A Fast and Accurate One-Stage Approach to Visual Grounding

Zhengyuan Yang2∗ Boqing Gong1† Liwei Wang1 Wenbing Huang1 Dong Yu1 Jiebo Luo2

†Tencent AI Lab 2University of Rochester

{zyang39, jluo}@cs.rochester.edu, boqinggo@outlook.com

{liweiwang, dongyu}@tencent.com, hwenbing@126.com

Abstract

We propose a simple, fast, and accurate one-stage approach to visual grounding, inspired by the following insight. The performances of existing propose-and-rank two-stage methods are capped by the quality of the region candidates they propose in the first stage — if none of the candidates could cover the ground truth region, there is no hope in the second stage to rank the right region to the top. To avoid this caveat, we propose a one-stage model that enables end-to-end joint optimization. The main idea is as straightforward as fusing a text query’s embedding into the YOLOv3 object detector, augmented by spatial features so as to account for spatial mentions in the query. Despite being simple, this one-stage approach shows great potential in terms of both accuracy and speed for both phrase localization and referring expression comprehension, according to our experiments. Given these results along with careful investigations into some popular region proposals, we advocate for visual grounding a paradigm shift from the conventional two-stage methods to the one-stage framework.

1. Introduction

We propose a simple, fast, and accurate one-stage approach to visual grounding, which aims to ground a natural language query (phrase or sentence) about an image onto a correct region of the image. By defining visual grounding at this level, we deliberately abstract away the subtle distinctions between phrase localization [30, 42], referring expression comprehension [15, 24, 48, 47, 22], natural language object retrieval [14, 16], visual question segmentation [9, 13, 20, 25], etc., each of which can be seen as a variation of the general visual grounding problem. We benchmark our one-stage approach for both phrase localization and referring expression comprehension. Results show that it is about 10 times faster than the state-of-the-art two-stage methods and meanwhile more accurate than them. Hence, we expect this work provides for visual grounding a new strong baseline, upon which one can conveniently build further to tackle variations (e.g., phrase localization) to the basic visual grounding problem by bringing in corresponding domain knowledge (e.g., attributes, relationship between phrases, spatial configuration of regions, etc.).

Visual grounding is key to machine intelligence and provides a natural channel for humans to communicate with machines about the physical world. Its potential applications include but are not limited to robotics, human-computer interaction, and early education. In addition, a good visual grounding model can benefit a variety of research problems such as visual question answering [53, 9, 17], image captioning [45, 1, 8], and image retrieval [37].

There are mainly two thriving threads of work in visual grounding: phrase localization [15, 30, 42] and referring expression comprehension [24, 48, 47, 14, 16] — plus some work on grounding as segmentation [13, 9, 20, 25]. The language query in the former is a local phrase of a full sentence
describing an image, implying that multiple phrase queries could co-occur in the sentence. In the latter, the query is an expression referring to a particular region of an image through a combination of object categories, attributes, relationships with other objects, etc. Notably, in phrase localization, an image region linked to a phrase of one sentence can also be linked to a phrase of another sentence, establishing a coreference chain. Compared with phrase localization, referring expression has less ambiguity in general.

Recent advances in computer vision and natural language processing offer a rich set of tools, such as region proposals [54, 41], object detection [10, 34, 11], text embedding [28, 26, 4], syntactic parsing [39], etc., leading to methods [43, 42, 29, 2, 49, 47] exploiting various cues in the visual grounding problem. However, somehow surprisingly, the main bodies of these methods are remarkably alike: they propose multiple region candidates per image and then rank them according to their similarities with the language query. We contend that this propose-and-rank two-stage framework is flawed in at least two major ways.

- If none of the region candidates of the first stage hits the ground truth region, the whole framework fails no matter how good the second ranking stage could perform. We find that 200 Edgebox region proposals [54] per image can only hit 68% of the ground truth regions in ReferItGame [15], a benchmark dataset for referring expression comprehension. A hit is considered successful if any of the 200 proposals could reach 0.5 or higher intersection-over-union (IoU) [30] with the ground truth region.
- Most of the computation spent on the region candidates, such as generating proposals, extracting features, fusing with the query embedding, scoring similarities, and so on, are merely to rank them down to the list. After all, in most test cases, only one or two region proposals are correct. We believe this scheme is a waste of computation and should be improved.

