Comprehension-guided referring expressions

Ruotian Luo
TTI-Chicago
rluo@ttic.edu

Gregory Shakhnarovich
TTI-Chicago
greg@ttic.edu

Abstract

We consider generation and comprehension of natural language referring expression for objects in an image. Unlike generic "image captioning" which lacks natural standard evaluation criteria, quality of a referring expression may be measured by the receiver's ability to correctly infer which object is being described. Following this intuition, we propose two approaches to utilize models trained for comprehension task to generate better expressions. First, we use a comprehension module trained on human-generated expressions, as a "critic" of referring expression generator. The comprehension module serves as a differentiable proxy of human evaluation, providing training signal to the generation module. Second, we use the comprehension module in a generate-and-rerank pipeline, which chooses from candidate expressions generated by a model according to their performance on the comprehension task. We show that both approaches lead to improved referring expression generation on multiple benchmark datasets.

1. Introduction

Image captioning, defined broadly as automatic generation of text describing images, has seen much recent attention. Deep learning, and in particular recurrent neural networks (RNNs), have led to a significant improvement in state of the art. However, the metrics currently used to evaluate image captioning are mostly borrowed from machine translation. This misses the naturally multi-modal distribution of appropriate captions for many scenes.

Referring expressions are a special case of image captions. Such expressions describe an object or region in the image, with the goal of identifying it uniquely to a listener. Thus, in contrast to generic captioning, referring expression generation has a natural evaluation metric: a human should easily comprehend the description and identify the object(s) being described.

In this paper, we consider two related tasks. One is the generation task: generating a discriminative referring expression for an object in an image. The other is the *comprehension* task (called natural language object retrieval in [15]): localizing an object in an image given a referring expression. Most prior work addressed both tasks by building a sequence generation model. Such a model can be used discriminatively for the comprehension task, by inferring the region which maximizes the expression posterior.

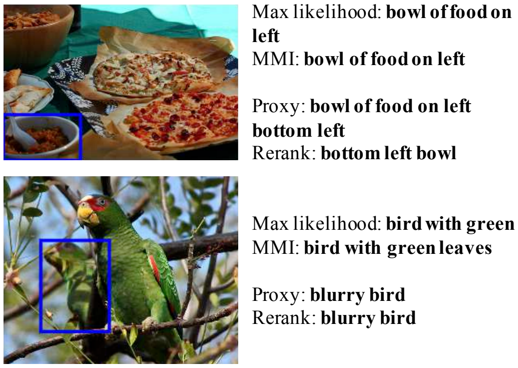
Max likelihood: **bowl of food on left**
MMI: **bowl of food on left**
Proxy: **bowl of food on left bottom left**
Rerank: **bottom left bowl**
Max likelihood: **bird with green**
MMI: **bird with green leaves**
Proxy: **blurry bird**
Rerank: **blurry bird**

Figure 1: For each image, the top two expressions are generated by baseline models proposed in [23]; the bottom two expressions are generated by our methods.

In contrast, we draw inspiration from the generator-discriminator structure in Generative Adversarial Networks[10, 26]. In GANs, the generator module tries to generate a signal (e.g., natural image), and the discriminator module tries to tell real images apart from the generated ones. For our task, the generator produces referring expressions. We would like these expressions to be both intelligible/fluent and unambiguous to human. Fluency can be encouraged by using the standard cross entropy loss with respect to human-generated expressions). On the other hand, we adopt a comprehension model as the "discriminator" which tells if the expression can be correctly dereferenced. Note that we can also regard the comprehension model as a "critic" of the "action" made by the generator where the "action" is each generated word.

Instead of an adversarial relationship between the two modules in GANs, our architecture is collaborative – the comprehension module "tells" the generator how to improve the expressions it produces. Our methods are much simpler than GANs as it avoids the alternating optimization strategy – the comprehension model is separately trained on ground truth data and then fixed. We adapt the comprehension model so it becomes differentiable with respect to its expression input. Thus we turn it into a proxy for human understanding providind an additional training signal for the generator. This (first, to our knowledge) attempt to integrate automatic referring expression generation with a discriminative comprehension model in a collaborative framework is our main controbution.

Specifically there are two ways that we utilize the comprehension model. The **generate-and-rerank** method uses comprehension on the fly, similarly to [1], where they tried to produce unambiguous captions for clip-art images. The generation model generates some candidate expressions and passes them through the comprehension model. The final output expression is the one with highest generation-comprehension score which we will describe later.

The **training by proxy** method is closer in spirit to GANs. The generation and comprehension model are connected and the generation model is optimized to lower discriminative comprehension loss (in addition to the cross-entropy loss). We investigate several training strategies for this method and a trick to make proxy model trainable by standard back-propagation. Compared to generate-and-rerank method, the training by proxy method doesn't require additional region proposals during test time.

2. Related work

The main approach in modern image captioning literature [32, 17, 22] is to encode an image using a convolutional neural network (CNN), and feed this as input to an RNN, able to generate a arbitrary-length sequence of words.

While captioning typically aims to describe an entire image, some work takes regions into consideration, by incorporating them in an attention mechanism [35, 21], alignment of words/phrases within sentences to regions [17], or by defining "dense" captioning on a per-region basis [16]. The latter includes a dataset of captions collected without requirement to be unambiguous, so they cannot be regarded as referring expression.

