One-Stage Object Referring with Gaze Estimation

Jianhang Chen[*1,2], Xu Zhang[*2], Yue Wu[2], Shalini Ghosh[2], Pradeep Natarajan[2],
Shih-Fu Chang[2,3], Jan Allebach[1]
[1]Purdue University [2]Amazon [3]Columbia University
{jianhang321, spongezhang}@gmail.com, {wuayue, ghoshsha, natarap}@amazon.com,
sc250@columbia.edu, allebach@purdue.edu

Abstract

The classic object referring task aims at localizing the referred object in the image and requires a reference image and a natural language description as inputs. Given the facts that gaze signal can be easily obtained by a modern human-computer interaction system with a camera and that human tends to look at the object when referring to it, we propose a novel gaze-assisted object referring framework. The formulation not only simplifies the state-of-the-art gaze-assisted object referring system requiring many input signals besides gaze, but also incorporates the one-stage object detection idea to improve the inference efficiency. More importantly, it implicitly considers all object candidates and thus resolves the main pain point of existing two-stage object referring solutions for proposing an appropriate number of candidates – it cannot be too large, otherwise the computational cost can be prohibitive; it cannot be too small, otherwise the chance of missing a referred object can be significant. To utilize the gaze information, we propose to build a gaze heatmap by using the anchor position encoding map and the gaze prediction result. The gaze heatmap and the language feature are then merged into the feature pyramid in the object detection as the final one-stage referring system. In the CityScapes-OR dataset, the proposed method outperforms the state-of-the-art by 7.8% for Acc@1.

1. Introduction

Object referring (*ObjRef* , a.k.a. visual grounding) [32] is a multi-modal task that requires an understanding of both image and language and making the proper connection(s) between them. It has a wide variety of real-world applications, especially for human-computer interaction [24].

In the past, this task is typically defined as follows [30, 32], *given an image and a natural language description about an object within it, locate the referred object. ObjRef* by language is a popular task in computer vision. However, in a real-world scenario, the referring expression could be ambiguous or incomplete. People often utilize gaze and/or body language to confirm the target.

Figure 1. Gaze information helps improve the virtual shopping experience. Without gaze information, to correctly refer to the bag highlighted in the red box, the person need to say a complex description, e.g. "show me the green bag second from the left in the bottom row". If the virtual assistant can detect the gaze information, the person only needs to say, "show me that bag". The red star shows the predicted gaze position of the person.

In order to improve the referring accuracy, Vasudevan *et al*. [31] proposed to use additional input sources for the *ObjRef* task as they can provide complementary clues to help locate a referred object. More precisely, they suggested using gaze estimate, motion feature (*i.e*. optical flow), and depth feature (*i.e*. content depth map) besides a scene image and a text description. It demonstrates noticeable performance improvement by leveraging those new input sources mentioned above. However, not all of these additional input sources can be obtained in all use cases, e.g., we do not have motion features for a still image, and estimating depth from the target image itself is a challenging task [14].

In this paper, we focus on using the gaze signal to improve the object referring performance. We suggest using

* Equal contribution.
Work performed during internship of Jianhang Chen with Amazon.

the gaze signal because it is very natural for a person to look at an object when he/she refers to it, and the gaze signal is much easier to obtain in many real-world applications. For example, many recent smartphones, smart TVs and smart speakers come with built-in camera(s), and one can easily obtain the scene image, user's language description and the gaze image simultaneously. Fig. 1 shows an example of how the gaze signal helps improve the overall user experience – one does not have to provide a detailed description in order to refer to a specific object in the scene. Instead, a short, possibly incomplete, description like *show me the bag* is sufficient for the *ObjRef* task when gaze information is available. Adding the gaze information can largely improve both the object referring accuracy and the user experience.

In this paper, we propose a new one-stage object detection-based *ObjRef* framework with gaze assistance. As opposed to the traditional object proposal based *ObjRef* solution, which proposes a list of object candidates from a given content image and then selects the one best matching other inputs, our new framework directly generates dense anchor boxes and selects the best one by incorporating both language and gaze information in one single-stage. In this way, we are not only able to simplify the training and inference processes but also overcome the dilemma of determining the number of the object proposals in the traditional solution [31] – one may miss a referred object if the number is too small; one may pay a high computational cost if the number is too large; the appropriate number of the proposals varies for different samples. A comparison between [31] and our one-stage model can be found in Tab. 1 (more details see Sec. 4).

