Visual Relationship Detection Using Joint Visual-Semantic Embedding

Binglin Li
School of Engineering Science
Simon Fraser University
Burnaby, BC, Canada, V5A 1S6
Email: binglinl@sfu.ca

Yang Wang
Department of Computer Science
University of Manitoba
Winnipeg, MB, Canada, R3T 2N2
Email: ywang@cs.umanitoba.ca

***Abstract*—Visual relationship detection can serve as the intermediate building block for higher level tasks such as image captioning, visual question answering, image-text matching. Due to the long tail of relationship distribution in real world images, zero-shot predication of relationships that it has never seen before can alleviate stress of collecting every possible relationship. Following zero-shot learning (ZSL) strategies, we propose a joint visual-semantic embedding model for visual relationship detection. In our model, the visual vector and semantic vector are projected to a shared latent space to learn the similarity between the two branches. In the semantic embedding, sequential features in terms of <*sub*, *pred*, *obj*> are learned to provide the context information and then concatenated with corresponding component vector of the relationship triplet. Experiments show that the proposed model achieves superior performance in zero-shot visual relationship detection and comparable results in non-zero-shot scenario.**

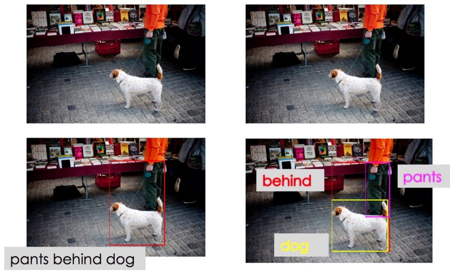


Fig. 1. Difference between visual phrase detection and visual relationship detection. (Left) In phrase detection, we only need to localize one bounding box for the entire phrase. (Right) In visual relationship detection, we need to localize the bounding boxes for all participating objects and the corresponding visual relationship.

I. Introduction

We consider the problem of visual relationship detection. A visual relationship is represented as a triplet <*sub*, *pred*, *obj*>. It involves two participating objects (sub and obj in the triplet). The predicate in a visual relationship can be a verb (e.g. *ride*), or preposition (e.g. *by*), spatial phrase (e.g. *in the front of*), or comparative phrase (e.g. *taller than*). The goal of visual relationship detection is to localize the two participating objects and their mutual relationship with bounding boxes. See Fig. 1 (2nd column) for an illustration of the visual relationship detection: the bounding boxes for pairs of objects ("pants" and "dog") and the corresponding visual relationship ("behind") are localized with separate bounding boxes. For a given image, the output will detect all the interacted objects pairs and their mutual relationships. In this work, we are particularly interested in methods that can perform *zero-shot visual relationship detection*. In this setting, we assume that the triplet <*sub*, *pred*, *obj*> never appears in training data, although each component (sub, $pred$, or obj) in the triplet has appeared during training.

The relationship triplet <*sub*, *pred*, *obj*> can serve as the intermediate building block for higher level tasks such as image captioning [1], visual question answering [2] and image-text matching [3]. It helps to better understand how the entities interact with each other at their current pixel locations in the images. Visual relationship detection is related to several standard visual recognition tasks, such as object detection, phrase detection. But there are some important differences as well. Unlike object detection where the visual appearance of an object is the most important cue for detection, relationship detection requires reasoning about the relative spatial relationship of objects. The relative spatial information also provides important cues for predicting the predicate in the visual relationship. Unlike phrase detection where the relationship is detected with one bounding box, relationship detection requires separate bounding box for each component in the triplet <*sub*, *pred*, *obj*>, as showed in Fig. 1. This will give more detailed information concerning how the subject interacts with the object. Since we can use off-the-shelf object detectors to detect sub and obj in a relationship, the key challenge of visual relationship detection is predicting the predicate given the candidate sub and obj bounding boxes.

Most previous work treats predicate prediction as a classification problem where a classifier is learned for each possible predicate. However, the classification-based approach usually does not consider the phrase context information when predicting the predicate. For example, for the relationship "person ride bike", most previous work simply learns a predicate classifier for "ride". But this approach ignores the fact that *person* is the subject and *bike* is the object in this relationship. This kind of sequential information has been well studied with LSTM [4] in natural language processing (NLP) and some computer vision tasks such as image captioning [1] and visual question answering [2]. When dealing with a text sequence, each word in the sequence corresponds to one unique word in the vocabulary and we assign a vector to each word in the vocabulary to represent its meaning. LSTM can learn the hidden relations among the word vectors during training and map the word meanings to the relationship space.