The two caveats are left unresolved probably due to the long-standing pursuit of how to model different cues in visual grounding. In this paper, we take a step back and re-examine the visual grounding problem at an abstract level, without discriminating the query types. We propose to shift the paradigm from grounding as ranking multiple region candidates to directly proposing one region as the output.

To this end, we study an end-to-end one-stage approach to visual grounding. The main idea is as straightforward as fusing a text query’s embedding into the YOLOv3 object detector [33]. Additionally, we augment the feature maps with spatial features to account for spatial mentions in the language queries (e.g., “the man on the right”). Finally, we replace the sigmoid output layer with a softmax function in order to enforce the network to generate only one image region in response to a query. Other cues explored in the two-stage methods, such as attributes, attention, bounding box annotations around extra objects, and so on, can be naturally added to our one-stage model. We focus on the vanilla model in the main text and examine its extensibilities to some other cues in supplementary materials.

The advantages of this one-stage approach are multiple-fold. First of all, it is fast in inference. It extracts features from the input image with only one pass and then directly predicts the coordinates of the output region. Without any code optimization, our implementation is about 10 times faster than state-of-the-art two-stage methods. Additionally, it is also accurate. Unlike the two-stage framework whose performance is capped by the region candidates, it enables end-to-end optimization. We show promising results on both phrase localization and referring expression comprehension. Finally, it generalizes better to different datasets than the two-stage methods because it does not depend on any additional tools or pre-trained models. Hence, we advocate this one-stage framework for future work on visual grounding and hope our approach in this work provides a new strong baseline.

2. Approach

In this section, we first review the existing two-stage frameworks for visual grounding [42, 30, 48, 24, 47, 35, 29] and then present our one-stage approach in detail.

2.1. Two-stage methods

Conventional methods for visual grounding, especially for the task of phrase localization [30, 42, 29, 2], are mainly composed of two separate stages. As shown in Figure 1, given an input image, the first step is to generate candidate regions using either unsupervised object proposal methods [54, 42, 29, 2] or a pre-trained object detection network [50, 47]. The second step is to rank the candidate regions conditioning on a language query about the image. Most existing two-stage methods differ from each other in the second step by scoring functions, network architectures, multi-task learning, and training algorithms. A number of studies [51, 42] cast the second step as a binary classification task, where a region-query pair is tagged “positive”, “negative”, or “ignored” based on the region’s IoU with the ground truth region. The maximum-margin ranking loss is another popular choice for the second stage [24, 27, 43].

As a concrete example, we next describe the similarity network [42, 29] since it gives rise to state-of-the-art performance on benchmark datasets. The authors employ a Fast R-CNN [10, 34] pre-trained on Pascal [7] to extract visual features for each candidate region. To embed the text query, they find the Fisher encoding [28] works as well as or better than recurrent neural networks. The region features and query embeddings are fed through two network branches, respectively, before they merge by a layer of element-wise
multiplication. A few nonlinear layers are added after they merge. Finally, the network outputs a similarity score by a sigmoid function. The authors train this network by a cross-entropy loss with positive labels for the matched pairs of regions and queries and negative labels for the mismatched pairs. A region is a match to the query if its IoU with the ground truth is greater than 0.7 and the regions with IoUs less than 0.3 are considered mismatches.

The overall performance of the two-stage framework is capped by the first stage. Besides, the candidate regions cause heavy computation cost. We next present a different paradigm, a one-stage visual grounding network which enables end-to-end optimization and is both fast and accurate.

2.2. Our one-stage approach

In short, our one-stage approach to the visual grounding is to fuse a text query’s embedding into YOLOv3 [33], augment it with spatial features as the spatial configuration is frequently used by a query, replace its sigmoid output layer with a softmax function because we only need to return one region for a query, and finally train the network with YOLO’s loss [31]. Despite being simple, this one-stage method signifies a paradigm shift away from the prevalent two-stage framework, and it gives rise to superior results in terms of both accuracy and speed.

We present this vanilla one-stage model as below and amend it in supplementary materials to account for some cues explored in the two-stage methods. Figure 2 illustrates the network architecture, mainly consisting of three feature encoding modules and three fusion modules.

Visual and text feature encoding. Our model is end-to-end, taking as input an image and a text query and then returning an image region as the response to the query. For the text query, we embed it to a 768D real-valued vector using the uncased version of Bert [4], followed by two fully connected layers either with 512 neurons. In addition, we also test the other embedding methods to fairly compare with the existing works. In particular, recent works [30, 42, 29] employ Fisher vectors of word2vec [28, 26]. Bidirectional LSTMs are adopted in [2, 35].