Text-based image retrieval has been considered as a task relying on image captioning [32, 17, 22, 35]. However, it can also be regarded as a multi-modal embedding task. In previous works [7, 33, 34] such embeddings have been trained separately for visual and textual input, with the objective to minimize matching loss, e.g., hinge loss on cosine distance, or to enforce partial order on captions and images [31]. [28] tried different text embedding networks for fine-grained image retrieval.

Closer to the focus of this paper, referring expressions

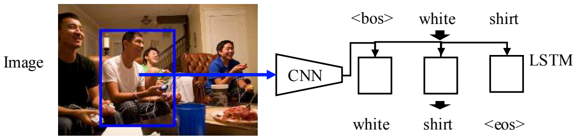

Figure 2: Illustration of how the generation model describes region inside the blue bounding box. <bos> and <eos> stand for beginning and end of sentence.

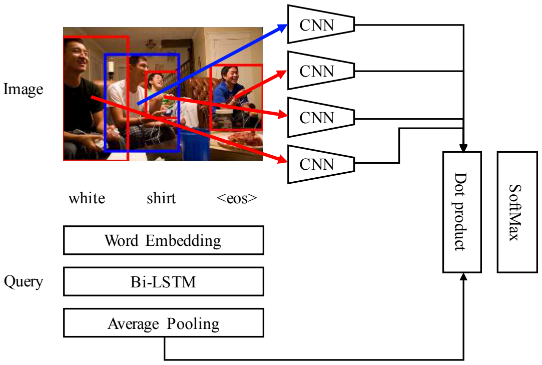

Figure 3: Illustration of comprehension model using softmax loss. The blue bounding box is the target region, and the red ones are incorrect regions. The CNNs share the weights.

have attracted interest after the release of the standard datasets [18, 36, 23]. In [15] a caption generation model is appropriated for a generation task, by evaluating the probability of a sentence given an image $P(S|I)$ as the matching score. Concurrently, [23] at the same time proposed a joint model, in which comprehension and generation aspects are trained using max-margin Maximum Mutual Information (MMI) training. Both papers used whole image, region and location/size features. Based on the model in [23], both [25] and [37] try to model context regions in their frameworks.

Our method is trying to combine simple models and replace the max margin loss, which is orthogonal to modeling context, with a surrogate closer to the eventual goal – human comprehension. This requires a comprehension model, which, given a referring expression, infers the appropriate region in the image.

Among comprehension models proposed in literature, [29] uses multi-modal embedding and sets up the comprehension task as a multi-class classification. Later, [8] achieves a slight improvement by replacing the concatenation layer with a compact bilinear pooling layer. The comprehension model used in this paper belongs to this multi-modal embedding category.

The "speaker-listener" model in [1] attempts to produce discriminative captions that can tell images apart. The speaker is trained to generate captions, and a listener to prefer the correct image over a wrong one, given the caption. At test time, the listener reranks the captions sampled from the speaker. Our generate-and-rerank method is based on translating this idea to referring expression generation.

3. Generation and comprehension models

We start by defining the two modules used in the collaborative architecture we propose. Each of these can be trained as a standalone machine for the task it solves, given a data set with ground truth regions/referring expressions.

3.1. Expression generation model

We use a simple expression generation model introduced in [36, 23]. The generation task takes inputs of an image I and an internal region r, and outputs an expression w, $G : I \times r \rightarrow w$. We consider a model $P_G(w|I, r)$, under which

$$G(I, r) = \underset{w}{\operatorname{argmax}}\, P_G(w|I, r) \tag{1}$$

Given a set of (image, region, expression) tuples, $\{(I_i, w_i, r_i)\}$, we train P_G model by maximizing the likelihood

$$P_G^* = \underset{P_G}{\operatorname{argmax}} \sum_i \log P_G(w_i|I_i, r_i) \tag{2}$$

Specifically, the generation model is an encoder-decoder network. First we need to encode the visual information from r_i and I_i. As in [15, 36, 25], we use encoding that includes: target object representation o_i, global context feature g_i and location/size feature l_i. In our experiments, o_i is the activation on the cropped region r_i of the last fully connected layer `fc7` of VGG-16 [30]; g_i is the `fc7` activation on the whole image I_i; l_i is a 5D vector encoding the opposite corners of the bounding box of r_i, as well as the bounding box size relative to the image size. The final visual feature vector v_i of the region is an linear transformation (plus bias terms) of the concatenation of three features $[o_i, g_i, l_i]$.

Figure 2 shows the structure of the generation model. To generate a sequence we use a uni-directional LSTM decoder[14]. Inputs of LSTM at each time step include the visual features and the previous word embedding. The output of the LSTM at a time step is the distribution of predicted next word. The model is trained to minimize cross entropy loss, equivalent to maximizing the likelihood,

$$L_{gen} = \sum_i \sum_{t=1}^{T_i} \log P_G(w_{i,t}|w_{i,<t}, I_i, r_i), \tag{3}$$

$$P_G^* = \underset{P_G}{\operatorname{argmin}}\, L_{gen}, \tag{4}$$

where , $w_{i,t}$ is the t-th word of ground truth expression w_i, and T_i is the length of w_i. In practice, instead of precisely inferring the $\operatorname{argmax}_w P_G(w|I, r)$, one uses beam search, greedy search or sampling to get the output.

3.2. Comprehension

The comprehension task is to select a region (bounding box) $\hat{r}$ from a set of regions $\mathcal{R} = \{r_i\}$ given a query expression q and the image I.

$$C : I \times q \times \mathcal{R} \rightarrow r,\ r \in \mathcal{R} \tag{5}$$

We also define the comprehension model as a posterior distribution $P_C(r|I, q, \mathcal{R})$. The estimated region given a comprehension model is: $\hat{r} = \operatorname{argmax}_r P_C(r|I, q, \mathcal{R})$.