Instead of considering predicate prediction as a classification problem, we propose a joint visual-semantic embedding approach for predicate prediction (see Fig. 2). Our model consists of a semantic embedding branch and a visual embedding branch. The goal of the semantic embedding branch is to embed a visual relationship triplet <*sub*, *pred*, *obj*> as a vector. The goal of the visual embedding branch is to represent the appearance and spatial features from subject, object and predicate bounding boxes as a vector. Finally, we project the semantic and visual vectors from these two branches in a shared latent space. The two vectors will be projected close to each other if the relationship triplet <*sub*, *pred*, *obj*> is a good match to the visual information from the bounding boxes. The advantage of this embedding approach is that we can easily handle zero-shot visual relationship detection.

II. Related Works

In this section, we review prior work in several lines of research relevant to our work.

A. Object Detection

There has been significant advances in object detection in the past few years. Some object detection systems (e.g. Fast/Faster-RCNN [5], [6]) generate object proposals in image and classify each proposal using convolutional neural networks (CNN). Recent work such as SSD [7] and YOLO [8] proposes more efficient methods that can detect objects in an image in one shot without generating object proposals.

B. Visual Relationship Detection

Recent visual relationship detection work follows two pipelines. Most of them train object and predicate detectors separately. Lu *et al.* [9] applies R-CNN for object detection and leverages language prior module that considers similarity between relationships and relative rank of frequent occurrence, along with the visual appearance features to predict different types of relationships. Dai *et al.* [10] integrates appearance and spatial features, and proposes a DR-Net to capture the statistical relations among the triplet components. Zhang *et al.* [11] extracts three types of object features and models the relationships as a vector translation into the relation space. Zhang *et al.* [12] proposes a context-aware model that can augment with an attention mechanism to improve the performance.

Others train object and relationship detectors in an end-to-end manner. Yi *et al.* [14] proposes a phrase-guided message-passing structure to learn the interdependency of the triplet components and predict them simultaneously. Zhang *et al.* [15] addresses it by using pairs of related regions in images to train a relationship proposer in order to reduce the related regions at test time.

C. Recurrent Neural Network and LSTM

Recurrent neural networks (RNN), especially the long-short term memory models [4], have achieved great success in many applications including natural language processing [17] and video processing [18]. Recently, RNN/LSTM has been widely applied in computer vision tasks such as image captioning [1] to generate language descriptions, natural language object retrieval [19] and referring image segmentation [20] to encode and comprehend language descriptions. As relationship phrases can be considered as a particular sequential representation (*sub* + *pred* + *obj*), we use LSTM to map the relationship triplet to a semantic embedding space in our work.

D. Zero-shot Learning

Zero-shot learning (ZSL) aims to recognize objects that are unseen during training. Humans have the ability to recognize objects without seeing these samples before but only based on some background knowledge, *e.g.*, attribute information and some similar objects. In computer vision, there is a surge of interest in ZSL recently [21]–[23]. Socher *et al.* [22] performs zero-shot learning by mapping the CNN visual feature vector to the semantic space. Lei *et al.* combines visual features and semantic features and learns a classifier based on the combined features [21]. In our work, we adopt an approach similar to [22].

III. Our Approach

Figure 2 shows an overview of our proposed model. Our model has a semantic branch and a visual branch. The goal of the semantic branch is to embed a triplet <*sub*, *pred*, *obj*> as a vector, while the goal of the visual branch is to embed the bounding boxes corresponding to the triplet as a vector. The distance of these two embedding vectors is used to indicate whether the visual information from bounding boxes and the triplet is a good match. In this section, we describe the details of each component of our model.

A. Visual Embedding

Given an input image, we first use a standard object detector (e.g. R-CNN [24], Faster-RCNN [6]) to detect the objects in an image. Given a pair of bounding boxes, the goal of visual embedding is to represent the visual information of the bounding boxes as a vector. In our work, we extract both appearance features and spatial features to form the visual embedding vector.