We use Darknet-53 [33] with feature pyramid networks [18] to extract visual features for the input image, which is resized to 256 × 256, at three spatial resolutions: 8 × 8 × D1, 16 × 16 × D2, and 32 × 32 × D3. In other words, the feature maps are respectively \( \frac{1}{32} \), \( \frac{1}{16} \), and \( \frac{1}{8} \) of the original image size. There are \( D_1 = 1024 \), \( D_2 = 512 \), and \( D_3 = 256 \) feature channels at the three resolutions, respectively. We add a 1 × 1 convolution layer with batch normalization and RELU to map them all to the same dimension \( D = 512 \).

Spatial feature encoding. We find that the text queries often use spatial configurations to refer to objects, such as “the man on the left” and “the bottom right grass”. However, the Darknet-53 features mainly capture visual appearances, lacking the position information. Hence, we explicitly encode some spatial features for each position of the three spatial resolutions. Specifically, as shown in Figure 2, we generate a coordinate map of size \( W' \times H' \times D_{\text{spatial}} \) at each resolution, where \( W' \) and \( H' \) are the spatial size of a visual feature map, i.e., 8 × 8, 16 × 16, or 32 × 32, and \( D_{\text{spatial}} = 8 \) indicating we encode eight spatial features. If we place the feature map in a coordinate system such that its top-left and bottom-right corners lie at \((0,0)\) and \((1,1)\), respectively, the eight features for any position \((i,j)\), \(i \in \{0,1,\cdots,W'-1\}\) and \(j \in \{0,1,\cdots,H'-1\}\), are calculated as follows:

\[
\left( \frac{i}{W'}, \frac{j}{H'}, \frac{i+0.5}{W'}, \frac{j+0.5}{H'}, \frac{i+1}{W'}, \frac{j+1}{H'}, \frac{1}{W'}, \frac{1}{H'} \right),
\]

which captures the coordinates of the top-left corner, center, and bottom-right corner of the grid at \((i,j)\), along with the inverse of \(W'\) and \(H'\).

Fusion. We use the same operation to fuse the visual, text, and spatial features at the three spatial resolutions. In particular, we first broadcast the query embedding to each spatial location \((i,j)\) and then concatenate it with the visual and spatial features, giving rise to a 512 + 512 + 8 = 1032D feature vector. The visual, text, and spatial features are \( \ell_2 \) normalized respectively before the concatenation. We add a 1 × 1 convolution layer to better fuse them at each location independently. We have also tested \( 3 \times 3 \) convolution kernels hoping to make the fusion aware of the neighborhood structure, and yet the results are about the same as the 1 × 1 fusion. After this fusion step, we have a 512D feature vector for each location of the three spatial resolutions, i.e., three feature blobs of the sizes 8 × 8 × 512, 16 × 16 × 512, and 32 × 32 × 512, respectively.

Grounding. The grounding module takes the fused features as input and generates a box prediction to ground the language query onto an image region. We design this module by following YOLOv3’s output layer except that we 1) recalibrate its anchor boxes and 2) change its sigmoid layer to a softmax function.

There are 8 × 8 + 16 × 16 + 32 × 32 = 1344 locations out of the three spatial resolutions — and each location is associated with a 512D feature vector as a result of the fusion module. YOLOv3 centers around each of the locations three anchor boxes. To better fit our grounding datasets, we customize the widths and heights of the anchors by \( K \)-means clustering over all the ground truth grounding boxes in the training set with \((1 - \text{IoU})\) as the distance [32, 33].

There are (3 anchors per location × 1344 locations) = 4032 anchor boxes in total. What YOLOv3 predicts is, out of each anchor box, four quantities by regression for shifting the center, width, and height of the anchor box, along with the fifth quantity by a sigmoid function about the confidence on this shifted box. We keep the regression branch as is. As only one region is desired as the output for grounding
the query — at least according to the current formalization of the visual grounding problem, we replace the sigmoid functions with a softmax function over all the 4032 boxes. Accordingly, we replace the loss function over the confidence scores by a cross entropy between this softmax and a one-hot vector — the anchor box which has the highest IoU with the ground truth region is labeled 1 and all the others are labeled 0. We refer readers to [33] for more details.