In general, our comprehension model is very similar to [29]. To build the model, we first define a similarity function f_{sim}. We use the same visual feature encoder structure as in generation model. For the query expression, we use a one-layer bi-directional LSTM [12] to encode it. We take the averaging over the hidden vectors of each timestep so that we can get a fixed-length representation for an arbitrary length of query.

$$h = f_{LSTM}(\mathbf{EQ}), \tag{6}$$

where $\mathbf{E}$ is the word embedding matrix initialized from pretrained word2vec[24] and $\mathbf{Q}$ is a one-hot representation of the query expression, i.e. $\mathbf{Q}_{i,j} = \mathbf{1}(q_i = j)$.

Unlike [29], which uses concatenation + MLP to calculate the similarity, we use a simple dot product as in [4].

$$f_{sim}(I, r_i, q) = v_i^T h. \tag{7}$$

We consider two formulations of the comprehension task as classification. The per-region logistic loss

$$P_C(r_i|I, q) = \sigma(f_{sim}(I, r_i, q)), \tag{8}$$

$$L_{bin} = -\log P_C(r_{i^*}|I, q) - \sum_{i \neq i^*} \log(1 - P_C(r_i|I, q)), \tag{9}$$

where r_{i^*} is ground truth region, corresponds to a per-region classification: is this region the right match for the expression or not. The softmax loss

$$P_C(r_i|I, q, \mathcal{R}) = \frac{e^{s_i}}{\sum_i e^{s_i}}, \tag{10}$$

$$L_{multi} = -\log P_C(r_{i^*}|I, q, \mathcal{R}), \tag{11}$$

where $s_i = f_{sim}(I, r_i, q)$, frames the task as a multi-class classification: which region in the set should be matched to the expression.

The model is trained to minimize the comprehension loss. $P_C^* = \operatorname{argmin}_{P_C} L_{com}$, where L_{com} is either L_{bin} or L_{multi}.

Figure 3 shows the structure of our generation model under multi-class classification formulation.

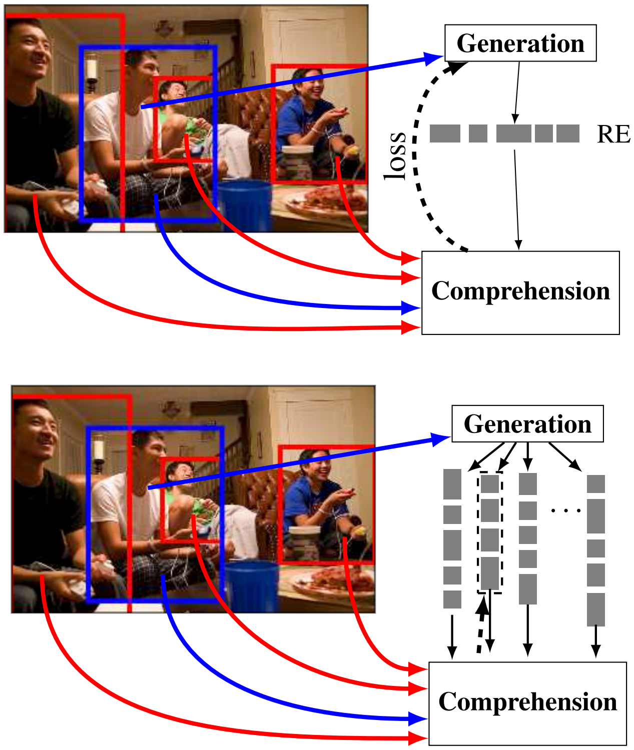

Figure 4: Top: Training by proxy. The comprehension model must correctly identify the target (blue) region based on a RE; identification loss (dashed) is propagated to the generator. Bottom: Generate and rerank. Generator produces multiple REs; comprehension model evaluates them based on its ability to identify the true (blue) region from them, and selects (dashed) the best RE.

4. Comprehension-guided generation

Once we have trained the comprehension model, we can start using it as a proxy for human comprehension, to guide expression generator.

4.1. Training by proxy

Consider a referring expression generated by G for a given training example of an image/region pair (I, r). The generation loss L_{gen} will inform the generator how to modify its model to maximize the probability of the ground truth expression w. The comprehension model C can provide an alternative, complementary signal: how to modify G to maximize the discriminativity of the generated expression, so that C selects the correct region r among the proposal set $\mathcal{R}$. Intuitively, this signal should push down on probability of a word if it's unhelpful for comprehension, and pull that probability up if it is helpful.

Ideally, we hope to minimize the comprehension loss of the output of the generation model $L_{com}(r|I, \mathcal{R}, \tilde{\mathbf{Q}})$, where $\tilde{\mathbf{Q}}$ is the 1-hot encoding of $\tilde{q} = G(I, r)$, with K rows (vocabulary size) and T columns (sequence length).

We hope to update the generation model according to the gradient of loss with respect to the model parameter θ_G. By chain rule,

$$\frac{\partial L_{com}}{\partial \theta_G} = \frac{\partial L_{com}}{\partial \tilde{\mathbf{Q}}} \frac{\partial \tilde{\mathbf{Q}}}{\partial \theta_G} \tag{12}$$

However, $\tilde{\mathbf{Q}}$ is inferred by some algorithm which is not differentiable. To address this issue, [27, 2, 37] applied reinforcement learning methods. However, here we use an approximate method borrowing from the idea of soft attention mechanism [35, 5].