In summary, we make the following key contributions

- We extend the one-stage object detector model to a gaze-assisted *ObjRef* system by incorporating the language description and a gaze heatmap.
- We develop a simple but effective method to estimate the gaze heatmap with the gaze prediction result and a position encoding map.
- The proposed method significantly outperforms the state-of-the-art *ObjRef* with gaze system [31] by 7.8% for Acc@1 in the CityScapes-OR dataset.
- Our experiments show that the gaze information helps the most when 1) the gaze is reliable, and 2) the language description is ambiguous.

2. Related Work

Our paper is relevant to research in gaze estimation, object referring and object detection.

2.1. Gaze Estimation

There are a wide variety of gaze estimation methods available. According to different device and setup requirements, we can roughly distinguish different gaze estimation methods into three categories: person-independent model, person-specific model and person-specific model with a special device. Person-independent model requires collecting large-scale gaze datasets from multiple people [7, 12, 13, 33]. It has good generalization ability but might not be very accurate for one specific person [2]. The person-specific model [23], on the other hand, requires collecting training samples from the specific person. It can achieve good accuracy on that specific person, but it may not generalize to other people. For applications requiring even higher accuracy, additional eye tracker devices can be used [9]. However, separate eye-tracking devices are generally costly comparing to a regular webcam.

2.2. Object Detection

Modern deep learning-based object detectors often take the two-stage or one-stage design. In the two-stage object detector [28], the first stage is to generate a sparse set of object candidates (Region Proposal) to remove easy background candidates quickly. The later stage is to categorize the remaining candidates to the background or one of the foreground classes and refine the position of the bounding box. The one-stage detector [21], on the other hand, directly generates dense candidates for detection. Due to the additional candidate generation stage, the two-stage detector is often slower than the one-stage detector. To address the high imbalance distribution between object anchors and background anchors, Lin *et al*. [18] proposed the focal loss. The one-stage detector now has comparable or even better performance than the two-stage detector [18, 27]. In this paper, we use a one-stage object detector with position encoding maps to solve the object referring problem.

2.3. Object Referring

The target of *ObjRef* is given an image and a language description as the input to localize the object referred by the language [32]. *ObjRef* generally contains two steps: 1) generating object candidates by using region proposal network or object detector [11,20], and 2) matching the feature of the candidate with the feature of the language description. Similar to the two-stage object detector, the two-stage *ObjRef* system is not computationally efficient too. Zhou *et al*. [34] proposed to use the one-stage structure for *ObjRef* by constructing an attention map using image and language features. Liao *et al*. [16] also proposed a one-stage correlation filtering method for mapping between language domain and visual domain. Sadhu *et al*. [29] proposed ZSGNet to combine the detector network and the *ObjRef* in a single stage. These papers do not address how to incorporate supplementary information from other modalities such as gaze — we address that in this paper, which is one of our key areas of novelty.

Methods	Design	Object Candidates	Input Sources					Acc@1 (%)
			Scene	Lang.	Gaze	Opt.Flow	Depth	
[31]	two-stage	Sparse	✓	✓	✓	✓	✓	47.0
			✓	✓	✓	✓		44.2
			✓	✓	✓			41.5
Ours	one-stage	Dense	✓	✓	✓			**54.8**

Table 1. Comparison between the SOTA model [31] and the proposed model. "Acc@1" is on the CityScapes-OR dataset [31].

2.4. Gaze for Object Referring

There are a number of recent papers that start to focus on understanding how people use gaze in daily communication for *ObjRef* . One of the most popular research topics is to detect gaze target when the target and the face are in the same image [3,4,10,15,26]. These methods are solely based on image information. Vasudevan *et al*. [31] proposed to combine the gaze signal, the language signal as well as other modalities to determine which target the person is referring to when looking at the image. However, as mentioned in the introduction part, it suffers from the two-stage design. In this paper, we propose a one-stage network to deal with the *ObjRef* problem with the gaze as an auxiliary input.

3. Methodology

3.1. Overview

Let us assume a person is looking at an image showing on the screen and referring to an object in the image with gaze and natural language description. Given the image shown on the screen (scene image, $\mathbf{I}_s$), an image of the face of the person at that time (gaze image, $\mathbf{I}_g$), and the text of the language description (description, $\mathbf{d}$), the goal of *ObjRef* with gaze is to localize which object in the scene image the person is referring to. Specifically, we are looking at finding out the bounding box of the object.