2.3. Comparison to other one-stage grounding work

We contrast our approach to some closely related works, including two existing one-stage grounding methods [52, 44] and some on grounding as segmentation [13, 20, 25, 9].

The Interpretable and Globally Optimal Prediction (IGOP) [44] also tries to solve supervised visual grounding in a one-stage manner. IGOP employs feature maps from multiple vision tasks (e.g., object detection, semantic segmentation, pose estimation, etc.) and models the phrase localization task as finding a box on the feature maps which encapsulates the smallest energy. Since IGOP relies on multiple extra pre-trained vision models, it is not clear how to optimize it end-to-end.

The Multiple-scale Anchored Transformer Network (MATN) [52] is also a one-stage grounding model. However, many design considerations of this network are account for weakly supervised visual grounding. Besides, MATN directly predicts a single box as the output, essentially searches for one box out of a huge search space at the scale $O(W^2H^2)$, where $W, H$ are width and height of the input image. This scheme has been shown inferior to those based on anchor boxes in object detection [32, 33], unless one has sufficiently big training sets.

We also briefly discuss some works on grounding text queries to segmentation masks [13, 20, 25, 9]. Due to the irregular shapes of the segmentation masks, it is hard to follow the propose-and-rank two-stage framework to output segmentation masks. Instead, they naturally employ one-stage frameworks. However, their network architectures, especially the output layer, are very different from ours.

3. Experiments

3.1. Datasets and experiment protocols

We evaluate the proposed one-stage visual grounding approach on the Flickr30K Entities dataset [30] and the ReferItGame dataset [15]. The supplementary materials contain additional results on RefCOCO [48]. Flickr30K Entities augments the original Flickr30K [46] with region-phrase correspondence annotations. It links 31,783 images in Flickr30K with 427K referred entities. We follow the same training/validation/test split used in the previous work [30] in our experiments. ReferItGame [15] has 20,000 images from the SAIAPR-12 dataset [6]. We employ a cleaned version of the split provided by [14], which has 9,000,1,000 and 10,000 images in the training, validation, and test sets, respectively. Following the same evaluation protocol in prior works [30, 35], given a language query, an output image region is considered correct if its IoU is at least 0.5 with the ground truth bounding box.

Some details of our model architecture. We use Darknet-53 [33] pre-trained on COCO object detection [19] as the visual encoder. To embed the language queries, we test Bert [4], a bi-LSTM framework used in [2], and a Fisher vector encoding used in [30, 42, 29]. We generate the anchor boxes by $K$-means clustering following the procedure of [32, 33]. The anchors on ReferItGame are $(18 \times 22), (48 \times 28), (29 \times 52), (91 \times 48), (50 \times 91), (203 \times 57), (96 \times 127), (234 \times 100), (202 \times 175)$ and on Flickr30K Entities are $(17 \times 16), (33 \times 35), (84 \times 43), (50 \times 74), (76 \times 126), (125 \times 81), (128 \times 161), (227 \times 104), (216 \times 180)$.

Training details. We keep the original image ratio when we resize an input image. We resize its long edge to 256 and then pad the image pixels’ mean value along the short edge so that the final image size is 256 × 256. We fol-
low [33] for data augmentation, i.e., adding randomization to the color space (saturation and intensity), horizontal flip, and random affine transformations. We train the model with RMSProp [40] optimization. We start with a learning rate of $10^{-4}$ and follow a polynomial schedule with a power of 1. As the Darknet has been pre-trained, we multiply the main learning rate by 0.1 for the Darknet portion of our model. The batch size is 32 in all our experiments. We observe about 1% improvement when we use larger batch sizes on a workstation with eight P100 GPUs, but we opt to report the results of the small batch size (32) so that one can easily reproduce our results on a desktop with two GPUs.

**Competing baselines beyond existing methods.** We compare with state-of-the-art visual grounding methods, about which the descriptions are deferred to Section 3.2. Beyond them, we also systematically study the following baselines and variations of our approach.

- **Similarity Net-Darknet.** Previous two-stage methods often use detection networks with a VGG-16 backbone [38] to extract visual features, while Darknet is adopted in our model. Naturally, one may wonder about the influence of the backbones in addition to the framework change from the two stages to the one stage. To single out the influence of the backbone networks, we construct a baseline using the two-stage similarity network [42] based on the Darknet visual features, modifying the code released with [29]. We first pool region features from all three feature blobs output by the feature pyramid network in YOLOv3, then $\ell_2$ normalize them, respectively, and finally concatenate them as the visual features.