Appearance Features: Each detected object comes with a bounding box and an appearance feature extracted from the image patch within the bounding box. To obtain a fixed length appearance feature vector, a region of interest (ROI) [6] pooling layer is applied for each detected object box. The bounding box for the visual relationship (which we refer to as the predicate bounding box) can be obtained directly as the union of the bounding boxes for <*sub*, *obj*>. In the end, we extract three appearance features, one for each of the bounding boxes in the relationship <*sub*, *pred*, *obj*>. Each appearance feature has a dimension of 1000.

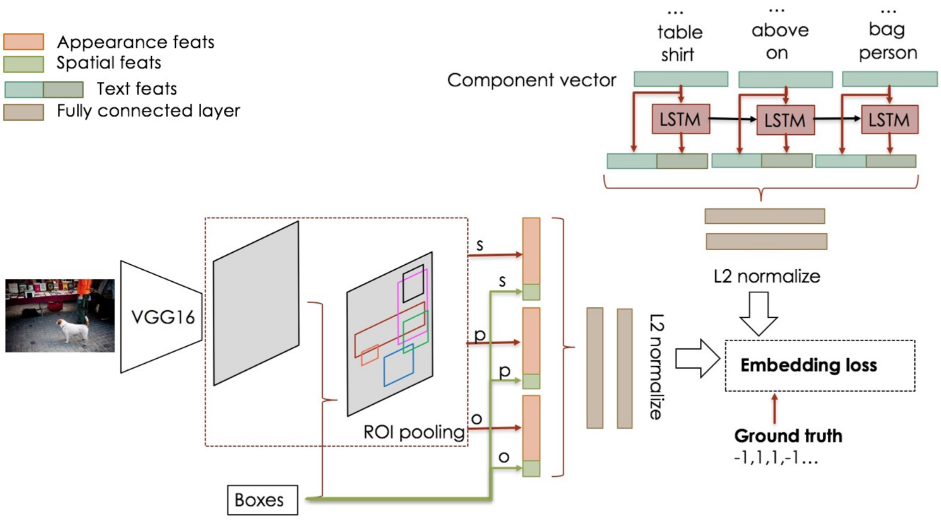


Fig. 2. Overview of our proposed model. The visual embedding branch (bottom) extracts appearance and spatial features from feature maps based on subject, predicate and object boxes as denoted with s, p and o. The semantic embedding branch (top) first embed the relationship components with vectors and then applies LSTM on these component vectors to encode the relationship triplet as a semantic vector. The projected semantic vector and visual vector should be close to each other if the relationship triplet is a good match to the visual information from the bounding boxes.

Spatial Features: The spatial relationship of the bounding boxes can be helpful in recognizing predicates such as spatial phrases or prepositions. From the object bounding boxes, we compute coordinate features as follows:

$$\begin{aligned} t_{xmin} &= \frac{x_{min}}{W}, t_{xmax} = \frac{x_{max}}{W}, \\ t_{ymin} &= \frac{y_{min}}{H}, t_{ymax} = \frac{y_{max}}{H}, \end{aligned} \tag{1}$$

where $(x_{min}, x_{max}, y_{min}, y_{max})$ represent the coordinates of subject/object/predicate box. W and H are the width and height of the input image.

Moreover, in order to keep the relative position of subject and object to be scale-invariant, we add another 4-dimension spatial features:

$$t_x = \frac{x - x'}{w'}, t_y = \frac{y - y'}{h'}, t_w = \log\frac{w}{w'}, t_h = \log\frac{h}{h'}, \tag{2}$$

where (x, y, w, h) and (x', y', w', h') represent subject/object and object/subject box coordinates, (t_x, t_y) specifies a scale-invariant translation and (t_w, t_h) specifies the relative height/width ratio. In the end, we get a 16-dimensional spatial feature vector representing the spatial information of each box pair.

The visual embedding vector is formed by concatenating the appearance features for <*sub*, *pred*, *obj*> and the spatial features.

B. Semantic Embedding

Given relationship triplets <*sub*, *pred*, *obj*> for one image, the goal of semantic embedding is to represent each triplet as a vector. In our work, we apply LSTM [4] to map the relationship triplet to a semantic embedding space.

