

MAttNet: Modular Attention Network for Referring Expression Comprehension

Licheng Yu¹, Zhe Lin², Xiaohui Shen², Jimei Yang², Xin Lu²,
Mohit Bansal¹, Tamara L. Berg¹

¹University of North Carolina at Chapel Hill ²Adobe Research

Abstract

In this paper, we address referring expression comprehension: localizing an image region described by a natural language expression. While most recent work treats expressions as a single unit, we propose to decompose them into three modular components related to subject appearance, location, and relationship to other objects. This allows us to flexibly adapt to expressions containing different types of information in an end-to-end framework. In our model, which we call the Modular Attention Network (MAttNet), two types of attention are utilized: language-based attention that learns the module weights as well as the word/phrase attention that each module should focus on; and visual attention that allows the subject and relationship modules to focus on relevant image components. Module weights combine scores from all three modules dynamically to output an overall score. Experiments show that MAttNet outperforms previous state-of-the-art methods by a large margin on both bounding-box-level and pixel-level comprehension tasks. Demo¹ and code² are provided.

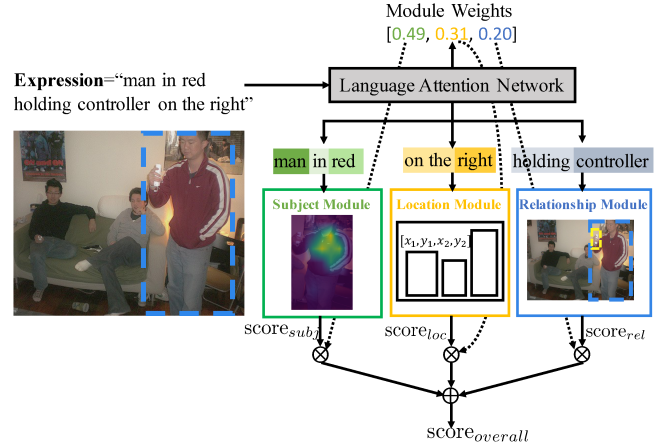


Figure 1: Modular Attention Network (MAttNet). Given an expression, we attentively parse it into three phrase embeddings, which are input to three visual modules that process the described visual region in different ways and compute individual matching scores. An overall score is then computed as a weighted combination of the module scores.

1. Introduction

Referring expressions are natural language utterances that indicate particular objects within a scene, e.g., “the woman in the red sweater” or “the man on the right”. For robots or other intelligent agents communicating with people in the world, the ability to accurately comprehend such expressions in real-world scenarios will be a necessary component for natural interactions.

Referring expression comprehension is typically formulated as selecting the best region from a set of proposals/objects $O = \{o_i\}_{i=1}^N$ in image I , given an input expression r . Most recent work on referring expressions uses CNN-LSTM based frameworks to model $P(r|o)$ [19, 11, 32, 20, 18] or uses a joint vision-language embedding framework to model $P(r, o)$ [22, 26, 27]. During test-

ing, the proposal/object with highest likelihood/probability is selected as the predicted region. However, most of these work uses a simple concatenation of all features (target object feature, location feature and context feature) as input and a single LSTM to encode/decode the whole expression, ignoring the variance among different types of referring expressions. Depending on what is distinctive about a target object, different kinds of information might be mentioned in its referring expression. For example, if the target object is a red ball among 10 black balls then the referring expression may simply say “the red ball”. If that same red ball is placed among 3 other red balls then location-based information may become more important, e.g., “red ball on the right”. Or, if there were 100 red balls in the scene then the ball’s relationship to other objects might be the most distinguishing information, e.g., “red ball next to the cat”. Therefore, it is natural and intuitive to think about the com-

¹Demo: vision2.cs.unc.edu/refer/comprehension

²Code: <https://github.com/lichengunc/MAttNet>

prehension model as a modular network, where different visual processing modules are triggered based on what information is present in the referring expression.

Modular networks have been successfully applied to address other tasks such as (visual) question answering [2, 3], visual reasoning [8, 12], relationship modeling [10], and multi-task reinforcement learning [1]. To the best of our knowledge, we present the first modular network for the general referring expression comprehension task. Moreover, these previous works typically rely on an off-the-shelf language parser [24] to parse the query sentence/question into different components and dynamically assemble modules into a model addressing the task. However, the external parser could raise parsing errors and propagate them into the model setup, adversely affecting performance.

Therefore, in this paper we propose a modular network for referring expression comprehension - Modular Attention Network (MAttNet) - that takes a natural language expression as input and softly decomposes it into three phrase embeddings. These embeddings are used to trigger three separate visual modules (for subject, location, and relationship comprehension, each with a different attention model) to compute matching scores, which are finally combined into an overall region score based on the module weights. Our model is illustrated in Fig. 1. There are 3 main novelties in MAttNet.

First, MAttNet is designed for general referring expressions. It consists of 3 modules: subject, location and relationship. As in [13], a referring expression could be parsed into 7 attributes: category name, color, size, absolute location, relative location, relative object and generic attribute. MAttNet covers all of them. The subject module handles the category name, color and other attributes, the location module handles both absolute and (some) relative location, and the relationship module handles subject-object relations. Each module has a different structure and learns the parameters within its own modular space, without affecting the others.

Second, MAttNet learns to parse expressions automatically through a soft attention based mechanism, instead of relying on an external language parser [24, 13]. We show that our learned “parser” attends to the relevant words for each module and outperforms an off-the-shelf parser by a large margin. Additionally, our model computes module weights which are adaptive to the input expression, measuring how much each module should contribute to the overall score. Expressions like “red cat” will have larger subject module weights and smaller location and relationship module weights, while expressions like “woman on left” will have larger subject and location module weights.

Third, we apply different visual attention techniques in the subject and relationship modules to allow relevant attention on the described image portions. In the subject mod-

ule, soft attention attends to the parts of the object itself mentioned by an expression like “man in red shirt” or “man with yellow hat”. We call this “in-box” attention. In contrast, in the relationship module, hard attention is used to attend to the relational objects mentioned by expressions like “cat on chair” or “girl holding frisbee”. Here the attention focuses on “chair” and “frisbee” to pinpoint the target object “cat” and “girl”. We call this “out-of-box” attention. We demonstrate both attentions play important roles in improving comprehension accuracy.

During training, the only supervision is object proposal, referring expression pairs, (o_i, r_i) , and all of the above are automatically learned in an end-to-end unsupervised manner, including the word attention, module weights, soft spatial attention, and hard relative object attention.

We demonstrate MAttNet has significantly superior comprehension performance over all state-of-the-art methods, achieving $\sim 10\%$ improvements on bounding-box localization and almost doubling precision on pixel segmentation.

2. Related Work

Referring Expression Comprehension: The task of referring expression comprehension is to localize a region described by a given referring expression. To address this problem, some recent work [19, 32, 20, 11, 18] uses CNN-LSTM structure to model $P(r|o)$ and looks for the object o maximizing the probability. Other recent work uses joint embedding model [22, 26, 16, 4] to compute $P(o|r)$ directly. In a hybrid of both types of approaches, [33] proposed a joint speaker-listener-reinforcer model that combined CNN-LSTM (speaker) with embedding model (listener) to achieve state-of-the-art results.

Most of the above treat comprehension as bounding box localization, but object segmentation from referring expression has also been studied in some recent work [9, 15]. These papers use FCN-style [17] approaches to perform expression-driven foreground/background classification. We demonstrate that in addition to bounding box prediction, we also outperform previous segmentation results.

Modular Networks: Neural module networks [3] were introduced for visual question answering. These networks decompose the question into several components and dynamically assemble a network to compute an answer to the given question. Since their introduction, modular networks have been applied to several other tasks: visual reasoning [8, 12], question answering [2], relationship modeling [10], multitask reinforcement learning [1], etc. While the early work [3, 12, 2] requires an external language parser to do the decomposition, recent methods [10, 8] propose to learn the decomposition end-to-end. We apply this idea to referring expression comprehension, also taking an end-to-end approach by bypassing the use of an external parser.

We find that our soft attention approach achieves better performance over the hard decisions predicted by a parser.