- **Similarity Net-Resnet.** We also test in the similarity network visual features extracted by Mask R-CNN [11] with a Resnet-101 [12] backbone, which is pre-trained on COCO detection. The feature dimension is 2048.

- **CITE-Resnet.** Furthermore, we compare to CITE [29] with Resnet-101 features. We keep the number of embeddings as the default value $K = 4$ in CITE. Region proposals and visual and language encoders remain the same as “Similarity Net-Resnet”.

- **Ours-FV.** The Fisher vector (FV) encoding [28] of word2vec [26] features is used in some state-of-the-art visual grounding methods [30, 42, 29]. We include it in our approach as well. A language query is encoded to a 6000D FV embedding.

- **Ours-LSTM.** The LSTM encoding of language queries is also frequently used in the literature [2, 35], so we investigate its effect in our approach as well. We use a bi-LSTM layer with 512D hidden states in this work. We do not use word2vec features to initialize the embedding layer.

- **Ours-Bert.** We use the uncased version of Bert [4] that outputs a 768D embedding as our main language query encoder. We do not update the Bert parameters during training.

- **Ours-Bert-no Spatial.** In this ablated version of our approach, we remove the spatial features and only fuse the visual and text features.

### 3.2. Visual grounding results

**Flickr30K Entities.** Table 1 reports the phrase localization results on the Flickr30K Entities dataset. The top portion of the table contains the numbers of several state-of-the-art visual grounding methods [14, 43, 35, 30, 44, 2, 42, 29]. The results of two additional versions of the similarity network [42], respectively based on Resnet and Darkent, are shown in the middle of the table. Finally, the four rows at the bottom are different variations of our own approach.

We list in the “Region Proposals” column different region proposal techniques used in the visual grounding methods, followed by the number of region candidates per image (e.g., N=100). Edgebox [54] and selective search [41] are two popular options for proposing the regions. In the “Language Features” column, we list the backbone networks followed by the datasets on which they are pre-trained. The “Language Embedding” column indicates the query embedding adopted by each grounding method.

Among the two-stage methods, not surprisingly, “Similarity Net-Resnet” gives much better results than “Similarity Net” because the Resnet visual features are generally higher-quality than the VGG features.

Although Darknet-53 and Resnet-101 give rise to comparable results on ImageNet [36], the Darknet features lead to poor visual grounding results. This is reasonable because Darknet does not have a separate region proposal network, making it tricky to extract the region features. Furthermore, the large down-scale ratios (1/8, 1/16, and 1/32) and the low feature dimensions (256, 512, 1024) of Darknet make its region features not as discriminative as Resnet’s.

Our one-stage method and its variations outperform the two-stage approaches with large margins. By the last two rows of the table, we investigate the effectiveness of our spatial features. It is clear that the spatial information boosts the accuracy of “Ours-Bert-no Spatial” by about 1.6%. Finally, we note that language embedding techniques only slightly influence the results within a small range.

**ReferitGame.** Table 2 reports the referring expression comprehension results on ReferitGame [15]. Organizing the results in the same way as Table 1, the top portion of the table is about state-of-the-art grounding methods [14, 23, 35, 30, 44, 2, 42, 29], the middle is two versions of the similarity network, and the bottom shows our results.

We draw from Table 2 about the same observation as from Table 1. In general, our model with Darknet visual fea-
Table 1. Phrase localization results on the test set of Flickr30K Entities [30].