LSTM Encoding: Assume that each component of a triplet is represented as a vector, we use $X = \{x_1, x_2, x_3\}$ to denote the relationship sequence of the input component vectors. Each LSTM unit includes three gates (*e.g.* input gate i, output gate o and forget gate f) and a memory cell c. At each time step t, given the input x_t and the previous hidden state h_{t-1}, LSTM updates as follows:

$$\begin{aligned} i_t &= \sigma(W_i x_t + U_i h_{t-1} + V_i c_{t-1} + b_i) \\ f_t &= \sigma(W_f x_t + U_f h_{t-1} + V_f c_{t-1} b_f) \\ z_t &= tanh(W_c x_t + U_c h_{t-1} + b_c) \\ c_t &= f_t \odot c_{t-1} + i_t \odot z_t \\ o_t &= \sigma(W_o x_t + U_o h_{t-1} + V_o c_t + b_o) \\ h_t &= o_t tanh(c_t) \end{aligned} \tag{3}$$

where σ is the sigmoid function and $\odot$ is the element-wise multiplication operator. W_*, U_* and V_* are the weight matrices, and b_* are the bias terms. The memory cell c_t is a weighted sum of the previous memory cell c_{t-1} and a function of the current input i_t. The last time step h_t can be viewed as an aggregated relationship information from the first time step to t, which contains the semantic context for this particular relationship.

Component Vectors: There are existing tools to embed words as vectors (*e.g.* word2vec [16], Glove [26]). We can integrate the vectors of object and subject classes as feature representations using pre-trained word2vec model which maps semantically similar words into similar vectors. This semantic similarity is commonly employed for *sub* and *obj* embeddings in previous work [9], [12]. But there are no off-the-shelf methods for embedding the relationship triplet. The pre-trained phrase vectors cannot be directly applied to produce relationship vectors because of different word combinations. In this work, we have experimented with two different strategies to obtain each component vector of a relationship triplet.

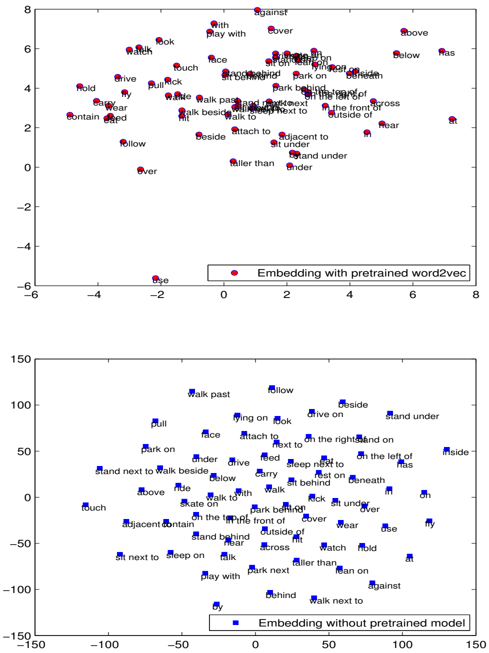


Fig. 3. An illustration of two methods for 70 predicate vectors in visual relationship dataset (VRD) [9] by using t-SNE visualizations [25]. This figure is best viewed with PDF magnification.

- We first attempt to use pre-trained word2vec [16] (we call it "w/ pre-trained"), where semantically similar words are mapped to vectors that are close (*e.g.* "adjacent to" and "next to"). Since some of the predicates contain more than one word (*e.g.* "in the front of", "next to"), we average each word vector for the whole predicate in this case as the second component vector of a relationship triplet. An illustration of 70 predicate vectors are given on the top of Fig. 3 by using the t-SNE visualization [25]. As a result, we have a 1000-dimension vector to represent each component of a triplet.
- We also experiment without pre-trained models (we call it "w/o pre-trained"). We define an index for each predicate and map all predicates to a $|V| \times D$ matrix, where D is the dimensionality of the embeddings and $|V|$ is the number of predicates. The weights are not initialized with pre-trained model, so each embedded component vector is independent and has no connection to each other. Accordingly, the 70 predicate vectors are reflected at the bottom of Fig. 3. We see that "adjacent to" and "next to" are not related to each other. We get the subject/object vectors in the same way. In the end, a 1000-dimension vector is formed for each component of a given <*sub*, *pred*, *obj*>.

Fused Semantic Embedding Vector: With the encoded feature vectors from LSTM, we concatenate them with each corresponding component vector of a triplet as the fused feature representation for the text branch. The intuition is that local evidence and context information both contribute to the text feature representation. Then they are followed by two more fully connected layers to make the output dimension same as that from the visual branch.