The most related work to us is [10], which decomposes the expression into (Subject, Preposition/Verb, Object) triples. However, referring expressions have much richer forms than this fixed template. For example, expressions like “left dog” and “man in red” are hard to model using [10]. In this paper, we propose a generic modular network addressing all kinds of referring expressions. Our network is adaptive to the input expression by assigning both word-level attention and module-level weights.

3. Model

MAttNet is composed of a language attention network plus visual subject, location, and relationship modules. Given a candidate object o_i and referring expression r , we first use the language attention network to compute a soft parse of the referring expression into three components (one for each visual module) and map each to a phrase embedding. Second, we use the three visual modules (with unique attention mechanisms) to compute matching scores for o_i to their respective embeddings. Finally, we take a weighted combination of these scores to get an overall matching score, measuring the compatibility between o_i and r .

3.1. Language Attention Network

Instead of using an external language parser [24][3][2] or pre-defined templates [13] to parse the expression, we propose to learn to attend to the relevant words automatically for each module, similar to [10]. Our language attention network is shown in Fig. 2. For a given expression $r = \{u_t\}_{t=1}^T$, we use a bi-directional LSTM to encode the context for each word. We first embed each word u_t into a vector e_t using an one-hot word embedding, then a bi-directional LSTM-RNN is applied to encode the whole expression. The final hidden representation for each word is the concatenation of the hidden vectors in both directions:

$$\begin{aligned} e_t &= \text{embedding}(u_t) \\ \vec{h}_t &= \text{LSTM}(e_t, \vec{h}_{t-1}) \\ \tilde{h}_t &= \text{LSTM}(e_t, \tilde{h}_{t+1}) \\ h_t &= [\vec{h}_t, \tilde{h}_t]. \end{aligned}$$

Given $H = \{h_t\}_{t=1}^T$, we apply three trainable vectors f_m where $m \in \{\text{subj}, \text{loc}, \text{rel}\}$, computing the attention on each word [29] for each module:

$$a_{m,t} = \frac{\exp(f_m^T h_t)}{\sum_{k=1}^T \exp(f_m^T h_k)}$$

The weighted sum of word embeddings is used as the modular phrase embedding:

$$q^m = \sum_{t=1}^T a_{m,t} e_t$$

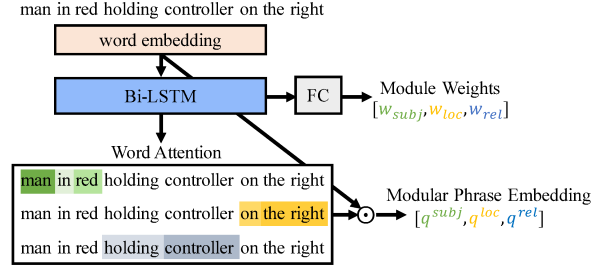


Figure 2: Language Attention Network

Different from relationship detection [10] where phrases are always decomposed as (Subject, Preposition/Verb, Object) triplets, referring expressions have no such well-posed structure. For example, expressions like “smiling boy” only contain language relevant to the subject module, while expressions like “man on left” are relevant to the subject and location modules, and “cat on the chair” are relevant to the subject and relationship modules. To handle this variance, we compute 3 module weights for the expression, weighting how much each module contributes to the expression-object score. We concatenate the first and last hidden vectors from H which memorizes both structure and semantics of the whole expression, then use another fully-connected (FC) layer to transform it into 3 module weights:

$$[w_{\text{subj}}, w_{\text{loc}}, w_{\text{rel}}] = \text{softmax}(W_m^T [h_0, h_T] + b_m)$$

3.2. Visual Modules

While most previous work [32, 33, 19, 20] evaluates CNN features for each region proposal/candidate object, we use Faster R-CNN [21] as the backbone net for a faster and more principled implementation. Additionally, we use ResNet [7] as our main feature extractor, but also provide comparisons to previous methods using the same VGGNet features [23] (in Sec. 4.2).

Given an image and a set of candidates o_i , we run Faster R-CNN to extract their region representations. Specifically, we forward the whole image into Faster R-CNN and crop the C3 feature (last convolutional output of 3rd-stage) for each o_i , following which we further compute the C4 feature (last convolutional output of 4th-stage). In Faster R-CNN, C4 typically contains higher-level visual cues for category prediction, while C3 contains relatively lower-level cues including colors and shapes for proposal judgment, making both useful for our purposes. In the end, we compute the matching score for each o_i given each modular phrase embedding, i.e., $S(o_i|q^{\text{subj}})$, $S(o_i|q^{\text{loc}})$ and $S(o_i|q^{\text{rel}})$.

3.2.1 Subject Module

Our subject module is illustrated in Fig. 3. Given the C3 and C4 features of a candidate o_i , we forward them to two

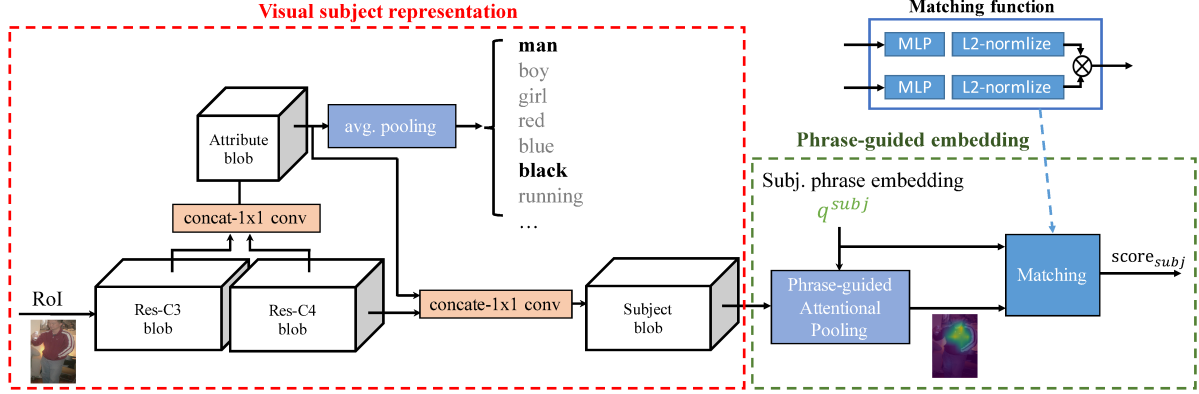


Figure 3: The subject module is composed of a visual subject representation and phrase-guided embedding. An attribute prediction branch is added after the ResNet-C4 stage and the 1x1 convolution output of attribute prediction and C4 is used as the subject visual representation. The subject phrase embedding attentively pools over the spatial region and feeds the pooled feature into the matching function.

tasks. The first is attribute prediction, helping produce a representation that can understand appearance characteristics of the candidate. The second is the phrase-guided attentional pooling to focus on relevant regions within object bounding boxes.

Attribute Prediction: Attributes are frequently used in referring expressions to differentiate between objects of the same category, e.g. “woman in red” or “the fuzzy cat”. Inspired by previous work [30, 28, 31, 16, 25], we add an attribute prediction branch in our subject module. While preparing the attribute labels in the training set, we first run a template parser [13] to obtain color and generic attribute words, with low-frequency words removed. We combine both C3 and C4 for predicting attributes as both low and high-level visual cues are important. The concatenation of C3 and C4 is followed with a 1×1 convolution to produce an attribute feature blob. After average pooling, we get the attribute representation of the candidate region. A binary cross-entropy loss is used for multi-attribute classification:

$$L_{subj}^{attr} = \lambda_{attr} \sum_i \sum_j w_j^{attr} [\log(p_{ij}) + (1 - y_{ij}) \log(1 - p_{ij})]$$

where $w_j^{attr} = 1/\sqrt{\text{freq}_{attr}}$ weights the attribute labels, easing unbalanced data issues. During training, only expressions with attribute words go through this branch.