We define a matrix $\mathbf{P}$ which has the same size as $\tilde{\mathbf{Q}}$. The i-th column of $\mathbf{P}$ is – instead of the one-hot vector of the generated word i – the distribution of the i-th word produced by P_G, i.e.

$$\mathbf{P}_{i,j} = P_G(w_i = j). \tag{13}$$

$\mathbf{P}$ has several good properties. First, $\mathbf{P}$ has the same size as $\tilde{\mathbf{Q}}$, so that the we can still compute the query feature by replacing the $\tilde{\mathbf{Q}}$ by $\mathbf{P}$, i.e. $h = f_{LSTM}(\mathbf{EP})$. Secondly, the sum of each column in $\mathbf{P}$ is 1, just like $\tilde{\mathbf{Q}}$. Thirdly, $\mathbf{P}$ is differentiable with respect to generator's parameters.

Now, the gradient of θ_G is calculated by:

$$\frac{\partial L_{com}}{\partial \theta_G} = \frac{\partial L_{com}}{\partial \mathbf{P}} \frac{\partial \mathbf{P}}{\partial \theta_G} \tag{14}$$

We will use this approximate gradient in the following three methods.

4.1.1 Compound loss

Here we introduce how we integrate the comprehension model to guide the training of the generation model.

The cross-entropy loss (3) encourages fluency of the generated expression, but disregards its discriminativity. We address this by using the comprehension model as a source of an additional loss signal. Technically, we define a compound loss

$$L = L_{gen} + \lambda L_{com} \tag{15}$$

where the comprehension loss L_{com} is either the logistic (8) or the softmax (10) loss; the balance term λ determines the relative importance of fluency vs. discriminativity in L.

Both L_{gen} and L_{com} take as input G's distribution over the i-th word $P_G(w_i|I, r, w_{<i})$, where the preceding words $w_{<i}$ are from the ground truth expression.

Replacing $\tilde{\mathbf{Q}}$ with $\mathbf{P}$ (Sec. 4.1) allows us to train the model by back-propogation from the compound loss (15).

4.1.2 Modified Scheduled sampling training

Our final goal is to generate comprehensible expression during test time. However, in compound loss, the loss is calculated given the ground truth input while during test time each token is generated by the model, thus yielding a discrepancy between how the model is used during training and at test time. Inspired by similar motivation, [3] proposed scheduled sampling which allows the model to be trained with a mixture of ground truth data and predicted data. Here, we propose this modified schedule sampling training to train our model.

During training, at each iteration i, we draw a random variable α from a Bernoulli distribution with probability ϵ_i. If $\alpha = 1$, we feed the ground truth expression to LSTM frames, and minimize cross entropy loss. If $\alpha = 0$, we sample the whole sequence step by step according to the posterior, and the input of comprehension model is $P_G(w_i|I, r, \hat{w}_{<i})$, where $\hat{w}_{<i}$ are the *sampled* words. We update the model by minimizing the comprehension loss. Therefore, α serves as a dispatch mechanism, randomly alternating between the sources of data for the LSTMs and the components of the compound loss.

We start the modified scheduled sampling training from a pretrained generation model trained on cross entropy loss using the ground truth sequences. As the training progresses, we linearly decay ϵ_i until a preset minimum value ϵ. The minimum probability prevents the model from degeneration. If we don't set the minimum, when ϵ_i goes to 0, the model will lose all the ground truth information, and will be purely guided by the comprehension model. This would lead the generation model to discover those pathological optimas that exist in neural classification models[11]. In this case, the generated expressions would do "well" on comprehension model, but no longer be intelligible to human. See Algorithm 1 for the pseudo-code.

Algorithm 1 Modified scheduled sampling training

```
Train the generation model $G$.
Set the offset $k$ ($0 \leq k \leq 1$), the slope of decay $c$, minimum probability $\epsilon$, number of iterations $N$.
for $i = 1, N$ do
    $\epsilon_i \leftarrow \max(\epsilon, k - ci)$
    Get a sample from training data, $(I, r, w)$
    Sample the $\alpha$ from Bernoulli distribution, where $P(\alpha = 1) = \epsilon_i$
    if $\alpha = 1$ then
        Minimize $L_{gen}$ with the ground truth input.
    else
        Sample a sequence $\hat{w}$ from $P_G(w|I, r)$
        Minimize $L_{com}$ with the input $P_G(w_j|I, r, \hat{w}_{<j}), j \in [1, T]$
```

4.1.3 Stochastic mixed sampling

Since modified scheduled sampling training samples a whole sentence at a time, it would be hard to get useful signal if there is an error at the beginning of the inference. We hope to find a method that can slowly deviate from the original model and explore.

Here we borrow the idea from mixed incremental cross-entropy reinforce(MIXER)[27]. Again, we start the model from a pretrained generator. Then we introduce model predictions during training with an annealing schedule so as to gradually teach the model to produce stable sequences. For each iteration i, We feed the input for the first s_i steps, and sample the rest $T - s_i$ words, where $0 \leq s_i \leq T$, and T is the maximum length of expressions. We define $s_i = s + \Delta s$, where s is a base step size which gradually decreases during training, and Δs is a random variable which follows geometric distribution: $P(\Delta s = k) = (1-p)^{k+1}p$. This Δs is the difference between our method and MIXER. We call this method: Stochastic mixed incremental cross-entropy comprehension(SMIXEC).