C. Loss Function

After we get the final embedding vectors from both branches, we apply the L1 hinge embedding loss (Eq. 4) to measure their distance:

$$l(x_1, x_2) = \begin{cases} ||x_1 - x_2||_1, & \text{if } y = 1 \\ max(0, m - ||x_1 - x_2||_1), & \text{if } y = -1 \end{cases} \quad (4)$$

where x_1 and x_2 are embedding vectors from the visual and text branches, and m is the margin with default value of 1. Label $y = 1$ if the visual representation and text representation match and $y = -1$ otherwise.

D. Testing

For a test image, we apply object detection first. With the detected object bounding boxes, visual and spatial features are extracted for each box pair. Suppose there are N objects detected, we will have $N(N-1)$ box pairs. We also filter out the box pairs that are too far away and unlikely to form a visual relationship. For each box pair, we compare the visual feature vector with the semantic embedding vector corresponding to a query triplet. The predicate associated with the query relationship phrase nearest to the given visual feature is the predicted interaction between the two object pair as showed in Eq. 5.

$$pred = \arg\min_{i \in M} ||V_{s,o} - T_{s,o}(P_i)||_1 \quad (5)$$

where $V_{s,o}$ denotes the visual embedding vector from a box pair <*sub*, obj>, $T_{s,o}(P_i)$ is the semantic embedding vector associated with predicate P_i for the object pair <*sub*, *obj*>. M is the number of predicates in the dataset.

The final relationship prediction score is calculated as:

$$S_{relation} = -d_p(1 - S_{sub})(1 - S_{obj}) \quad (6)$$

where d_p is shortest distance between $V_{s,o}$ and $T_{s,o}(P_i)$. S_{sub} and S_{obj} are the corresponding subject and object detection scores respectively.

IV. Experiments

In this section, we perform experimental evaluation of our proposed method and compare with other baseline approaches.

A. Dataset

We evaluate our work on the Visual relationship dataset (VRD) [9]. It contains 5000 images with 100 object categories and 70 predicates. There are 4000 images for training and 1000 for testing. In total, the dataset have 37,993 relationships with 6672 relationship types and on average 24.25 predicates per object category. Due to the long tail of relationship distribution in real world images, zero-shot predication of relationships that it has never seen before can alleviate stress of collecting every possible relationship. It is inevitable that some relationships like "computer on stove" never appear in the training data.

	Phrase Det.		Relation Det.	
	R@100	R@50	R@100	R@50
Lu's-V [9]	2.61	2.24	1.85	1.58
Lu's-VLK [9]	17.03	**16.17**	14.70	13.86
CLS	10.28	9.14	8.86	7.87
Ours (w/ pre-trained)	12.37	11.43	10.75	9.91
Ours (w/o pre-trained)	**17.28**	15.87	**15.34**	**14.00**
VTransE [11]*	22.42	19.42	15.20	14.07
Ours (w/o pre-trained*)	**24.12**	**20.53**	**16.26**	**14.23**

TABLE I
NON-ZERO-SHOT VISUAL RELATIONSHIP DETECTION ON VRD DATASET. * DENOTES USING FASTER-RCNN FOR OBJECT DETECTION. CLS TREATS PREDICATE PREDICTIONS AS A CLASSIFICATION PROBLEM BY USING CROSS ENTROPY LOSS WITH THREE TYPES OF FEATURES (APPEARANCE + SPATIAL + SUB AND OBJ WORD VECTORS). OURS (W/ PRE-TRAINED) OBTAINS EACH COMPONENT VECTOR OF A RELATIONSHIP TRIPLET BASED ON PRE-TRAINED WORD2VEC [16] AND WE AVERAGE VECTORS IF THE PREDICATE CONTAINS MORE THAN ONE WORD. OURS (W/O PRE-TRAINED) GETS EACH COMPONENT VECTOR OF A RELATIONSHIP TRIPLET WITHOUT PRE-TRAINED MODELS.

	Phrase Det.		Relation Det.	
	R@100	R@50	R@100	R@50
Lu's-V [9]	1.12	0.95	0.78	0.67
Lu's-VLK [9]	3.75	3.36	3.52	3.13
VTransE [11]*	3.51	2.65	2.14	1.71
CLS	4.45	3.85	4.19	3.59
Ours (w/ pre-trained)	5.73	**5.30**	5.30	**4.88**
Ours (w/o pre-trained)	**6.16**	5.05	**5.73**	4.79

TABLE II
ZERO-SHOT VISUAL RELATIONSHIP DETECTION ON VRD DATASET. * DENOTES USING FASTER-RCNN FOR OBJECT DETECTION.