Phrase-guided Attentional Pooling: The subject description varies depending on what information is most salient about the object. Take people for example. Sometimes a person is described by their accessories, e.g., “girl in glasses”; or sometimes particular clothing items may be mentioned, e.g., “woman in white pants”. Thus, we allow our subject module to localize relevant regions within a bounding box through “in-box” attention. To compute spatial attention, we first concatenate the attribute blob and C4,

then use a 1×1 convolution to fuse them into a subject blob, which consists of spatial grid of features $V \in R^{d \times G}$, where $G = 14 \times 14$. Given the subject phrase embedding q^{subj} , we compute its attention on each grid location:

$$H_a = \tanh(W_v V + W_q q^{subj})$$

$$a^v = \text{softmax}(w_{h,a}^T H_a)$$

The weighted sum of V is the final subject visual representation for the candidate region o_i :

$$\tilde{v}_i^{subj} = \sum_{i=1}^G a_i^v v_i$$

Matching Function: We measure the similarity between the subject representation $\tilde{v}_i^{subj}$ and phrase embedding q_{subj} using a matching function, i.e., $S(o_i | q^{subj}) = F(\tilde{v}_i^{subj}, q^{subj})$. As shown in top-right of Fig. 3, it consists of two MLPs (multi-layer perceptions) and two L2 normalization layers following each input. Each MLP is composed of two fully connected layers with ReLU activations, serving to transform the visual and phrase representation into a common embedding space. The inner-product of the two l2-normalized representations is computed as their similarity score. The same matching function is used to compute the location score $S(o_i | q^{loc})$, and relationship score $S(o_i | q^{rel})$.

3.2.2 Location Module

Our location module is shown in Fig. 4. Location is frequently used in referring expressions with about 41% expressions from RefCOCO and 36% expressions from RefCOCOG containing absolute location words [13], e.g. “cat on the right” indicating the object location in the image.

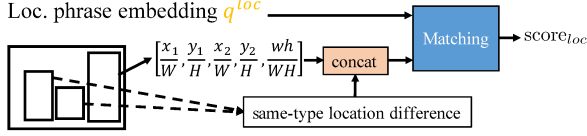


Figure 4: Location Module

Following previous work [32][33], we use a 5-d vector l_i to encode the top-left position, bottom-right position and relative area to the image for the candidate object, i.e., $l_i = [\frac{x_{tl}}{W}, \frac{y_{tl}}{H}, \frac{x_{br}}{W}, \frac{y_{br}}{H}, \frac{w \cdot h}{W \cdot H}]$.

Additionally, expressions like “dog in the middle” and “second left person” imply relative positioning among objects of the same category. We encode the relative location representation of a candidate object by choosing up to five surrounding objects of the same category and calculating their offsets and area ratio, i.e., $\delta l_{ij} = [\frac{[\Delta x_{tl}]_{ij}}{w_i}, \frac{[\Delta y_{tl}]_{ij}}{h_i}, \frac{[\Delta x_{br}]_{ij}}{w_i}, \frac{[\Delta y_{br}]_{ij}}{h_i}, \frac{w_j h_j}{w_i h_i}]$. The final location representation for the target object is:

$$\tilde{l}_i^{loc} = W_l[l_i; \delta l_i] + b_l$$

and the location module matching score between o_i and q^{loc} is $S(o_i|q^{loc}) = F(\tilde{l}_i^{loc}, q^{loc})$.

3.2.3 Relationship Module

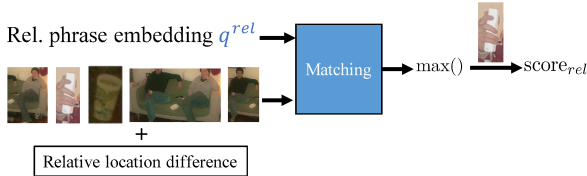


Figure 5: Relationship Module

While the subject module deals with “in-box” details about the target object, some other expressions may involve its relationship with other “out-of-box” objects, e.g., “cat on chaise lounge”. The relationship module is used to address these cases. As in Fig. 5, given a candidate object o_i we first look for its surrounding (up-to-five) objects o_{ij} regardless of their categories. We use the average-pooled C4 feature as the appearance feature v_{ij} of each supporting object. Then, we encode their offsets to the candidate object via $\delta m_{ij} = [\frac{[\Delta x_{tl}]_{ij}}{w_i}, \frac{[\Delta y_{tl}]_{ij}}{h_i}, \frac{[\Delta x_{br}]_{ij}}{w_i}, \frac{[\Delta y_{br}]_{ij}}{h_i}, \frac{w_j h_j}{w_i h_i}]$. The visual representation for each surrounding object is then:

$$\tilde{v}_{ij}^{rel} = W_r[v_{ij}; \delta m_{ij}] + b_r$$

We compute the matching score for each of them with q^{rel} and pick the highest one as the relationship score, i.e.,

$$S(o_i|q^{rel}) = \max_{j \neq i} F(\tilde{v}_{ij}^{rel}, q^{rel})$$

This can be regarded as weakly-supervised Multiple Instance Learning (MIL) which is similar to [10][20].

3.3. Loss Function

The overall weighted matching score for candidate object o_i and expression r is:

$$S(o_i|r) = w_{subj}S(o_i|q^{subj}) + w_{loc}S(o_i|q^{loc}) + w_{rel}S(o_i|q^{rel}) \quad (1)$$

During training, for each given positive pair of (o_i, r_i) , we randomly sample two negative pairs (o_i, r_j) and (o_k, r_i) , where r_j is the expression describing some other object and o_k is some other object in the same image, to calculate a combined hinge loss,

$$L_{rank} = \sum_i [\lambda_1 \max(0, \Delta + S(o_i|r_j) - S(o_i|r_i)) + \lambda_2 \max(0, \Delta + S(o_k|r_i) - S(o_i|r_i))]$$

The overall loss incorporates both attributes cross-entropy loss and ranking loss: $L = L_{subj}^{attr} + L_{rank}$.

4. Experiments

4.1. Datasets

We use 3 referring expression datasets: RefCOCO, RefCOCO+ [13], and RefCOCOg [19] for evaluation, all collected on MS COCO images [14], but with several differences. 1) RefCOCO and RefCOCO+ were collected in an interactive game interface, while RefCOCOg was collected in a non-interactive setting thereby producing longer expressions, 3.5 and 8.4 words on average respectively. 2) RefCOCO and RefCOCO+ contain more same-type objects, 3.9 vs 1.63 respectively. 3) RefCOCO+ forbids using absolute location words, making the data more focused on appearance differentiators.

During testing, RefCOCO and RefCOCO+ provide person vs. object splits for evaluation, where images containing multiple people are in “testA” and those containing multiple objects of other categories are in “testB”. There is no overlap between training, validation and testing images. RefCOCOg has two types of data partitions. The first [19] divides the dataset by randomly partitioning objects into training and validation splits. As the testing split has not been released, most recent work evaluates performance on the validation set. We denote this validation split as RefCOCOg’s “val*”. Note, since this data is split by objects the same image could appear in both training and validation. The second partition [20] is composed by randomly partitioning images into training, validation and testing splits. We denote its validation and testing splits as RefCOCOg’s “val” and “test”, and run most experiments on this split.

4.2. Results: Referring Expression Comprehension

Given a test image, I , with a set of proposals/objects, $O = \{o_i\}_{i=1}^N$, we use Eqn. 1 to compute the matching score

		feature	RefCOCO			RefCOCO+			RefCOCOg		
			val	testA	testB	val	testA	testB	val*	val	test
1	Mao [19]	vgg16	-	63.15	64.21	-	48.73	42.13	62.14	-	-
2	Varun [20]	vgg16	76.90	75.60	78.00	-	-	-	-	-	68.40
3	Luo [18]	vgg16	-	74.04	73.43	-	60.26	55.03	65.36	-	-
4	CMN [10]	vgg16-frcn	-	-	-	-	-	-	69.30	-	-
5	Speaker/visdif [32]	vgg16	76.18	74.39	77.30	58.94	61.29	56.24	59.40	-	-
6	Listener [33]	vgg16	77.48	76.58	78.94	60.50	61.39	58.11	71.12	69.93	69.03
7	Speaker+Listener+Reinforcer [33]	vgg16	79.56	78.95	80.22	62.26	64.60	59.62	72.63	71.65	71.92
8	Speaker+Listener+Reinforcer [33]	vgg16	78.36	77.97	79.86	61.33	63.10	58.19	72.02	71.32	71.72
9	MAttN:subj(+attr)+loc(+dif)+rel	vgg16	80.94	79.99	82.30	63.07	65.04	61.77	73.08	73.04	72.79
10	MAttN:subj(+attr)+loc(+dif)+rel	res101-frcn	83.54	82.66	84.17	68.34	69.93	65.90	-	76.63	75.92
11	MAttN:subj(+attr+attn)+loc(+dif)+rel	res101-frcn	85.65	85.26	84.57	71.01	75.13	66.17	-	78.10	78.12

Table 1: Comparison with state-of-the-art approaches on ground-truth MS COCO regions.