By introducing this term Δs, we can control how much supervision we want to get from ground truth by tuning the value p. This is also for preventing the model from producing pathological optimas. Note that, when p is 0, Δs will always be large enough so that it's just cross entropy loss training. When p is 1, Δs will always equal to 0, which is equivalent to MIXER annealing schedule. See Algorithm 2 for the pseudo-code.

Algorithm 2 Stochastic mixed incremental cross-entropy comprehension (SMIXEC)

```
Train the generation model $G$.
Set the geometric distribution parameter $p$, maximum sequence length $T$, period of decay $d$, number of iterations $N$.
for $i = 1, N$ do
    $s \leftarrow \max(0, T - \lceil i/d \rceil)$
    Sample $\Delta s$ from geometric distribution with success probability $p$
    $s_i \leftarrow \min(T, s + \Delta s)$
    Get a sample from training data, $(I, r, w)$
    Run the $G$ with ground truth input in the first $s_i$ steps, and sampled input in the remaining $T - s_i$
    Get $L_{gen}$ on first $s_i$ steps, and $L_{com}$ on whole sentence but with input $\{w_{1 \ldots s_i}, P_G(w_{s_i+1 \ldots T}|I, r, w_{1 \cdots s_i}, \hat{w}_{s_i+1 \ldots T})\}$
    Minimize $L_{com} + \lambda L_{gen}$. (Not backprop through $w_{1 \ldots s_i}$)
```

4.2. Generate-and-rerank

Here we propose a different strategy to generate better expressions. Instead of using comprehension model for training a generation model, we compose the comprehension model during test time. The pipeline is similar to [1].

Unlike in Sec. 3.1, we not only need image I and region r as input, but also a region set $\mathcal{R}$. Suppose we have a generation model and a comprehension model which are trained pretrained. The steps are as follows:

1. Generate candidate expressions $\{c_1, \dots, c_n\}$ according to $P_G(\cdot|I, r)$.
2. Select c_k with $k = \text{argmax}_i\, score(c_i)$.

Here, we don't use beam search because we want the candidate set to be more diverse. And we define the score function as a weighted combination of the log perplexity and comprehension loss (we assume to use softmax loss here).

$$score(c) = \frac{1}{T}\sum_{k=1}^{T} \log p_G(c_k|r, c_{1..k-1}) \\ +\gamma \log p_C(r|I, \mathcal{R}, c), \qquad (16)$$

where c_k is the k-th token of c, T is the length of c.

This can be viewed as a weighted joint log probability that an expression to be both nature and unambiguous. The log perplexity term ensures the fluency, and the comprehension loss ensures the chosen expression to be discriminative.

5. Experiments

We base our experiments on the following data sets.

RefClef(ReferIt)[18] contains 20,000 images from IAPR TC-12 dataset[13], together with segmented image regions from SAIAPR-12 dataset[6]. The dataset is split into 10,000 for training/validation and 10,000 for test. There are 59,976 (image, bounding box, description) tuples in the trainval set and 60,105 in the test set.

RefCOCO(UNC RefExp)[36] consists of 142,209 referring expressions for 50,000 objects in 19,994 images from COCO[20], collected using the ReferitGame [18]

RefCOCO+[36] has 141,564 expressions for 49,856 objects in 19,992 images from COCO. "Location words" are disallowed, focusing the data set more on appearance based description.

RefCOCOg(Google RefExp)[23] consists of 85,474 referring expressions for 54,822 objects in 26,711 images from COCO; it contains longer and more flowery expressions than RefCOCO and RefCOCO+.

5.1. Comprehension

We first evaluate our comprehension model on human-made expressions, to assess its ability to provide useful signal. We consider two comprehension settings as in [23, 36, 25]. First, the input region set $\mathcal{R}$ contains only ground truth bounding boxes for objects, and a hit is defined by the model choosing the correct region the expression refers to. In the second setting, $\mathcal{R}$ contains proposal regions generated by FastRCNN detector[9], or by other proposal generation methods[38]. Here a hit occurs when the model chooses a proposal with intersection over union(IoU) with the ground truth of 0.5 or higher. We used precomputed proposals from [36, 23, 15] for all four datasets.

In RefCOCO and RefCOCO+, we have two test sets: testA contains people and testB contains all other objects. For RefCOCOg, we evaluate on the validation set. For RefClef, we evaluate on the test set.

We train the model using Adam optimizer [19]. The word embedding size is 300, and the hidden size of bi-LSTM is 512. The length of visual feature is 1024. For RefCOCO, RefCOCO+ and RefCOCOg, we train the model using softmax loss, with ground truth regions as training data. For RefClef dataset, we use the logistic loss. The training regions are composed of ground truth regions and all the proposals from Edge Box [38]. The binary classification is to tell if the proposal is a hit or not.

Table 1 shows our results on RefCOCO, RefCOCO+ and RefCOCOg compared to recent algorithms. Among these, MMI represents Maximum Mutual Information which uses max-margin loss to help the generation model better comprehend. With the same visual feature encoder, our model can get a better result compared to MMI in [36]. Our model is also competitive with recent, more complex state-of-the-art models [36, 25]. Table 2 shows our results on RefClef where we only test in the second setting to compare to existing results; our model, which is a modest modification of [29], obtains state of the art accuracy in this experiment.