The overall framework of the proposed method is shown in Fig. 2. The framework has three main branches. The top branch of the network is a natural language processing module $f_d(\cdot)$ that takes the description $\mathbf{d}$ as input and outputs a language feature $f_d(\mathbf{d})$. The bottom branch is a gaze estimation network that takes a gaze image $\mathbf{I}_g$ as input and outputs a two-dimension coordinate (x, y) indicating where in the image the person is looking at.

The center part is the scene image processing network $f_s(\cdot)$, which is an extension of the one-stage object detector – RetinaNet. It takes the scene image $\mathbf{I}_s$ as input and first produces a feature pyramid. The feature pyramid defines a dense anchor box set. The proposed network enhances each layer in the feature pyramid with both the language feature and a gaze heatmap (constructed by the gaze prediction result and an anchor position encoding map) and reproduces a new enhanced feature. The enhanced feature will be sent to a classification branch and a regression branch to predict the final bounding box by selecting the "best" anchor box. Before introducing the architecture of the proposed network, let's first review the structure of RetinaNet.

3.2. Revisiting RetinaNet

The RetinaNet model [18] is one of the most popular one-stage object detectors in recent years. The model is a unified network that consists of three major modules: a feature extraction backbone, a classification sub-network and a regression sub-network.

The RetinaNet chooses Feature Pyramid Network (FPN [17]) as the feature extraction network. FPN is an extension of the standard CNN – it constructs a multi-scale feature pyramid from the original convolutional feature map in a top-down manner. Different layers in the feature pyramid, with different sizes of receptive fields, are used to detect objects with different scales. RetinaNet extracts the feature pyramid from $P_3 - P_7$, where P_l means the feature map with the width and height of $1/2^l$ of the width and height of the original image.

Dense anchors are defined on top of the feature pyramid. We assume that feature map P_l has a resolution of $W_l \times H_l$ and a channel depth of C_l, where W_l and H_l are the width and height of P_l respectively. Each location (x, y) in P_l has a number of A anchors with different scales and aspect ratios on a base size of $2^l \times 2^l$, where $0 \leqslant x < W_l$ and $0 \leqslant y < H_l$. On top of the feature pyramid, a classification sub-network is used to determine the object category each anchor belongs to; and a bounding box regression sub-network is used to refine the location and the size of the anchor box.

The classification sub-network is a small Fully Convolutional Network (FCN) [22] that contains multiple convolutional layers. Taking feature map P_l of size $[W_l, H_l, C_l]$ as input, the output of the classification sub-network has a dimension of $[W_l, H_l, K * A]$, where K is the number of object categories, and A is the number of anchors per location. Note the output of the classification sub-network for this feature map has the same width and height as the corresponding feature map. The output indicates the object category of the corresponding anchor.

Similar to the classification sub-network, the regression sub-network is also an FCN. The regression sub-network output has a dimension of $[W_l, H_l, 4 * A]$, indicating the