		RefCOCO			RefCOCO+			RefCOCOg	
		val	testA	testB	val	testA	testB	val	test
1	Matching:subj+loc	79.14	79.42	80.42	62.17	63.53	59.87	70.45	70.92
2	MAttN:subj+loc	79.68	80.20	81.49	62.71	64.20	60.65	72.12	72.62
3	MAttN:subj+loc(+dif)	82.06	81.28	83.20	64.84	65.77	64.55	75.33	74.46
4	MAttN:subj+loc(+dif)+rel	82.54	81.58	83.34	65.84	66.59	65.08	75.96	74.56
5	MAttN:subj(+attr)+loc(+dif)+rel	83.54	82.66	84.17	68.34	69.93	65.90	76.63	75.92
6	MAttN:subj(+attr+attn)+loc(+dif)+rel	85.65	85.26	84.57	71.01	75.13	66.17	78.10	78.12
7	parser+MAttN:subj(+attr+attn)+loc(+dif)+rel	80.20	79.10	81.22	66.08	68.30	62.94	73.82	73.72

Table 2: Ablation study of MAttNet using different combination of modules. The feature used here is res101-frcn.

$S(o_i|r)$ for each proposal/object given the input expression r , and pick the one with the highest score. For evaluation, we compute the intersection-over-union (IoU) of the selected region with the ground-truth bounding box, considering $\text{IoU} > 0.5$ a correct comprehension.

First, we compare our model with previous methods using COCO’s ground-truth object bounding boxes as proposals. Results are shown in Table. 1. As all of the previous methods (Line 1-8) used a 16-layer VGGNet (vgg16) as the feature extractor, we run our experiments using the same feature for fair comparison. Note the flat fc7 is a single 4096-dimensional feature which prevents us from using the phrase-guided attentional pooling in Fig. 3, so we use average pooling for subject matching. Despite this, our results (Line 9) still outperform all previous state-of-the-art methods. After switching to the res101-based Faster R-CNN (res101-frcn) representation, the comprehension accuracy further improves another $\sim 3\%$ (Line 10). Note our Faster R-CNN is pre-trained on COCO’s training images, excluding those in RefCOCO, RefCOCO+, and RefCOCOg’s validation+testing. Thus no training images are seen during our evaluation³. Our full model (Line 11) with phrase-guided attentional pooling achieves the highest accuracy over all others by a large margin.

Second, we study the benefits of each module of MAttNet by running ablation experiments (Table. 2) with the

³Such constraint forbids us to evaluate on RefCOCOg’s val* using the res101-frcn feature in Table 1.

same res101-frcn features. As a baseline, we use the concatenation of the regional visual feature and the location feature as the visual representation and the last hidden output of LSTM-encoded expression as the language representation, then feed them into the matching function to obtain the similarity score (Line 1). Compared with this, a simple two-module MAttNet using the same features (Line 2) already outperforms the baseline, showing the advantage of modular learning. Line 3 shows the benefit of encoding location (Sec. 3.2.2). After adding the relationship module, the performance further improves (Line 4). Lines 5 and Line 6 show the benefits brought by the attribute sub-branch and the phrase-guided attentional pooling in our subject module. We find the attentional pooling (Line 6) greatly improves on the person category (testA of RefCOCO and RefCOCO+), demonstrating the advantage of modular attention on understanding localized details like “girl with red hat”.

Third, we tried training our model using 3 hard-coded phrases from a template language parser [13], shown in Line 7 of Table. 2, which is $\sim 5\%$ lower than our end-to-end model (Line 6). The main reason for this drop is errors made by the external parser which is not tuned for referring expressions.

Fourth, we show results using automatically detected objects from Faster R-CNN, providing an analysis of fully automatic comprehension performance. Table. 3 shows the ablation study of fully-automatic MAttNet. While perfor-

		detector	RefCOCO			RefCOCO+			RefCOCOg	
			val	testA	testB	val	testA	testB	val	test
1	Speaker +Listener+Reinforcer [33]	res101-frcn	69.48	73.71	64.96	55.71	60.74	48.80	60.21	59.63
2	Speaker+ Listener +Reinforcer [33]	res101-frcn	68.95	73.10	64.85	54.89	60.04	49.56	59.33	59.21
3	Matching:subj+loc	res101-frcn	72.28	75.43	67.87	58.42	61.46	52.73	64.15	63.25
4	MAttN:subj+loc	res101-frcn	72.72	76.17	68.18	58.70	61.65	53.41	64.40	63.74
5	MAttN:subj+loc(+dif)	res101-frcn	72.96	76.61	68.20	58.91	63.06	55.19	64.66	63.88
6	MAttN:subj+loc(+dif)+rel	res101-frcn	73.25	76.77	68.44	59.45	63.31	55.68	64.87	64.01
7	MAttN:subj(+attr)+loc(+dif)+rel	res101-frcn	74.51	77.81	68.39	62.13	66.33	55.75	65.33	65.19
8	MAttN:subj(+attr+attn)+loc(+dif)+rel	res101-frcn	76.40	80.43	69.28	64.93	70.26	56.00	66.67	67.01
9	MAttN:subj(+attr+attn)+loc(+dif)+rel	res101-mrcn	76.65	81.14	69.99	65.33	71.62	56.02	66.58	67.27

Table 3: Ablation study of MAttNet on fully-automatic comprehension task using different combination of modules. The features used here are res101-frcn, except the last row using res101-mrcn.

RefCOCO								
Model	Backbone Net	Split	Pr@0.5	Pr@0.6	Pr@0.7	Pr@0.8	Pr@0.9	IoU
D+RMI+DCRF [15]	res101-DeepLab	val	42.99	33.24	22.75	12.11	2.23	45.18
MAttNet	res101-mrcn	val	75.16	72.55	67.83	54.79	16.81	56.51
D+RMI+DCRF [15]	res101-DeepLab	testA	42.99	33.59	23.69	12.94	2.44	45.69
MAttNet	res101-mrcn	testA	79.55	77.60	72.53	59.01	13.79	62.37
D+RMI+DCRF [15]	res101-DeepLab	testB	44.99	32.21	22.69	11.84	2.65	45.57
MAttNet	res101-mrcn	testB	68.87	65.06	60.02	48.91	21.37	51.70

RefCOCO+								
Model	Backbone Net	Split	Pr@0.5	Pr@0.6	Pr@0.7	Pr@0.8	Pr@0.9	IoU
D+RMI+DCRF [15]	res101-DeepLab	val	20.52	14.02	8.46	3.77	0.62	29.86
MAttNet	res101-mrcn	val	64.11	61.87	58.06	47.42	14.16	46.67
D+RMI+DCRF [15]	res101-DeepLab	testA	21.22	14.43	8.99	3.91	0.49	30.48
MAttNet	res101-mrcn	testA	70.12	68.48	63.97	52.13	12.28	52.39
D+RMI+DCRF [15]	res101-DeepLab	testB	20.78	14.56	8.80	4.58	0.80	29.50
MAttNet	res101-mrcn	testB	54.82	51.73	47.27	38.58	17.00	40.08

RefCOCOg								
Model	Backbone Net	Split	Pr@0.5	Pr@0.6	Pr@0.7	Pr@0.8	Pr@0.9	IoU
MAttNet	res101-mrcn	val	64.48	61.52	56.50	43.97	14.67	47.64
MAttNet	res101-mrcn	test	65.60	62.92	57.31	44.44	12.55	48.61

Table 4: Comparison of segmentation performance on RefCOCO, RefCOCO+, and our results on RefCOCOg.

mance drops due to detection errors, the overall improvements brought by each module are consistent with Table. 2, showing the robustness of MAttNet. Our results also outperform the state-of-the-art [33] (Line 1,2) with a big margin. Besides, we show the performance when using the detector branch of Mask R-CNN [6] (res101-mrcn) in Line 9, whose results are even better than using Faster R-CNN.