<table>
<thead>
<tr>
<th>Method</th>
<th>Region Proposals</th>
<th>Visual Features</th>
<th>Language Embedding</th>
<th>Accu@0.5</th>
<th>Time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCRC [14]</td>
<td>Edgebox N=100</td>
<td>VGG16-Imagenet</td>
<td>LSTM</td>
<td>27.80</td>
<td>-</td>
</tr>
<tr>
<td>DSPE [43]</td>
<td>Edgebox N=100</td>
<td>VGG19-Pascal</td>
<td>Word2vec, FV</td>
<td>43.89</td>
<td>-</td>
</tr>
<tr>
<td>GroundeR [35]</td>
<td>Select. Search N=100</td>
<td>VGG16-Pascal</td>
<td>LSTM</td>
<td>47.81</td>
<td>-</td>
</tr>
<tr>
<td>CCA [30]</td>
<td>Edgebox N=200</td>
<td>VGG19-Pascal</td>
<td>Word2vec, FV</td>
<td>50.89</td>
<td>-</td>
</tr>
<tr>
<td>IGOP [44]</td>
<td>None</td>
<td>Multiple Network</td>
<td>N-hot</td>
<td>53.97</td>
<td>-</td>
</tr>
<tr>
<td>MCB + Reg + Spatial [2]</td>
<td>Select. Search N=100</td>
<td>VGG16-Pascal</td>
<td>LSTM</td>
<td>51.01</td>
<td>-</td>
</tr>
<tr>
<td>MNN + Reg + Spatial [2]</td>
<td>Select. Search N=100</td>
<td>VGG16-Pascal</td>
<td>LSTM</td>
<td>55.99</td>
<td>-</td>
</tr>
<tr>
<td>Similarity Net [42]</td>
<td>Edgebox N=200</td>
<td>VGG19-Pascal</td>
<td>Word2vec, FV</td>
<td>51.05</td>
<td>-</td>
</tr>
<tr>
<td>Similarity Net by CITE [29]</td>
<td>Edgebox N=200</td>
<td>VGG16-Pascal</td>
<td>Word2vec, FV</td>
<td>54.52</td>
<td>-</td>
</tr>
<tr>
<td>CITE [29]</td>
<td>Edgebox N=500</td>
<td>VGG16-Pascal</td>
<td>Word2vec, FV</td>
<td>59.27</td>
<td>-</td>
</tr>
<tr>
<td>CITE [29]</td>
<td>Edgebox N=500</td>
<td>VGG16-Flickr30K</td>
<td>Word2vec, FV</td>
<td>61.89</td>
<td>-</td>
</tr>
<tr>
<td>Similarity Net-Resnet [42]</td>
<td>Edgebox N=200</td>
<td>Res101-COCO</td>
<td>Word2vec, FV</td>
<td>60.89</td>
<td>184</td>
</tr>
<tr>
<td>CITE-Resnet [29]</td>
<td>Edgebox N=200</td>
<td>Res101-COCO</td>
<td>Word2vec, FV</td>
<td>61.33</td>
<td>196</td>
</tr>
<tr>
<td>Similarity Net-Darknet [42]</td>
<td>Edgebox N=200</td>
<td>Darknet53-COCO</td>
<td>Word2vec, FV</td>
<td>41.04</td>
<td>305</td>
</tr>
<tr>
<td>Ours-FV</td>
<td>None</td>
<td>Darknet53-COCO</td>
<td>Word2vec, FV</td>
<td>68.38</td>
<td>16</td>
</tr>
<tr>
<td>Ours-LSTM</td>
<td>None</td>
<td>Darknet53-COCO</td>
<td>LSTM</td>
<td>67.62</td>
<td>21</td>
</tr>
<tr>
<td>Ours-Bert-no Spatial</td>
<td>None</td>
<td>Darknet53-COCO</td>
<td>Bert</td>
<td>67.08</td>
<td>38</td>
</tr>
<tr>
<td>Ours-Bert</td>
<td>None</td>
<td>Darknet53-COCO</td>
<td>Bert</td>
<td>59.30</td>
<td>38</td>
</tr>
</tbody>
</table>

Table 2. Referring expression comprehension results on the test set of ReferItGame [15].