5.2. Generation

Table 3, 4 shows our evaluation of different methods based on automatic caption generation metrics. We also add an 'Acc' column, which is the "comprehension accuracy" of the generated expressions according to our comprehension model: how well our comprehension model can comprehend the generated expressions.

The two baseline models are max likelihood(MLE) and maximum mutual information(MMI) from [36]. Our methods include compound loss(CL), modified scheduled sampling(MSS), stochastic mixed incremental cross-entropy comprehension(SMIXEC) and also generate-and-rerank(Rerank). And the MLE+sample is designed for better analyzing rerank model.

For the two baseline models and our three strategies for training by proxy method, we use greedy search to generate an expression. The MLE+sample and Rerank methods generate an expression by choosing a best one from 100

	RefCOCO				RefCOCO+				RefCOCOg	
	Test A		Test B		Test A		Test B		Val	
	GT	DET	GT	DET	GT	DET	GT	DET	GT	DET
MLE[36]	63.15%	58.32%	64.21%	48.48%	48.73%	46.86%	42.13%	34.04%	55.16%	40.75%
MMI[36]	71.72%	64.90%	71.09%	54.51%	52.44%	54.03%	47.51%	42.81%	62.14%	45.85%
visdif+MMI[36]	73.98%	67.64%	76.59%	55.16%	59.17%	55.81%	**55.62%**	43.43%	64.02%	46.86%
Neg Bag[25]	**75.6%**	58.6%	**78.0%**	**56.4%**	-	-	-	-	**68.4%**	39.5%
Ours	74.14%	**68.11%**	71.46%	54.65%	59.87%	56.61%	54.35%	**43.74%**	63.39%	47.60%
Ours(w2v)	74.04%	67.94%	73.43%	55.18%	**60.26%**	**57.05%**	55.03%	43.33%	65.36%	**49.07%**

Table 1: Comprehensions results on RefCOCO, RefCOCO+, RefCOCOg datasets. GT: the region set contains ground truth bounding boxes; DET: region set contains proposals generated from detectors. w2v means initializing the embedding layer using pretrained word2vec.

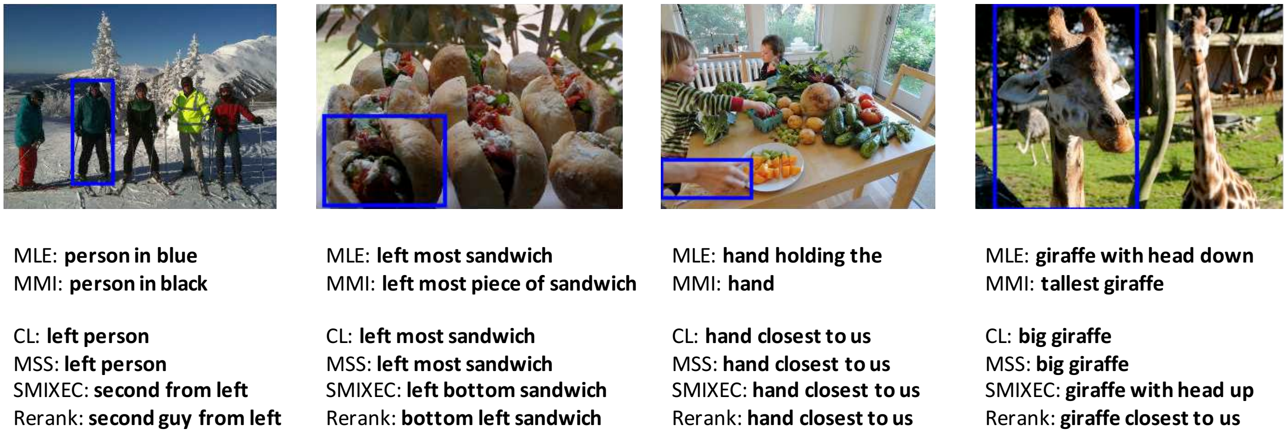

Figure 5: Generation results on (L to R)RefCOCO testA, RefCOCO testB, RefCOCO+ testA and RefCOCO+ testB.

	RefCLEF Test
SCRC[15]	17.93%
GroundR[29]	26.93%
MCB[8]	28.91%
Ours	31.25%
Ours(w2v)	**31.85%**

Table 2: Comprehension on RefClef (EdgeBox proposals)

sampled expressions.

Our generate-and-rerank (Rerank in Table 3, 4) model gets consistently better results on automatic comprehension accuracy and on fluency-based metrics like BLEU. To see if the improvement is from sampling or reranking, we also sampled 100 expressions on MLE model and choose the one with the lowest perplexity (MLE+sample in Table 3, 4). The generate-and-rerank method still has better results, showing benefit from comprehension-guided reranking.

We can see that training by proxy can get higher accuracy under the comprehension model (Acc).

Among the three training schedules of training by proxy, there is no clear winner. In RefCOCO, our SMIXEC method outperforms basic MMI method with higher comprehension accuracy and higher caption generation metrics. The compound loss and modified scheduled sampling seem to suffer from optimizing over the accuracy. However, in RefCOCO+ and RefCOCOg, our three models seem to perform very differently. The compound loss works better on RefCOCO+ TestB and RefCOCOg; the SMIXEC works best on RefCOCO+ TestA. The source of this disparity is unclear to us.

Human evaluations We also evaluated human comprehension of the generated expressions, since this is known to not be perfectly correlated with automatic metrics [36]. For 100 images randomly chosen from each split of RefCOCO and RefCOCO+, subjects had to click on the object which they thought was the best match for a generated expression. Each image/expression example was presented to two subjects, with a hit recorded only when both subjects clicked inside the correct region.