Finally, we show some example visualizations of comprehension using our full model in Fig. 6 as well as visualizations of the attention predictions. We observe that our language model is able to attend to the right words for each module even though it is learned in a weakly-supervised manner. We also observe the expressions in RefCOCO and RefCOCO+ describe the location or details of the target object more frequently while RefCOCOg mentions the relationship between target object and its surrounding object

more frequently, which accords with the dataset property. Note that for some complex expressions like “woman in plaid jacket and blue pants on skis” which contains several relationships (last row in Fig. 6), our language model is able to attend to the portion that should be used by the “in-box” subject module and the portion that should be used by the “out-of-box” relationship module. Additionally our subject module also displays reasonable spatial “in-box” attention, which qualitatively explains why attentional pooling (Table. 2 Line 6) outperforms average pooling (Table. 2 Line 5). For comparison, some incorrect comprehension are shown in Fig. 7. Most errors are due to sparsity in the training data, ambiguous expressions, or detection error.

4.3. Segmentation from Referring Expression

Our model can also be used to address referential object segmentation [9, 15]. Instead of using Faster R-

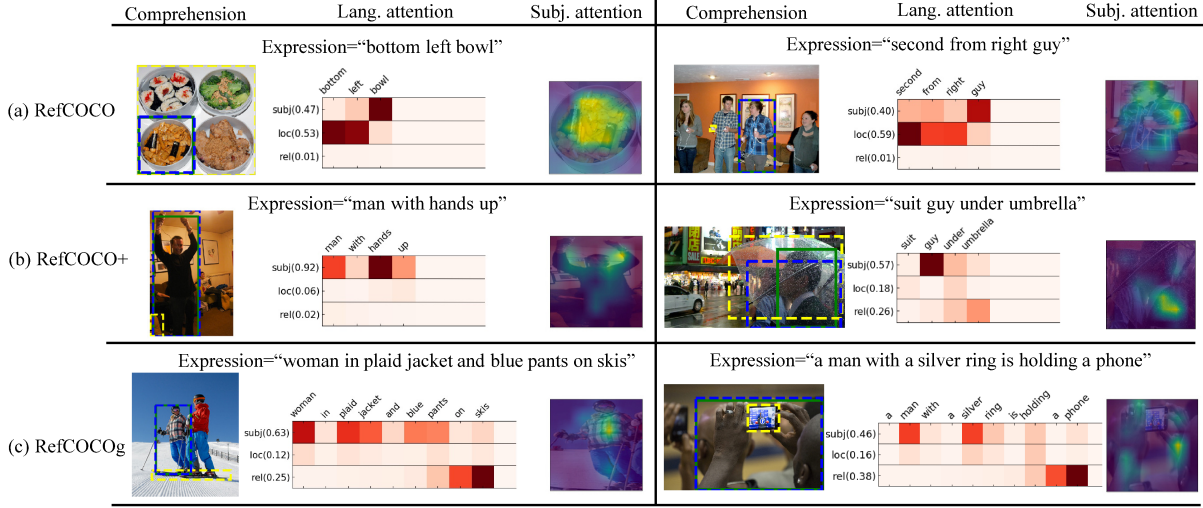


Figure 6: Examples of fully automatic comprehension. The blue dotted boxes show our prediction with the relative regions in yellow dotted boxes, and the green boxes are the ground-truth. The word attention is multiplied by module weight.

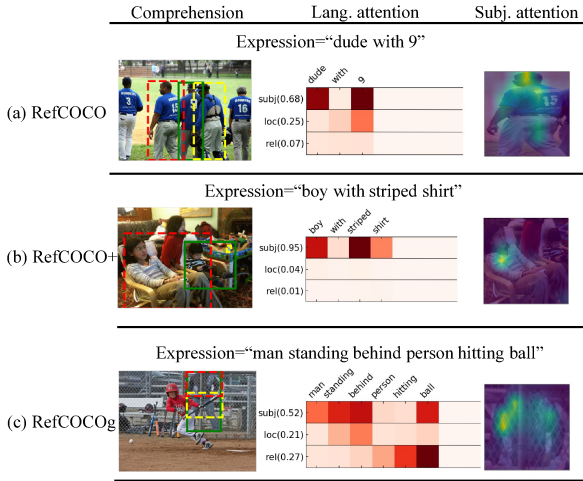


Figure 7: Examples of incorrect comprehensions. Red dotted boxes show our wrong prediction.

CNN as the backbone net, we now turn to res101-based Mask R-CNN [6] (res101-mrcn). We apply the same procedure described in Sec. 3 on the detected objects, and use the one with highest matching score as our prediction. Then we feed the predicted bounding box to the mask branch to obtain a pixel-wise segmentation. We evaluate the full model of MAttNet and compare with the best results reported in [15]. We use Precision@X ($X \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$)⁴ and overall Intersection-over-Union (IoU) as metrics. Results are shown in Table. 4

⁴Precision@0.5 is the percentage of expressions where the IoU of the predicted segmentation and ground-truth is at least 0.5.

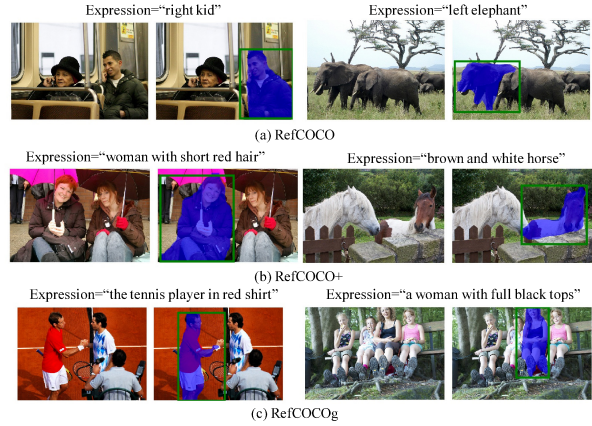


Figure 8: Examples of fully-automatic MAttNet referential segmentation.

with our model outperforming state-of-the-art results by a large margin under all metrics⁵. As both [15] and MAttNet use res101 features, such big gains may be due to our proposed model. We believe decoupling box localization (comprehension) and segmentation brings a large gain over FCN-style [17] foreground/background mask classification [9, 15] for this instance-level segmentation problem, but a more end-to-end segmentation system may be studied in future work. Some referential segmentation examples are shown in Fig. 8.

⁵There is no experiments on RefCOCOG's val/test splits in [15], so we show our performance only for reference in Table 4.

5. Conclusion

Our modular attention network addresses variance in referring expressions by attending to both relevant words and visual regions in a modular framework, and dynamically computing an overall matching score. We demonstrate our model’s effectiveness on bounding-box-level and pixel-level comprehension, significantly outperforming state-of-the-art.

Acknowledgements: This research is supported by NSF Awards #1405822, 1562098, 1633295, NVidia, Google Research, Microsoft Research and Adobe Research.

A. Appendix

A.1. Training Details

We optimize our model using Adam with an initial learning rate of 0.0004 and with a batch size of 15 images (and all their expressions). The learning rate is halved every 8,000 iterations after the first 8,000-iteration warm-up. The word embedding size and hidden state size of the LSTM are set to 512. We also set the output of all MLPs and FCs within our model to be 512-dimensional. To avoid overfitting, we regularize the word-embedding and output layers of the LSTM in the language attention network using dropout with ratio of 0.5. We also regularize the two inputs (visual and language) of matching function using a dropout with a ratio of 0.2. For the constrastive pairs, we set $\lambda_1 = 1.0$ and $\lambda_2 = 1.0$ in the ranking loss L_{rank} . Besides, we set $\lambda_{attr} = 1.0$ for multi-label attribute cross-entropy loss L_{subj}^{attr} .

