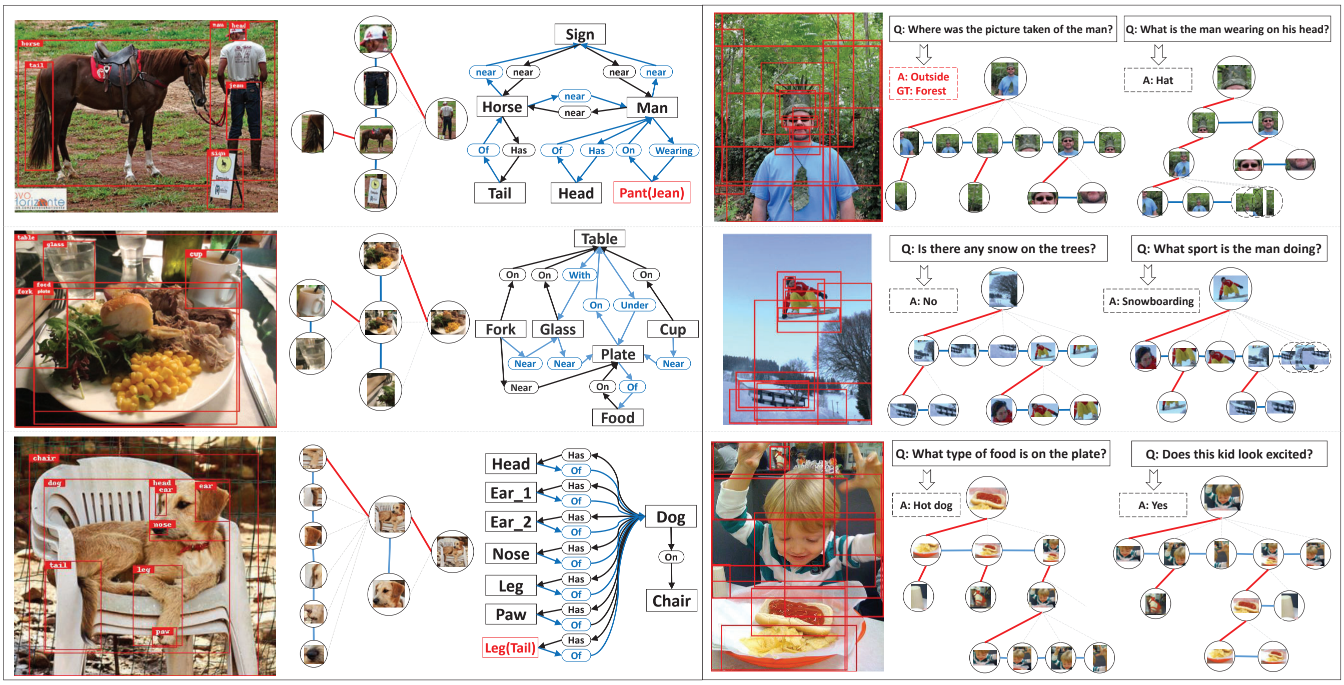

Figure 7. **Left:** the learned tree structure and generated scene graphs in VG. Black color indicates correctly detected objects or predicates; red indicates the misclassified ones; blue indicates correct predictions that not labeled as ground-truth. **Right:** interpretable and dynamic trees subject to different questions in VQA2.0.

### 5.3. Ablation Studies

In addition to the 5 structure construction policies introduced in Section 4.3, we also implemented a fully-connected graph structure using the message passing mechanism [50]. From Table 3, the proposed VCTREE-HL outperforms all the context models on three answer types.

We further evaluated the above context models on VQA2.0 balanced pair subset [45]: the last column of Table 3, and found that the absolute gains between VCTREE-HL and other structures are even larger than those on the original validation set. Meanwhile, as reported in [45], different architectures or hyper-parameters in non-contextual VQA model normally gain less improvements on the balanced pair subset than overall validation set. Thus, it suggests that VCTREE indeed use better context structures to alleviate the question-answer bias in VQA.

### 5.4. Comparisons with State-of-the-Arts

**Comparing Methods.** Table 4 & 5 reports the single-model performances of various state-of-the-art methods [3, 5, 22, 45, 59] on both test-dev and test-standard sets. For fair comparison, the reported methods are all using the same Faster-RCNN features [1] as ours.
**Quantitative Analysis.** The proposed VCTREE-HL shows the best overall performance in both test-dev and test-standard. Note that though Count [59] has close overall performance to our VCTREE, it mainly improves the "Number" task by the elaborately designed model, while the proposed VCTREE is a more general solution.
**Qualitative Analysis.** We visualized several examples of VCTREE-HL on the validation set. They illustrate that the proposed VCTREE is able to learn dynamic structures with interpretability, *e.g.*, in Figure 7, given the right middle image with the question "Is there any snow on the trees?", the generated VCTREE locates the "tree" then searching for the "snow", while with question "What sport is the man doing?", the "man" appears to be the root.