The 1000 test image set contains 1,877 relationships that never occur in the training set, which allows us to evaluate for the zero-shot relationship detection task.

B. Evaluation Metric

Following [9], Recall@**x** is applied to measure the performance. This metric computes the fraction of times the correct relationship is predicated in the top **x** relationship predictions ranked by their confidence scores. Compared with mean average precision (mAP), Recall@**x** is more appropriate in this problem since the annotations on the dataset are incomplete. We evaluate two tasks on this dataset:

Phrase detection (Fig. 1 left): Given an input image and a query triplet $<sub, pred, obj>$, the goal is to localize the entire relationship with one bounding box. We consider the localization to be correct if the intersection-over-union (IoU) between the predicted bounding box and the ground-truth box is at least 0.5.

Relation detection (Fig. 1 right): Given an input image and a query triplet $<sub, pred, obj>$, the goal is to localize subject, predicate, object with separate bounding boxes. The localization is considered correct if all three bounding boxes have at 0.5 IoU with their corresponding ground-truth boxes.

C. Implementation Details

We use VGG16 [27] to obtain the feature maps that are pre-trained on PASCAL [28] for object detection [5]. To compare with [9] and [11], we use the object detection results provided in [9] and trained object detector provided in [11] respectively during the object detection stage. Other than the object detection, the rest of the architecture is trained end-to-end. The learning rate is 0.001, and is decreased by a factor of 10 every 10 epochs. Training is stopped when reaching 50 epochs and the loss almost does not change. Batch size is set to 1. During training, we sample negative samples by randomly selecting the box pairs in this image that their visual features do not match with their relationship triplets. We keep the positive and negative sample ratio as 1 and randomly shuffle these samples before training.

D. Results

The results for non-zero-shot and zero-shot visual relationship detection are shown in Tab. I and Tab. II respectively.

From Tab. I, ours (w/ pre-trained) does not perform very well. It is probably because the average of pre-trained word vectors cannot differentiate between predicates with overlapping words, such as "sleep next to", "stand next to" and "sit next to". Furthermore, some similar predicates (such as "next to" and "near") are treated as two different entries in the ground-truth annotation. The pre-trained word embedding considers these two predicates to be close to each other, so it is difficult to distinguish these two predicates. Ours (w/o pre-trained) achieves big improvement in terms of prediction accuracy. In particular, the performance of our method is either better than or comparable to other state-of-the-art approaches.

In the zero-shot visual relationship detection (Tab. II), our proposed methods clearly outperform other baselines. This demonstrates the advantage of the proposed embedding approach in the zero-shot scenario.

We also experiment with Faster-RCNN for object detection by using the trained object detector provided in [11] to compare with [11]. In Tab. I, Faster-RCNN has a significant impact on improvement in relationship detection. Ours (w/o pre-trained*) performs better than the state-of-the-art result in [11].

In Fig. 4, we show some qualitative results of both [9] and ours (w/o pre-trained). Fig. 5 displays sample results from ours (w/o pre-trained*).

V. CONCLUSION

In this paper, we have proposed a joint visual-semantic embedding model that maps the visual vector and semantic vector to a shared latent space to learn the similarity between the two branches. Our model can easily handle zero-shot visual relationship detection. We experiment on VRD dataset for phrase detection and relationship detection tasks. The proposed model achieves superior performance in zero-shot visual relationship detection and comparable results in non-zero-shot scenario.

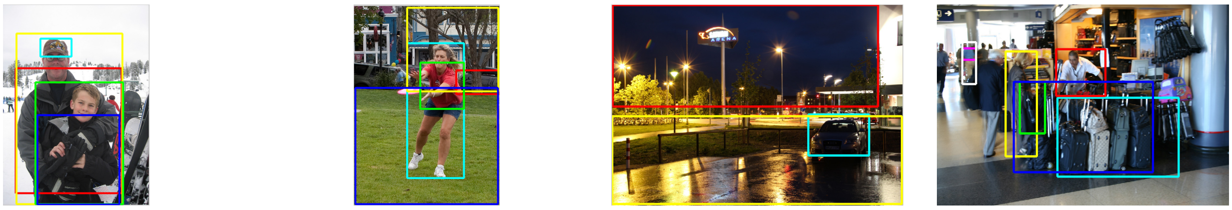
REFERENCES

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

[2] Z. Yu, J. Yu, J. Fan, and D. Tao, "Multi-modal factorized bilinear pooling with co-attention learning for visual question answering," *arXiv:1708.01471*, 2017.