A.2. Computational Efficiency

During training, the full model of MAttNet converges at around 30,000 iterations, which takes around half day using single Titan-X(Pascal). At inference time, our fully automatic system goes through both Mask R-CNN and MAttNet, which takes on average 0.33 seconds for a forward, where 0.31 seconds are spent on Mask R-CNN and 0.02 seconds on MAttNet.

A.3. Attribute Prediction

Our full model is also able to predict attributes during testing. Our attribute labels are extracted using the template parser [13]. We fetch the object name, color and generic attribute words from each expression, with low-frequency words removed. We use 50 most frequently used attribute words for training. The histograms for top-20 attribute words are shown in Fig. 9, and the quantitative analysis of our multi-attribute prediction results is shown in Table. 5.

A.4. MAttNet + Grabcut

In Section 4.3, we show MAttNet could be extended to referential segmentation by using Mask R-CNN as the

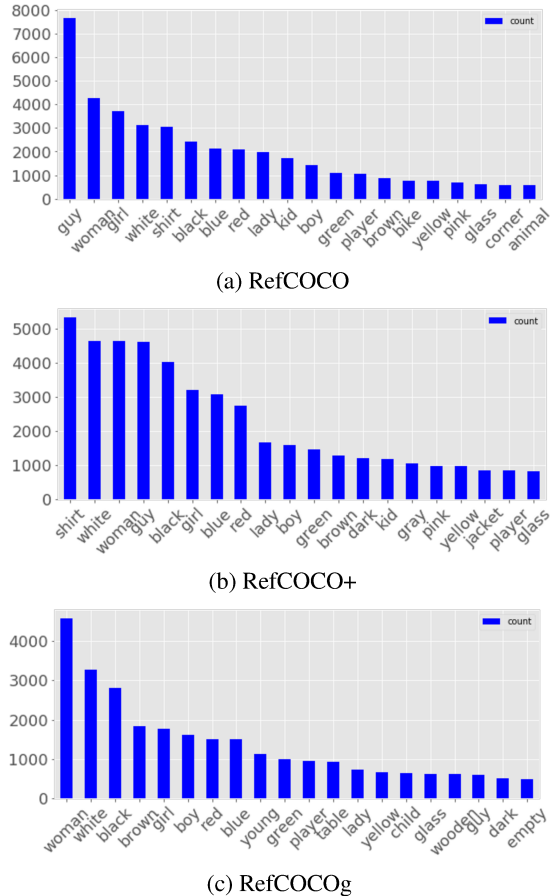


Figure 9: Attribute histogram for three datasets.

	Split	Precision	Recall	F1
RefCOCO	val	63.48	29.91	40.66
RefCOCO+	val	61.78	20.11	30.14
RefCOCOg	val	68.18	34.79	46.07

Table 5: Multi-attribute prediction on the validation split of each dataset.

backbone net. Actually, the mask branch of MAttNet could be any foreground-background decomposition method. The simplest replacement might be GrabCut. We show the results of MatNet+GrabCut in Table 6. Note even though GrabCut is an inferior segmentation method, it still far outperforms previous state-of-the-art results [15]. Thus, we believe the way of decoupling box localization (comprehension) and segmentation is more suitable for instance-level referential segmentation task.

A.5. Mask R-CNN Implementation

Our implementation of Mask R-CNN⁶ is based on the single-GPU Faster R-CNN implementation [5]. For the mask branch, we follow the structure in the original pa-

⁶Our implementation: <https://github.com/lichengunc/mask-faster-rcnn>.

RefCOCO							
Model	Split	Pr@0.5	Pr@0.6	Pr@0.7	Pr@0.8	Pr@0.9	IoU
D+RMI+DCRF [15]	val	42.99	33.24	22.75	12.11	2.23	45.18
MAttNet+GrabCut	val	51.25	41.89	29.77	17.13	5.38	42.86
D+RMI+DCRF [15]	testA	42.99	33.59	23.69	12.94	2.44	45.69
MAttNet+GrabCut	testA	52.94	42.60	27.68	13.29	2.92	44.37
D+RMI+DCRF [15]	testB	44.99	32.21	22.69	11.84	2.65	45.57
MAttNet+GrabCut	testB	47.18	38.27	29.97	20.35	7.85	40.71

RefCOCO+							
Model	Split	Pr@0.5	Pr@0.6	Pr@0.7	Pr@0.8	Pr@0.9	IoU
D+RMI+DCRF [15]	val	20.52	14.02	8.46	3.77	0.62	29.86
MAttNet+GrabCut	val	45.24	37.09	26.51	14.95	4.34	37.18
D+RMI+DCRF [15]	testA	21.22	14.43	8.99	3.91	0.49	30.48
MAttNet+GrabCut	testA	47.10	37.86	24.66	11.67	2.27	38.32
D+RMI+DCRF [15]	testB	20.78	14.56	8.80	4.58	0.80	29.50
MAttNet+GrabCut	testB	38.52	31.13	24.44	16.71	6.20	33.30

Table 6: Comparison of referential segmentation performance between D+RMI+DCRF [15] and MatNet+GrabCut.

per [6], with several differences: 1) We sample $R = 256$ regions from $N = 1$ image during each forward-backward propagation due to the constraint of single GPU, while [6] samples $R = 128$ regions from $N = 16$ images using 8 GPUs. 2) During training, the shorter edge of our resized image is 600 pixels instead of 800 pixels, for saving memory. 3) Our model is trained on a union of COCO’s 80k train and 35k subset of val (trainval35k) images excluding the val/test (valtest4k) images in RefCOCO, RefCOCO+ and RefCOCOg.

We firstly show the comparison between Faster R-CNN and Mask R-CNN on object detection in Table. 7. Both models are based on ResNet101 and were trained using same setting. In the main paper, we denote them as res101-frcn and res101-mrcn respectively. It shows that Mask R-CNN has higher AP than Faster R-CNN due to the multi-task training (with additional mask supervision).

net	AP^{bb}	AP_{50}^{bb}	AP_{75}^{bb}
res101-frcn	34.1	53.7	36.8
res101-mrcn	35.8	55.3	38.6

Table 7: Object detection results.

net	AP	AP_{50}	AP_{75}
res101-mrcn (ours)	30.7	52.3	32.4
res101-mrcn [6]	32.7	54.2	34.0

Table 8: Instance segmentation results.

We then compare our Mask R-CNN implementation with the original one [6] in Table 8. Note this is not a strictly fair comparison as our model was trained with fewer images. Overall, the AP of our implementation is ~ 2 points

lower. The main reason may due to the shorter 600-pixel edge setting and smaller training batch size. Even though, our pixel-wise comprehension results already outperform the state-of-the-art ones with a huge margin (see Table. 4, and we believe there exists space for further improvements.

A.6. More Examples

We show more examples of comprehension using our full model in Fig. 10 (RefCOCO), Fig. 11 (RefCOCO+) and Fig. 12 (RefCOCOg). For each example, we show the input image (1st column), the input expression with our predicted module weights and word attention (2nd column), the subject attention (3rd column) and top-5 attributes (4th column), box-level comprehension (5th column), and pixel-wise segmentation (6th column). As comparison, we also show some incorrect comprehension in Fig. 13.