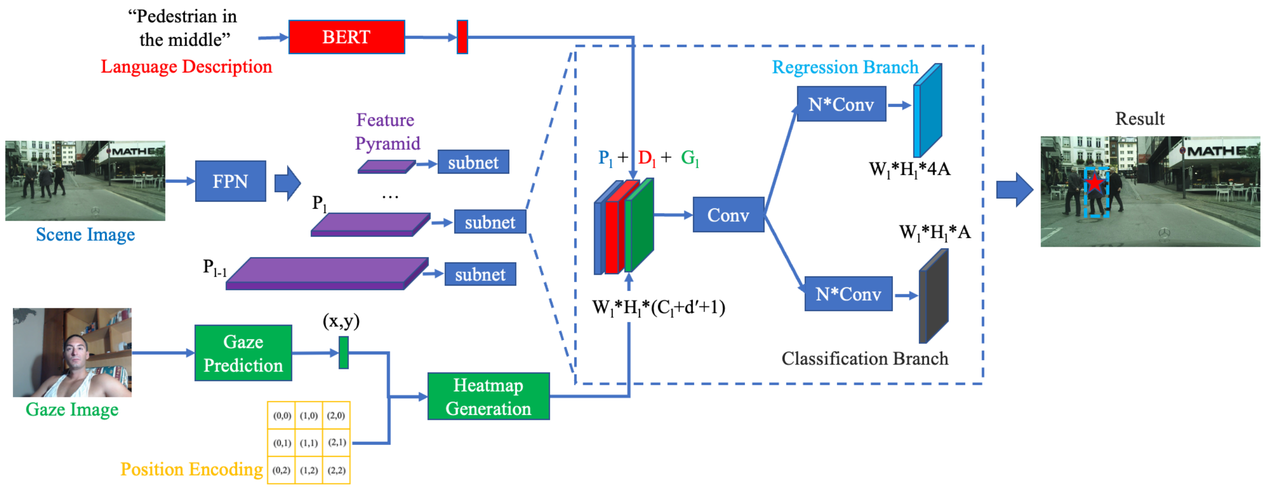


Figure 2. The proposed one-stage *ObjRef* gaze-assisted network. The network contains three branches: language processing (red), scene image processing (blue) and gaze image processing (green). The network is based on a one-stage object detector RetinaNet, and we extend it by incorporating the language feature and the gaze information into the feature pyramid. To efficiently use the gaze signal, we build a gaze heatmap with the gaze prediction and an anchor position encoding map. In the figure, P_l is the feature map from the feature pyramid with a channel depth of C_l, W_l and H_l are the width and height of P_l. D_l is the feature map from the language descriptor with size $[W_l, H_l, d']$. G_l is the gaze heatmap with size $[W_l, H_l, 1]$. A is the number of anchors per location. The blue box shows the referred person (object), and the red star shows the gaze location.

offset of the center and size between the anchor box and the predicted bounding box.

During the model inference stage, Non-Maximum Suppression (NMS) is applied to the anchor level prediction result to get the final bounding box prediction.

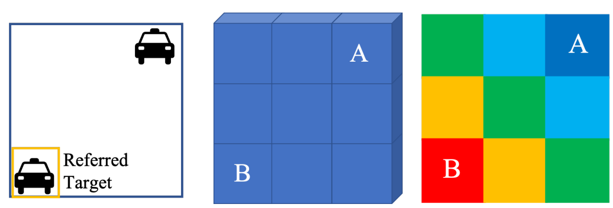
3.3. From Object Detection to Object Referring

Object detection targets finding out all objects appearing in the image, no matter where they are. For example, in Fig. 3a, there are two cars in the image and the object detector wants to detect both cars. Considering that, we want the feature map (Fig. 3b) at corresponding locations (A and B) to be similar. Object detector achieves that by FCN, which applies the same operation (convolution with the same kernel) to all locations in the image.

However, for *ObjRef* with gaze, we want to select the object that is close to the gaze point. Considering the two cars in Fig. 3a, if the person is looking at the car at the bottom left corner, the *ObjRef* system only wants to select that car and ignores the one at the top-right corner. Intuitively, we can build a heatmap (Fig. 3c) into the feature pyramid to highlight the feature close to the gaze point and suppress the feature far from the gaze point.

3.4. One-Stage Object Referring with Gaze

We extend the RetinaNet object detector for the *ObjRef* with gaze problem. The proposed network starts from the



(a) Image (b) Detector Feature (c) Gaze Heatmap

Figure 3. Incorporate gaze signal into the detector feature with a heatmap. Fig. 3a shows the original image with two cars. Fig. 3b shows the corresponding detector feature. Since the detector wants to detect all the cars, the corresponding feature A and B should be similar. Fig. 3c: to use the gaze signal, we want to build a heatmap (warmer color means higher value) to highlight feature B and suppress feature A.

FPN. We want to enhance the original FPN feature with the language feature and the gaze heatmap. The enhancement is the same for all layers in the FPN. Without loss of generality, we detail the enhancement process for layer P_l.

We start with the gaze heatmap. Assuming P_l has a size of $[W_l, H_l, C_l]$ and the predicted gaze coordinate is $\mathbf{g} \in \mathbb{R}^2$. To build the heatmap, we first construct a position encoding map E_l with size $[W_l, H_l, 2]$, which marks the position of each anchor [29]. Each point (x, y) in the position encoding map E_l is defined as:

$$E_l(x,y) = [\frac{x+\frac{1}{2}}{W_l},\ \frac{y+\frac{1}{2}}{H_l}],\quad 0 \leqslant x < W_l,\ 0 \leqslant y < H_l \tag{1}$$