The results from human evaluations with MMI, RefCOCO

	Test A					Test B				
	Acc	BLEU 1	BLEU 2	ROUGE	METEOR	Acc	BLEU 1	BLEU 2	ROUGE	METEOR
MLE[36]	74.80%	0.477	0.290	0.413	0.173	72.81%	0.553	**0.343**	0.499	0.228
MMI[36]	78.78%	0.478	**0.295**	0.418	0.175	74.01%	0.547	0.341	0.497	0.228
CL	**80.14%**	0.4586	0.2552	0.4096	0.178	75.44%	0.5434	0.3266	**0.5056**	**0.2326**
MSS	79.94%	0.4574	0.2532	0.4126	0.1759	**75.93%**	0.5403	0.3232	0.5010	0.2297
SMIXEC	79.99%	**0.4855**	0.2800	**0.4212**	**0.1848**	75.60%	**0.5536**	0.3426	0.5012	0.2320
MLE+sample	78.38%	0.5201	**0.3391**	0.4484	0.1974	73.08%	0.5842	0.3686	0.5161	0.2425
Rerank	**97.23%**	**0.5209**	**0.3391**	**0.4582**	**0.2049**	**94.96%**	**0.5935**	**0.3763**	**0.5259**	**0.2505**

RefCOCO+

	Test A					Test B				
	Acc	BLEU 1	BLEU 2	ROUGE	METEOR	Acc	BLEU 1	BLEU 2	ROUGE	METEOR
MLE[36]	62.10%	**0.391**	**0.218**	**0.356**	0.140	46.21%	0.331	0.174	0.322	0.135
MMI[36]	67.79%	0.370	0.203	0.346	0.136	55.21%	0.324	0.167	0.320	0.133
CL	68.54%	0.3683	0.2041	0.3386	0.1375	**55.87%**	**0.3409**	**0.1829**	**0.3432**	**0.1455**
MSS	**69.41%**	0.3763	0.2126	0.3425	0.1401	55.59%	0.3386	0.1823	0.3365	0.1424
SMIXEC	69.05%	0.3847	0.2125	0.3507	**0.1436**	54.71%	0.3275	0.1716	0.3194	0.1354
MLE+sample	62.45%	0.3925	0.2256	0.3581	0.1456	47.86%	0.3354	0.1819	0.3370	0.1470
Rerank	**77.32%**	**0.3956**	**0.2284**	**0.3636**	**0.1484**	**67.65%**	**0.3368**	**0.1843**	**0.3441**	**0.1509**

Table 3: Expression generation evaluated by automated metrics. Acc: accuracy of the trained comprehension model on generated expressions. We separately mark in bold the best results for single-output methods (top) and sample-based methods (bottom) that generate multiple expressions and select one.

RefCOCOg (val)

	Acc(%)	Bleu 1	Bleu 2	Rouge	Meteor
Max Lik.	61.96%	0.437	0.273	0.363	0.149
MMI	70.38%	0.428	0.263	0.354	0.144
CL	70.74%	**0.4439**	**0.2751**	**0.3695**	0.1552
MSS	**70.80%**	0.4377	0.2697	0.3633	0.1524
SMIXEC	70.02%	0.4338	0.2683	0.3650	**0.1575**
sample	66.72%	0.4406	0.2755	0.3748	0.1526
Rerank	**76.65%**	**0.4410**	**0.2772**	**0.3782**	**0.1536**

Table 4: Expression generation result on RefCOCOg val.

Table 5: Human evaluation results

	RefCOCO		RefCOCO+	
	Test A	Test B	TestA	TestB
MMI[36]	53%	61%	39%	35%
SMIXEC	62%	68%	**46%**	25%
Rerank	**66%**	**75%**	43%	**47%**

SMIXEC and our generate-and-rerank method are in Table 5. On RefCOCO, both of our comprehension-guided methods appear to generate better (more informative) referring expressions. On RefCOCO+, the result are similar to those on RefCOCO on TestA, but our training by proxy methods performs less well on TestB. Fig 5 shows some example generation results on test images.

6. Conclusion

In this paper, we propose to use learned comprehension models to guide generating better referring expressions. Comprehension guidance can be incorporated at training time, with a training by proxy method, where the discriminative comprehension loss (region retrieval based on generated referring expressions) is included in training the expression generator. Alternatively comprehension guidance can be used at test time, with a generate-and-rerank method which uses model comprehension score to select among multiple proposed expressions. Empirical evaluation shows both to be promising, with the generate-and-rerank method obtaining particularly good results across data sets.

Among directions for future work we are interested to explore alternative training regimes, in particular an adaptation of the GAN protocol to referring expression generation. We will try to incorporate context objects (other regions in the image) into representation for a reference region. Finally, while at the moment the generation and comprehension models are completely separate, it is interesting to consider weight sharing.

Acknowledgments We thank Licheng Yu for providing the baseline code. We also thank those who participated in evaluations. Finally, we gratefully acknowledge the support of NVIDIA for the donation of GPUs used in this research.