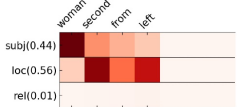
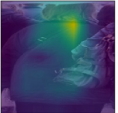
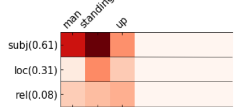
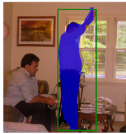
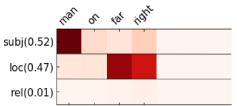
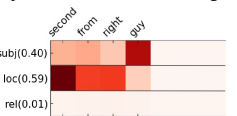
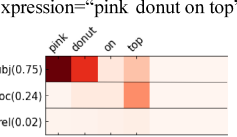
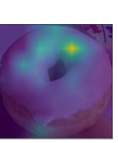
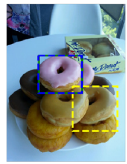
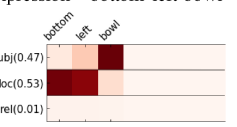
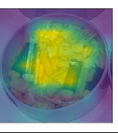
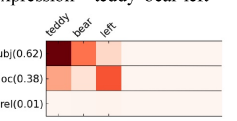
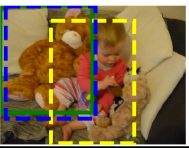
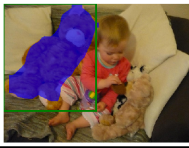
Input image	Expression & Lang. attention	Subj. attention	Top-5 attributes	Box localization	Segmentation
	Expression="woman second from left" 		1. guy (0.91) 2. black (0.20) 3. white (0.04) 4. woman (0.04) 5. lady (0.02)		
	Expression="man standing up" 		1. guy (0.44) 2. woman (0.20) 3. gray (0.19) 4. white (0.16) 5. shirt (0.16)		
	Expression="man on far right" 		1. guy (0.58) 2. jacket (0.49) 3. woman (0.24) 4. black (0.24) 5. lady (0.12)		
	Expression="second from right guy" 		1. guy (0.44) 2. blue (0.29) 3. woman (0.25) 4. girl (0.12) 5. shirt (0.11)		
	Expression="pink donut on top" 		1. pink (0.64) 2. purple (0.15) 3. white (0.13) 4. baby (0.04) 5. brown (0.04)		
	Expression="bottom left bowl" 		1. food (0.80) 2. plate (0.42) 3. white (0.12) 4. brown (0.01) 5. black (0.01)		
	Expression="teddy bear left" 		1. food (0.80) 2. plate (0.42) 3. white (0.12) 4. brown (0.01) 5. black (0.01)		

Figure 10: Examples of fully automatic comprehension on **RefCOCO**. The 1st column shows the input image. The 2nd column shows the expression, word attention and module weights. The 3rd column shows our predicted subject attention, and the 4th column shows its top-5 attributes. The 5th column shows box-level comprehension where the red dotted boxes show our prediction and yellow dotted boxes shows the relative object, and the green boxes are the ground-truth. The 6th column shows the segmentation.

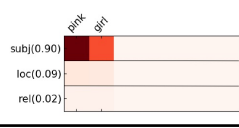
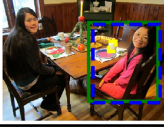
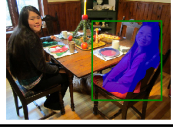
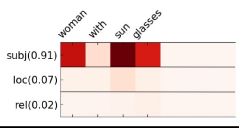
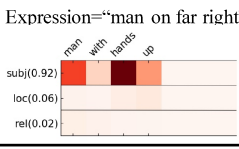
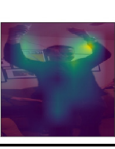
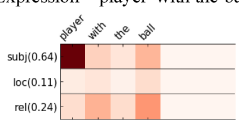
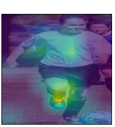
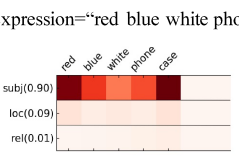
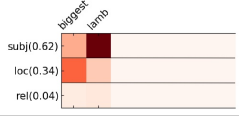
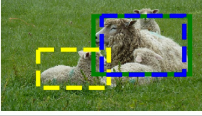
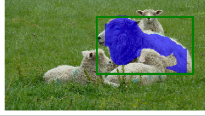
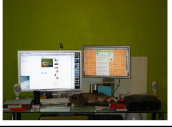
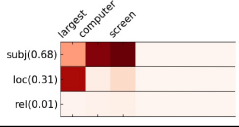
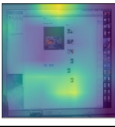
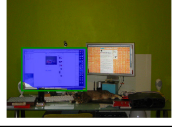
Input image	Expression & Lang. attention	Subj. attention	Top-5 attributes	Box localization	Segmentation
	Expression="pink girl" 		1. girl (0.33) 2. pink (0.26) 3. shirt (0.24) 4. white (0.12) 5. guy (0.11)		
	Expression="woman with sun glasses" 		1. woman (0.44) 2. shirt (0.27) 3. girl (0.25) 4. lady (0.19) 5. guy (0.10)		
	Expression="man on far right" 		1. black (0.58) 2. guy (0.56) 3. shirt (0.43) 4. hand (0.25) 5. woman (0.18)		
	Expression="player with the ball" 		1. blue (0.37) 2. white (0.29) 3. guy (0.22) 4. boy (0.19) 5. black (0.16)		
	Expression="red blue white phone case" 		1. red (0.16) 2. black (0.11) 3. white (0.11) 4. girl (0.04) 5. blue (0.02)		
	Expression="biggest lamb" 		1. animal (0.32) 2. face (0.26) 3. white (0.20) 4. dark (0.11) 5. black (0.10)		
	Expression="largest computer screen" 		1. white (0.94) 2. gray (0.13) 3. number (0.04) 4. old (0.04) 5. black (0.03)		

Figure 11: Examples of fully automatic comprehension on **RefCOCO+**. The 1st column shows the input image. The 2nd column shows the expression, word attention and module weights. The 3rd column shows our predicted subject attention, and the 4th column shows its top-5 attributes. The 5th column shows box-level comprehension where the red dotted boxes show our prediction and yellow dotted boxes shows the relative object, and the green boxes are the ground-truth. The 6th column shows the segmentation.

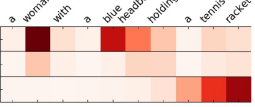
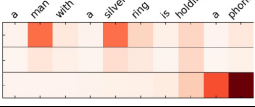
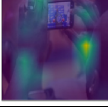
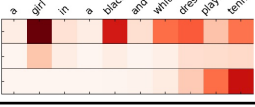
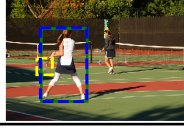
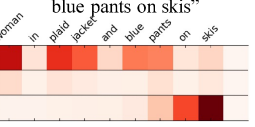
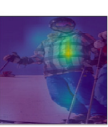
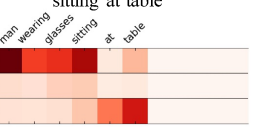
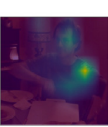
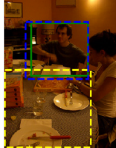
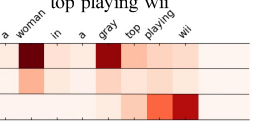
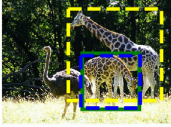
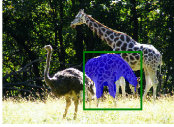
Input image	Expression & Lang. attention	Subj. attention	Top-5 attributes	Box localization	Segmentation
	Expression="a woman with a blue headband holding a tennis racket"  subj(0.50) loc(0.14) rel(0.36)		1. woman (0.45) 2. player (0.23) 3. boy (0.09) 4. young (0.07) 5. gray (0.05)		
	Expression="a man with a silver ring is holding a phone"  subj(0.46) loc(0.16) rel(0.38)		1. guy (0.39) 2. black (0.24) 3. hand (0.14) 4. woman (0.07) 5. grey (0.05)		
	Expression="a girl in a black and white dress playing tennis"  subj(0.61) loc(0.13) rel(0.26)		1. woman (0.67) 2. player (0.57) 3. girl (0.25) 4. red (0.11) 5. blonde (0.04)		
	Expression="woman in plaid jacket and blue pants on skis"  subj(0.63) loc(0.12) rel(0.25)		1. woman (0.45) 2. skier (0.40) 3. blue (0.16) 4. girl (0.12) 5. child (0.08)		
	Expression="man wearing glasses sitting at table"  subj(0.64) loc(0.14) rel(0.22)		1. guy (0.66) 2. woman (0.15) 3. boy (0.09) 4. young (0.08) 5. lady (0.08)		
	Expression="a woman in a gray top playing wii"  subj(0.58) loc(0.19) rel(0.23)		1. woman (0.80) 2. girl (0.17) 3. lady (0.13) 4. young (0.10) 5. boy (0.07)		
	Expression="giraffe bending down"  subj(0.72) loc(0.24) rel(0.04)		1. baby (0.40) 2. young (0.26) 3. brown (0.25) 4. white (0.05) 5. adult (0.02)		

Figure 12: Examples of fully automatic comprehension on **RefCOCOg**. The 1st column shows the input image. The 2nd column shows the expression, word attention and module weights. The 3rd column shows our predicted subject attention, and the 4th column shows its top-5 attributes. The 5th column shows box-level comprehension where the red dotted boxes show our prediction and yellow dotted boxes shows the relative object, and the green boxes are the ground-truth. The 6th column shows the segmentation.