Lu's-VLK [9]	<person, wear, plate> <person, on, table>	<pot, under, cup>	<person, has, shoes> <skies, under, person>	<cat, sit on, chair>
Ours (w/o pre-trained)	<person, has, plate> <person, next to, table>	<pot, next to, cup>	<person, wear, shoes> <skies, next to, person>	<cat, on, chair>

Fig. 4. Qualitative results. Predicates with blue color are correct predictions and those with red color are wrong predications.



hat, on, person person, wear, hat person, wear, jacket person, wear, jacket person, stand behind, person person, next to, person (person, in the front of, person)	car, behind, tree car, behind, person tree, behind, person person, wear, shirt person, stand on, grass tree, behind, car (tree, in the front of, car) (person, next to, tree) (person, in the front of, tree)	car, under, sky sky, above, street	luggage, next to, person luggage, next to, person person, next to, luggage person, wear, shirt bag, next to, person person, hold, bag (person, carry, bag)

Fig. 5. Sample results from ours (w/o pre-trained*). Predicates with blue color are correct predictions and those with red color are wrong predications. Predicates with green color in parenthesis are ground-truth labels corresponding to the previous row.

[3] L. Wang, Y. Li, and S. Lazebnik, "Learning deep structure-preserving image-text embeddings," in *CVPR*, 2016.

[4] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, 1997.

[5] R. Girshick, "Fast R-CNN," in *ICCV*, 2015.

[6] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *NIPS*, 2015.

[7] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *ECCV*, 2016. 21–37.

[8] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *CVPR*, 2016.

[9] C. Lu, R. Krishna, M. Bernstein, and L. Fei-Fei, "Visual relationship detection with language priors," in *ECCV*, 2016.

[10] B. Dai, Y. Zhang, and D. Lin, "Detecting visual relationships with deep relational networks," in *CVPR*, 2017.

[11] H. Zhang, Z. Kyaw, S.-F. Chang, and T.-S. Chua, "Visual translation embedding network for visual relation detection," in *CVPR*, 2017.

[12] B. Zhuang, L. Liu, C. Shen, and I. Reid, "Towards context-aware interaction recognition," in *ICCV*, 2017.

[13] R. Yu, A. Li, V. I. Morariu, and L. S. Davis, "Visual relationship detection with internal and external linguistic knowledge distillation," in *ICCV*, 2017.

[14] Y. Li, W. Ouyang, X. Wang, and X. Tang, "ViP-CNN: Visual phrase guided convolutional neural network," in *CVPR*, 2017.

[15] J. Zhang, M. Elhoseiny, S. Cohen, W. Chang, and A. Elgammal, "Relationship proposal networks," in *CVPR*, 2017.

[16] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv:1301.3781*, 2013.

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

[18] J. Yue-Hei Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond short snippets: Deep networks for video classification," in *CVPR*, 2015.

[19] R. Hu, H. Xu, M. Rohrbach, J. Feng, K. Saenko, and T. Darrell, "Natural language object retrieval," in *CVPR*, 2016.

[20] R. Hu, M. Rohrbach, and T. Darrell, "Segmentation from natural language expressions," in *ECCV*, 2016.

[21] J. Lei Ba, K. Swersky, S. Fidler *et al.*, "Predicting deep zero-shot convolutional neural networks using textual descriptions," in *ICCV*, 2015.

[22] R. Socher, M. Ganjoo, C. D. Manning, and A. Ng, "Zero-shot learning through cross-modal transfer," in *NIPS*, 2013.

[23] L. Zhang, T. Xiang, and S. Gong, "Learning a deep embedding model for zero-shot learning," *arXiv:1611.05088*, 2016.

[24] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *CVPR*, 2014.

[25] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *JMLR*, 2011.

[26] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *EMNLP*, 2014.

[27] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *ICLR*, 2015.

[28] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL visual object classes (VOC) challenge," *IJCV*, 2010.