## 6. Conclusions

In this paper, we proposed a dynamic tree structure called VCTREE to capture task-specific visual contexts, which can be encoded to support two high-level vision tasks: SGG and VQA. By exploiting VCTREE, we observed consistent performance gains in SGG on Visual Genome and in VQA on VQA2.0, compared to models with or without visual contexts. Besides, to justify that VCTREE learns non-trivial contexts, we conducted additional experiments against the category bias in SGG and the question-answer bias in VQA, respectively. In the future, we intend to study the potential of a dynamic forest as the underlying context structure.

## References

[1] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *CVPR*, 2018. 3, 5, 7, 8

[2] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *ICCV*, 2015. 7

[3] Yalong Bai, Jianlong Fu, Tiejun Zhao, and Tao Mei. Deep attention neural tensor network for visual question answering. In *ECCV*, 2018. 3, 7, 8

[4] Moshe Bar. Visual objects in context. *Nature Reviews Neuroscience*, 2004. 1

[5] Hedi Ben-Younes, Rémi Cadene, Matthieu Cord, and Nicolas Thome. Mutan: Multimodal tucker fusion for visual question answering. In *ICCV*, 2017. 7, 8

[6] Irving Biederman, Robert J Mezzanotte, and Jan C Rabinowitz. Scene perception: Detecting and judging objects undergoing relational violations. *Cognitive Psychology*, 1982. 1

[7] Long Chen, Hanwang Zhang, Jun Xiao, Xiangnan He, Shiliang Pu, and Shih-Fu Chang. Scene dynamics: Counterfactual critic multi-agent training for scene graph generation. *arXiv preprint arXiv:1812.02347*, 2018. 2

[8] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *TPAMI*, 2018. 1, 2

[9] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017. 2

[10] Xinlei Chen, Li-Jia Li, Li Fei-Fei, and Abhinav Gupta. Iterative visual reasoning beyond convolutions. In *CVPR*, 2018. 2

[11] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. In *SSST-8*, 2014. 2

[12] Jihun Choi, Kang Min Yoo, and Sang-goo Lee. Learning to compose task-specific tree structures. In *AAAI*, 2018. 2

[13] Myung Jin Choi, Antonio Torralba, and Alan S Willsky. A tree-based context model for object recognition. *TPAMI*, 2012. 2

[14] Thomas H. Cormen, Clifford Stein, Ronald L. Rivest, and Charles E. Leiserson. *Introduction to Algorithms*. McGraw-Hill Higher Education, 2001. 1, 2

[15] Bo Dai, Yuqi Zhang, and Dahua Lin. Detecting visual relationships with deep relational networks. In *CVPR*, 2017. 2

[16] Santosh K Divvala, Derek Hoiem, James H Hays, Alexei A Efros, and Martial Hebert. An empirical study of context in object detection. In *CVPR*, 2009. 1

[17] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *CVPR*, 2017. 2, 7

[18] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017. 1, 3, 4

[19] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 1997. 2

[20] Ronghang Hu, Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Kate Saenko. Learning to reason: End-to-end module networks for visual question answering. In *ICCV*, 2017. 2, 5

[21] Seong Jae Hwang, Sathya N Ravi, Zirui Tao, Hyunwoo J Kim, Maxwell D Collins, and Vikas Singh. Tensorize, factorize and regularize: Robust visual relationship learning. In *CVPR*, 2018. 6, 7

[22] Jin-Hwa Kim, Kyoung-Woon On, Woosang Lim, Jeonghee Kim, Jung-Woo Ha, and Byoung-Tak Zhang. Hadamard product for low-rank bilinear pooling. In *ICLR*, 2016. 7, 8

[23] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. Panoptic segmentation. *arXiv preprint arXiv:1801.00868*, 2018. 3

[24] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *IJCV*, 2017. 2, 3, 5

[25] Yikang Li, Wanli Ouyang, Bolei Zhou, Jianping Shi, Chao Zhang, and Xiaogang Wang. Factorizable net: An efficient subgraph-based framework for scene graph generation. In *ECCV*, 2018. 2, 3

[26] Yikang Li, Wanli Ouyang, Bolei Zhou, Kun Wang, and Xiaogang Wang. Scene graph generation from objects, phrases and caption regions. In *ICCV*, 2017. 2, 3, 5

[27] Tsung-Yi Lin, Piotr Dollár, Ross B Girshick, Kaiming He, Bharath Hariharan, and Serge J Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017. 1, 2

[28] Daqing Liu, Zheng-Jun Zha, Hanwang Zhang, Yongdong Zhang, and Feng Wu. Context-aware visual policy network for sequence-level image captioning. *ACM on Multimedia Conference*, 2018. 2

[29] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *ECCV*, 2016. 1

[30] Yong Liu, Ruiping Wang, Shiguang Shan, and Xilin Chen. Structure inference net: Object detection using scene-level context and instance-level relationships. In *CVPR*, 2018. 2

[31] Cewu Lu, Ranjay Krishna, Michael Bernstein, and Li Fei-Fei. Visual relationship detection with language priors. In *ECCV*, 2016. 3, 6, 7