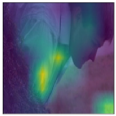
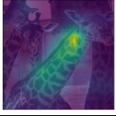
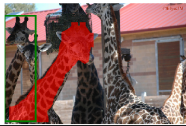
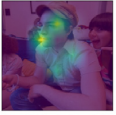
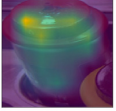
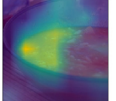
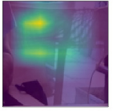
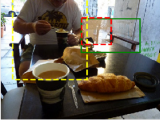
	Input image	Expression & Lang. attention	Subj. attention	Top-5 attributes	Box localization	Segmentation
RefCOCO:		Expression="man" subj(0.76) loc(0.19) rel(0.05)		1. guy (0.40) 2. shirt (0.35) 3. white (0.15) 4. blue (0.05) 5. green (0.04)		
		Expression="giraffe to far left" subj(0.59) loc(0.41) rel(0.00)		1. animal (0.50) 2. face (0.08) 3. pink (0.03) 4. middle (0.03) 5. baby (0.02)		
RefCOCO+:		Expression="pointing and smiling" subj(0.66) loc(0.20) rel(0.15)		1. shirt (0.33) 2. woman (0.29) 3. girl (0.18) 4. white (0.16) 5. guy (0.15)		
		Expression="red cover mustard" subj(0.71) loc(0.21) rel(0.08)		1. red (0.81) 2. full (0.03) 3. pink (0.02) 4. glass (0.01) 5. dark (0.01)		
RefCOCOg:		Expression="the empty part of the blue plate on the left" subj(0.75) loc(0.24) rel(0.01)		1. blue (0.24) 2. plate (0.08) 3. glass (0.06) 4. full (0.05) 5. red (0.05)		
		Expression="a white chair behind a man" subj(0.63) loc(0.14) rel(0.24)		1. wooden (0.87) 2. brown (0.33) 3. table (0.20) 4. empty (0.17) 5. white (0.07)		

Figure 13: Examples of incorrect comprehension on three datasets. The 1st column shows the input image. The 2nd column shows the expression, word attention and module weights. The 3rd column shows our predicted subject attention, and the 4th column shows its top-5 attributes. The 5th column shows box-level comprehension where the red dotted boxes show our prediction and yellow dotted boxes shows the relative object, and the green boxes are the ground-truth. The 6th column shows the segmentation.

References

- [1] J. Andreas, D. Klein, and S. Levine. Modular multitask reinforcement learning with policy sketches. *ICML*, 2017. 2
- [2] J. Andreas, M. Rohrbach, T. Darrell, and D. Klein. Learning to compose neural networks for question answering. *NAACL*, 2016. 2, 3
- [3] J. Andreas, M. Rohrbach, T. Darrell, and D. Klein. Neural module networks. In *CVPR*, 2016. 2, 3
- [4] K. Chen, R. Kovvuri, and R. Nevatia. Query-guided regression network with context policy for phrase grounding. In *ICCV*, 2017. 2
- [5] X. Chen and A. Gupta. An implementation of faster rcnn with study for region sampling. *arXiv preprint arXiv:1702.02138*, 2017. 9
- [6] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *ICCV*, 2017. 7, 8, 9, 10
- [7] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 3
- [8] R. Hu, J. Andreas, M. Rohrbach, T. Darrell, and K. Saenko. Learning to reason: End-to-end module networks for visual question answering. *ICCV*, 2017. 2
- [9] R. Hu, M. Rohrbach, and T. Darrell. Segmentation from natural language expressions. In *ECCV*, 2016. 2, 7, 8
- [10] R. Hu, M. Rohrbach, J. Andreas, T. Darrell, and K. Saenko. Modeling relationship in referential expressions with compositional modular networks. In *CVPR*, 2017. 2, 3, 5, 6
- [11] R. Hu, H. Xu, M. Rohrbach, J. Feng, K. Saenko, and T. Darrell. Natural language object retrieval. *CVPR*, 2016. 1, 2
- [12] J. Johnson, B. Hariharan, L. van der Maaten, J. Hoffman, L. Fei-Fei, C. L. Zitnick, and R. Girshick. Inferring and executing programs for visual reasoning. *ICCV*, 2017. 2
- [13] S. Kazemzadeh, V. Ordonez, M. Matten, and T. L. Berg. Referitgame: Referring to objects in photographs of natural scenes. In *EMNLP*, 2014. 2, 3, 4, 5, 6, 9
- [14] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 5
- [15] C. Liu, Z. Lin, X. Shen, J. Yang, X. Lu, and A. Yuille. Recurrent multimodal interaction for referring image segmentation. In *ICCV*, 2017. 2, 7, 8, 9, 10
- [16] J. Liu, L. Wang, and M.-H. Yang. Referring expression generation and comprehension via attributes. In *ICCV*, 2017. 2, 4
- [17] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 2, 8
- [18] R. Luo and G. Shakhnarovich. Comprehension-guided referring expressions. *CVPR*, 2017. 1, 2, 6
- [19] J. Mao, J. Huang, A. Toshev, O. Camburu, A. Yuille, and K. Murphy. Generation and comprehension of unambiguous object descriptions. *CVPR*, 2016. 1, 2, 3, 5, 6
- [20] V. K. Nagaraja, V. I. Morariu, and L. S. Davis. Modeling context between objects for referring expression understanding. In *ECCV*, 2016. 1, 2, 3, 5, 6
- [21] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, 2015. 3
- [22] A. Rohrbach, M. Rohrbach, R. Hu, T. Darrell, and B. Schiele. Grounding of textual phrases in images by reconstruction. In *ECCV*, 2016. 1, 2
- [23] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 3
- [24] R. Socher, J. Bauer, C. D. Manning, et al. Parsing with compositional vector grammars. In *ACL*, 2013. 2, 3
- [25] J.-C. Su, C. Wu, H. Jiang, and S. Maji. Reasoning about fine-grained attribute phrases using reference games. *ICCV*, 2017. 4
- [26] L. Wang, Y. Li, and S. Lazebnik. Learning deep structure-preserving image-text embeddings. *CVPR*, 2016. 1, 2
- [27] L. Wang, Y. Li, and S. Lazebnik. Learning two-branch neural networks for image-text matching tasks. *arXiv preprint arXiv:1704.03470*, 2017. 1
- [28] Q. Wu, C. Shen, P. Wang, A. Dick, and A. van den Hengel. Image captioning and visual question answering based on attributes and external knowledge. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017. 4
- [29] Z. Yang, D. Yang, C. Dyer, X. He, A. J. Smola, and E. H. Hovy. Hierarchical attention networks for document classification. In *HLT-NAACL*, 2016. 3
- [30] T. Yao, Y. Pan, Y. Li, Z. Qiu, and T. Mei. Boosting image captioning with attributes. *arXiv preprint arXiv:1611.01646*, 2016. 4
- [31] Q. You, H. Jin, Z. Wang, C. Fang, and J. Luo. Image captioning with semantic attention. In *CVPR*, 2016. 4
- [32] L. Yu, P. Poirson, S. Yang, A. C. Berg, and T. L. Berg. Modeling context in referring expressions. In *ECCV*, 2016. 1, 2, 3, 5, 6
- [33] L. Yu, H. Tan, M. Bansal, and T. L. Berg. A joint speaker-listener-reinforcer model for referring expressions. In *CVPR*, 2017. 2, 3, 5, 6, 7