References

[1] J. Andreas and D. Klein. Reasoning About Pragmatics with Neural Listeners and Speakers. *1604.00562V1*, 2016. 2, 6

[2] D. Bahdanau, P. Brakel, K. Xu, A. Goyal, R. Lowe, J. Pineau, A. Courville, and Y. Bengio. An Actor-Critic Algorithm for Sequence Prediction. *arXiv:1607.07086v1 [cs.LG]*, 2016. 4

[3] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*, pages 1171–1179, 2015. 5

[4] B. Dhingra, H. Liu, W. W. Cohen, and R. Salakhutdinov. Gated-Attention Readers for Text Comprehension. *ArXiV*, 2016. 3

[5] Dzmitry Bahdana, D. Bahdanau, K. Cho, and Y. Bengio. Neural Machine Translation By Jointly Learning To Align and Translate. *Iclr 2015*, pages 1–15, 2014. 4

[6] H. J. Escalante, C. A. Hernández, J. A. Gonzalez, A. López-López, M. Montes, E. F. Morales, L. Enrique Sucar, L. Villaseñor, and M. Grubinger. The segmented and annotated IAPR TC-12 benchmark. *Computer Vision and Image Understanding*, 114(4):419–428, 2010. 6

[7] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, T. Mikolov, et al. Devise: A deep visual-semantic embedding model. In *Advances in neural information processing systems*, 2013. 2

[8] A. Fukui, D. H. Park, D. Yang, A. Rohrbach, T. Darrell, and M. Rohrbach. Multimodal Compact Bilinear Pooling for Visual Question Answering and Visual Grounding. *Arxiv*, 2016. 2, 7

[9] R. Girshick. Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1440–1448, 2015. 6

[10] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014. 1

[11] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014. 5

[12] A. Graves, A.-r. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. IEEE, 2013. 3

[13] M. Grübinger, P. Clough, H. Müller, and T. Deselaers. The IAPR TC-12 Benchmark: A New Evaluation Resource for Visual Information Systems. *LREC Workshop OntoImage Language Resources for Content-Based Image Retrieval*, pages 13–23, 2006. 6

[14] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 3

[15] R. Hu, H. Xu, M. Rohrbach, J. Feng, K. Saenko, and T. Darrell. Natural Language Object Retrieval. *arXiv preprint*, pages 4555–4564, 2015. 1, 2, 3, 6, 7

[16] J. Johnson, A. Karpathy, and L. Fei-Fei. DenseCap: Fully Convolutional Localization Networks for Dense Captioning. *arXiv preprint*, 2015. 2

[17] A. Karpathy and L. Fei-Fei. Deep visual-semantic alignments for generating image descriptions. *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*, pages 3128–3137, 2015. 2

[18] S. Kazemzadeh, V. Ordonez, M. Matten, and T. L. Berg. ReferItGame: Referring to Objects in Photographs of Natural Scenes. *Emnlp*, pages 787–798, 2014. 2, 6

[19] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[20] 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 *European Conference on Computer Vision*, pages 740–755. Springer, 2014. 6

[21] C. Liu, J. Mao, F. Sha, and A. Yuille. Attention Correctness in Neural Image Captioning. pages 1–11, 2016. 2

[22] L. Ma, Z. Lu, L. Shang, and H. Li. Multimodal convolutional neural networks for matching image and sentence. *Proceedings of the IEEE International Conference on Computer Vision*, 11-18-Dece:2623–2631, 2016. 2

[23] J. Mao, J. Huang, A. Toshev, O. Camburu, A. Yuille, and K. Murphy. Generation and Comprehension of Unambiguous Object Descriptions. *Cvpr*, pages 11–20, 2016. 1, 2, 3, 6

[24] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013. 3

[25] V. K. Nagaraja, V. I. Morariu, and L. S. Davis. Modeling Context Between Objects for Referring Expression Understanding. *Eccv*, 2016. 2, 3, 6, 7

[26] A. Radford, L. Metz, and S. Chintala. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. *arXiv*, pages 1–15, 2015. 1

[27] M. Ranzato, S. Chopra, M. Auli, and W. Zaremba. Sequence Level Training with Recurrent Neural Networks. *Iclr*, pages 1–15, 2016. 4, 5

[28] S. Reed, Z. Akata, H. Lee, and B. Schiele. Learning Deep Representations of Fine-Grained Visual Descriptions. *Cvpr*, pages 49–58, 2016. 2

[29] A. Rohrbach, M. Rohrbach, R. Hu, T. Darrell, and B. Schiele. Grounding of Textual Phrases in Images by Reconstruction. *1511.03745V1*, 1:1–10, 2015. 2, 3, 6, 7

[30] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 3

[31] I. Vendrov, R. Kiros, S. Fidler, and R. Urtasun. Order-Embeddings of Images and Language. *arXiv preprint*, (2005):1–13, 2015. 2

[32] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3156–3164, 2015. 2

[33] L. Wang, Y. Li, and S. Lazebnik. Learning Deep Structure-Preserving Image-Text Embeddings. *Cvpr*, (Figure 1):5005–5013, 2016. 2

[34] J. Weston, S. Bengio, and N. Usunier. Wsabie: Scaling up to large vocabulary image annotation. In *Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI*, 2011. 2

[35] K. Xu, J. L. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. S. Zemel, and Y. Bengio. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. *Icml-2015*, 2015. 2, 4

[36] L. Yu, P. Poirson, S. Yang, A. C. Berg, and T. L. Berg. Modeling Context in Referring Expressions. In *Eccv*, 2016. 2, 3, 6, 7, 8

[37] L. Yu, W. Zhang, J. Wang, and Y. Yu. SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient. 2016. 2, 4

[38] L. Zitnick and P. Dollar. Edge boxes: Locating object proposals from edges. In *ECCV*. European Conference on Computer Vision, September 2014. 6