<table>
<thead>
<tr>
<th>Method</th>
<th>Region Proposals</th>
<th>Visual Features</th>
<th>Language Embedding</th>
<th>Accu@0.5</th>
<th>Time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCRC [14]</td>
<td>Edgebox N=100</td>
<td>VGG16-Imagenet</td>
<td>LSTM</td>
<td>17.93</td>
<td>-</td>
</tr>
<tr>
<td>VCC + Spatial [35]</td>
<td>Edgebox N=100</td>
<td>VGG16-Pascal</td>
<td>LSTM</td>
<td>26.93</td>
<td>-</td>
</tr>
<tr>
<td>VC [50]</td>
<td>SSD Detection [21]</td>
<td>VGG16-COCO</td>
<td>LSTM</td>
<td>31.13</td>
<td>-</td>
</tr>
<tr>
<td>CGRE [23]</td>
<td>Edgebox</td>
<td>VGG16</td>
<td>LSTM</td>
<td>31.85</td>
<td>-</td>
</tr>
<tr>
<td>MCB + Reg + Spatial [2]</td>
<td>Edgebox N=100</td>
<td>VGG16-Pascal</td>
<td>LSTM</td>
<td>26.54</td>
<td>-</td>
</tr>
<tr>
<td>MNN + Reg + Spatial [2]</td>
<td>Edgebox N=100</td>
<td>VGG16-Pascal</td>
<td>LSTM</td>
<td>32.21</td>
<td>-</td>
</tr>
<tr>
<td>Similarity Net by CITE [29]</td>
<td>Edgebox N=500</td>
<td>VGG16-Pascal</td>
<td>Word2vec, FV</td>
<td>31.26</td>
<td>-</td>
</tr>
<tr>
<td>CITE [29]</td>
<td>Edgebox N=500</td>
<td>VGG16-Pascal</td>
<td>Word2vec, FV</td>
<td>34.13</td>
<td>-</td>
</tr>
<tr>
<td>IGOP [44]</td>
<td>None</td>
<td>Multiple Network</td>
<td>N-hot</td>
<td>34.70</td>
<td>-</td>
</tr>
<tr>
<td>Similarity Net-Resnet [42]</td>
<td>Edgebox N=200</td>
<td>Res101-COCO</td>
<td>Word2vec, FV</td>
<td>34.54</td>
<td>184</td>
</tr>
<tr>
<td>CITE-Resnet [29]</td>
<td>Edgebox N=200</td>
<td>Res101-COCO</td>
<td>Word2vec, FV</td>
<td>35.07</td>
<td>196</td>
</tr>
<tr>
<td>Similarity Net-Darknet [42]</td>
<td>Edgebox N=200</td>
<td>Darknet53-COCO</td>
<td>Word2vec, FV</td>
<td>22.37</td>
<td>305</td>
</tr>
<tr>
<td>Ours-FV</td>
<td>None</td>
<td>Darknet53-COCO</td>
<td>Word2vec, FV</td>
<td>59.18</td>
<td>16</td>
</tr>
<tr>
<td>Ours-LSTM</td>
<td>None</td>
<td>Darknet53-COCO</td>
<td>LSTM</td>
<td>58.76</td>
<td>21</td>
</tr>
<tr>
<td>Ours-Bert-no Spatial</td>
<td>None</td>
<td>Darknet53-COCO</td>
<td>Bert</td>
<td>58.16</td>
<td>38</td>
</tr>
<tr>
<td>Ours-Bert</td>
<td>None</td>
<td>Darknet53-COCO</td>
<td>Bert</td>
<td>59.30</td>
<td>38</td>
</tr>
</tbody>
</table>

The performance of the one-stage methods is significantly improved over the two-stage ones. The two-stage approaches generally take more than 180ms to generate region candidates and extract features for them. In contrast, our one-stage approaches all take less than 40ms to ground a language query to an image — especially, “Ours-FV” takes only 16ms, making it potentially feasible for real-time applications.

3.3. Inference time comparison

A fast inference speed is one of the major advantages of our one-stage method. We list the inference time in the rightmost columns of Tables 1 and 2, respectively. We conduct all the tests on a desktop with Intel Core i9-9900K@3.60GHz and NVIDIA 1080TI. Typical two-stage approaches generally take more than 180ms to process one image-query pair, and they spend most of the time on generating region candidates and extracting features for them. In contrast, our one-stage approaches all take less than 40ms to ground a language query to an image — especially, “Ours-FV” takes only 16ms, making it potentially feasible for real-time applications.

3.4. Oracle analyses about the region candidates

Why could the one-stage methods achieve those big improvements over the two-stage ones? We conjecture that it is mainly because our one-stage framework can avoid imperfect region candidates. In contrast, the performances of the two-stage methods are capped by the hit rate of the region candidates they propose in the first stage. We say a
ground truth region is hit by the region candidates if its IoU is greater than 0.5 with any of the candidates, and the hit rate is the number of ground truth regions hit by the candidates divided by the total number of ground truth regions.

We study the hit rates of some popular region proposal methods: Edgebox [54], selective search [41], the region proposal network in Mask R-CNN [11] pre-trained on COCO [11], Mask R-CNN itself whose detection results are regarded as region candidates, and our one-stage approach whose box predictions are considered the region candidates. We keep top N=200 region candidates for each of them or as many regions as possible if it outputs less than 200 regions.