[32] Tomasz Malisiewicz and Alyosha Efros. Beyond categories: The visual memex model for reasoning about object relationships. In *NIPS*, 2009. 2

[33] Volodymyr Mnih, Nicolas Heess, Alex Graves, et al. Recurrent models of visual attention. In *NIPS*, 2014. 5

[34] Alejandro Newell and Jia Deng. Pixels to graphs by associative embedding. In *NIPS*, 2017. 3, 6, 7

[35] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *ECCV*, 2016. 7

[36] Yulei Niu, Hanwang Zhang, Manli Zhang, Jianhong Zhang, Zhiwu Lu, and Ji-Rong Wen. Recursive visual attention in visual dialog. *CVPR*, 2019. 2

[37] Aude Oliva and Antonio Torralba. The role of context in object recognition. *Trends in Cognitive Sciences*, 2007. 1

[38] Pedro O Pinheiro, Tsung-Yi Lin, Ronan Collobert, and Piotr Dollár. Learning to refine object segments. In *ECCV*, 2016. 1

[39] Robert Clay Prim. Shortest connection networks and some generalizations. *Bell System Technical Journal*, 1957. 4

[40] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, 2015. 1, 2, 3, 6, 7

[41] Steven J. Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. Self-critical sequence training for image captioning. In *CVPR*, 2017. 2, 5

[42] Yang Shi, Tommaso Furlanello, Sheng Zha, and Animashree Anandkumar. Question type guided attention in visual question answering. In *ECCV*, 2018. 5

[43] Richard Socher, Cliff C Lin, Chris Manning, and Andrew Y Ng. Parsing natural scenes and natural language with recursive neural networks. In *ICML*, 2011. 2

[44] Kai Sheng Tai, Richard Socher, and Christopher D Manning. Improved semantic representations from tree-structured long short-term memory networks. In *ACL*, 2015. 1, 2, 4, 6

[45] Damien Teney, Peter Anderson, Xiaodong He, and Anton van den Hengel. Tips and tricks for visual question answering: Learnings from the 2017 challenge. In *CVPR*, 2018. 2, 3, 5, 7, 8

[46] Damien Teney, Lingqiao Liu, and Anton van den Hengel. Graph-structured representations for visual question answering. In *CVPR*, 2017. 2

[47] Takeo Watanabe, Alexander M Harner, Satoru Miyauchi, Yuka Sasaki, Matthew Nielsen, Daniel Palomo, and Ikuko Mukai. Task-dependent influences of attention on the activation of human primary visual cortex. *Proceedings of the National Academy of Sciences*, 1998. 2

[48] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 1992. 2, 5

[49] Yingjie Xia, Luming Zhang, Zhenguang Liu, Liqiang Nie, and Xuelong Li. Weakly supervised multimodal kernel for categorizing aerial photographs. *IEEE Transactions on Image Processing*, 2017. 2

[50] Danfei Xu, Yuke Zhu, Christopher B Choy, and Li Fei-Fei. Scene graph generation by iterative message passing. In *CVPR*, 2017. 1, 2, 3, 5, 6, 7, 8

[51] Jianwei Yang, Jiasen Lu, Stefan Lee, Dhruv Batra, and Devi Parikh. Graph r-cnn for scene graph generation. In *ECCV*, 2018. 3, 6, 7

[52] Xu Yang, Kaihua Tang, Hanwang Zhang, and Jianfei Cai. Auto-encoding graphical inductive bias for descriptive image captioning. *arXiv preprint arXiv:1812.02378*, 2018. 3

[53] Xu Yang, Hanwang Zhang, and Jianfei Cai. Shuffle-then-assemble: learning object-agnostic visual relationship features. In *ECCV*, 2018. 3

[54] Ting Yao, Yingwei Pan, Yehao Li, and Tao Mei. Exploring visual relationship for image captioning. In *ECCV*, 2018. 2

[55] Guojun Yin, Lu Sheng, Bin Liu, Nenghai Yu, Xiaogang Wang, Jing Shao, and Chen Change Loy. Zoom-net: Mining deep feature interactions for visual relationship recognition. In *ECCV*, 2018. 2

[56] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2016. 2

[57] Rowan Zellers, Mark Yatskar, Sam Thomson, and Yejin Choi. Neural motifs: Scene graph parsing with global context. In *CVPR*, 2018. 1, 2, 4, 6, 7

[58] Ji Zhang, Mohamed Elhoseiny, Scott Cohen, Walter Chang, and Ahmed M Elgammal. Relationship proposal networks. In *CVPR*, 2017. 5

[59] Yan Zhang, Jonathon Hare, and Adam Prügel-Bennett. Learning to count objects in natural images for visual question answering. In *ICLR*, 2018. 5, 7, 8

[60] Rui Zhao, Wanli Ouyang, Hongsheng Li, and Xiaogang Wang. Saliency detection by multi-context deep learning. In *CVPR*, 2015. 1

[61] 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*, 2015. 2