Table 3 shows the hit rates on both Flickr30K Entities and ReferItGame. It is interesting to see that the hit rates are in general higher on Flickr30 than on ReferItGame, especially when Edgebox generated proposals are used, somehow explaining why the two-stage grounding results on Flickr30K Entities (Table 1) are better than those on ReferItGame (Table 2). Another notable observation is that the top 200 boxes of our approach have much higher hit rates than the other techniques, verifying the benefit of learning in an end-to-end way.

One may wonder under what scenarios the region proposal methods but ours fail to hit the ground truth regions. Figure 3 gives some insights by showing the Edgebox region candidates on ReferItGame. We find that the region candidates mainly fail to hit stuff ground truth regions (e.g., the “grass” in Figure 3 (c)). Tiny objects are also hard to hit (cf. Figure 3 (e) and (f)). Finally, when a query refers to more than one objects, it might fail region proposal methods which are mostly designed to place a tight bounding box around only one object (cf. Figure 3 (a) and (b)).

### 3.5. Qualitative results analyses

In this section, we analyze the success and failure cases of the two-stage similarity network as well as our model to show the advantages and limitations of the proposed one-stage method. Figure 4 shows the mistakes made by the similarity network that can be corrected by our method. The blue boxes are predictions, and the yellow boxes represent the ground truths. We group some common mistakes into the following scenarios.

- **Queries referring to multiple objects.** A language query in the visual grounding problem can refer to more than one objects, but the region proposals by de-

![Table 3. Hit rates of region proposal methods.](image)

![Figure 3. Failure cases of the Edgebox region candidates (boxes shown in the red color) on ReferItGame. The yellow boxes are ground truths. For visualization purpose, we randomly hide some region candidates.](image)

- **Queries referring to stuff as opposed to things.** The second kind of common errors made by the two-stage methods is on the queries referring to stuff as opposed to things, such like the “grass” and “city” shown in Figures 4 (c) and (d), respectively. This kind of errors is again due to that the region proposals mostly focus on thing classes — the “objectness” is often an important cue for proposing the regions. In sharp contrast, stuff regions generally have low “objectness” scores. Hence, we argue that the two-stage methods are incapable of handling such stuff regions given the status quo of region proposal techniques. Our one-stage method can instead learn to handle the stuff regions from the training set of the visual grounding datasets.

- **Challenging regions.** In the third kind of common errors, the two-stage methods fail to deal with challenging test cases, such as the small regions referred to by the queries in Figures 4 (e) and (f). There are mainly three reasons that fail the two-stage methods. First, the region candidates of the first stage may not provide a good coverage especially over small objects. Second, the visual features of small regions are not discriminative enough for the second stage to learn how to rank. Third, the image depicts complicated scenes or many duplicated objects. The last point could equally harm our approach as well as the two-stage ones.
Failure cases of our approach. Figure 5 shows extra success and failure cases of our approach. The first row shows the typical success cases. The “bike of blue pant lady” in Figure 5 (a) queries an example image with multiple objects of the same class. Figure 5 (b) provides an example of correct predictions on tiny objects. (c) and (d) showcase our approach is able to interpret location information in the queries. The query in (e) contains a distracting noun, “sheep”. Our model in (f) successfully predicts a region containing two objects.

Figures 5 (g)–(l) are some failure cases of our model. We find our model is insensitive to attributes, such as the “blue” in (g) and (k). It fails on (h) and (i) simply because those are very difficult test cases (e.g., one has to recognize the word “colonial” in the image in order to make the right prediction). Finally, (j) and (l) give two ambiguous queries for which our model happens to predict different boxes from those annotated by users.

4. Conclusion

We have proposed a simple and yet effective one-stage method for visual grounding. We merge language queries and spatial features into the YOLOv3 object detector and build an end-to-end trainable visual grounding model. It is about 10 times faster than state-of-the-art two-stage methods and achieves superior grounding accuracy. Besides, our analyses reveal that existing region proposal methods are generally not good enough, capping the performance of the two-stage methods and indicating the need of a paradigm shift to the one-stage framework. In future work, we plan to investigate the extensibility of the proposed one-stage framework for modeling other cues in the visual grounding problem.

Acknowledgment

This work is partially supported by NSF awards #1704337, #1722847, and #1813709.
References


[43] Liwei Wang, Yin Li, and Svetlana Lazebnik. Learning deep structure-preserving image-text embeddings. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5005–5013, 2016. 2, 5, 6