For the gaze position $\mathbf{g}$, we first normalize both dimensions using the width and height of the scene image respectively and get a normalized position $\mathbf{g}'$. Note that the normalized position $\mathbf{g}'$ and the position encoding map E_l are now in the same normalized coordinate. Therefore, the gaze heatmap G_l can be defined as:

$$G_l(x,y) = 1 - \frac{\parallel E_l(x,y) - \mathbf{g}' \parallel_2}{\sqrt{2}},\quad 0 \leqslant x < W_l,\ 0 \leqslant y < H_l \tag{2}$$

where $\parallel \cdot \parallel_2$ is the ℓ_2 norm and $\sqrt{2}$ is the longest possible distance between two points in the normalized coordinate. The closer the anchor center to the gaze point, the larger the value, and the value is normalized to $[0,1]$. G_l has a dimension of $[W_l, H_l, 1]$.

Assuming the language descriptor is $f_d(\mathbf{d}) \in \mathbb{R}^{d'}$, we first duplicate and expand it to a feature map D_l such that it has the same width and height as P_l. The size of D_l is $[W_l, H_l, d']$.

The enhanced feature map P_l' can be derived as

$$P_l' = f'([P_l, D_l, G_l]) \tag{3}$$

where $[\cdot]$ means concatenation and $f'(\cdot)$ is a 1×1 convolutional layer, which maps the concatenated feature of size $[W_l, H_l, C_l + d' + 1]$ to $[W_l, H_l, C_l]$. Note that P_l' and P_l have the same size.

On top of the feature map, there are one classification sub-network and one regression sub-network. For the classification sub-network, instead of determining which object category the anchor belongs to, it only makes a binary decision: whether the anchor is the referring target or not. The regression sub-network is the same as the one in the object detector, which is to calculate the offset of the bounding box.

3.5. Loss Function and Model Inference

Following [18], we also apply the focal loss for the classification sub-network and the smooth ℓ_1 loss for the regression sub-network.

During inference time, unlike object detection, *ObjRef* only needs to select the target with the highest score. Therefore, instead of using NMS, we just simply select the bounding box with the highest score. We can also apply NMS and then set a threshold to get multiple targets.

3.6. Language Feature and Gaze Estimation

The model can be trained end-to-end with all the modalities (language, scene image, and gaze image). However, since learning the language feature and estimating the gaze position are not the major focus of the paper, we pretrained the language and gaze models and fixed the models during *ObjRef* model training (to reduce the computational complexity). For the language part, we use a pretrained BERT-base model [6] to extract the language feature. For the gaze estimation part, following [31], we apply the gaze estimation model used in [13]. To train the model, we use the bounding box center of the referred object as the target.

3.7. Extension to Two-Stage Models

Although the main focus of the paper is the one-stage model, the proposed idea of feature fusion and gaze heatmap can be easily extended to a two-stage model. For example, for Faster-RCNN [28], the region proposal network will generate multiple regions of interest (ROIs) as candidates. For each ROI, we also calculate the attention value between the center of the ROI and the predicted gaze. The concatenated vector consisting of ROI feature, language feature and attention value is fed into the last fully connected layer.

4. Experiment

4.1. Dataset

We run *ObjRef* with gaze experiment on the CityScapes-OR dataset [31], which is an enhancement of the CityScapes dataset [5].

CityScapes: The CityScapes dataset [5] contains 5,000 videos captured with a car-mounted camera from 50 cities in Europe. The resolution of the video is 2048×1024.

CityScapes-OR: The CityScapes-OR dataset [31] is a multi-modality dataset with both the scene video and the gaze video. The scene video is sampled from the CityScapes dataset. For each scene video, the annotator is asked to annotate the bounding box for a few objects in the video and give a comprehensive description for each object. The average length of the description is 15.59 words. Then the annotator is asked to watch the annotated object in the scene video when the camera is recording the audience video. The recording of the face of the annotator is treated as the gaze video for the target object. There are a total of 30,000 annotated objects. Each object has one corresponding gaze video. Following the experiment setting in [31], the dataset is split into training (19,375 objects), validation (2,977 objects) and test (7,757 objects) subsets.

Although this is a video dataset, to reduce the computation cost, we only use the 30th frame in the video for both training and testing, which is the frame where the object gets annotated. By only using one single frame, the proposed method already outperforms the video-based state-of-the-art method [31].

4.2. Implementation Details

We use the pretrained BERT-based model to extract the feature of the language description. Given a natural language description as input, the BERT model outputs a 768-dimension pooled sentence feature. For gaze estimation, we train the iTracker network [13] to predict the gaze point on the scene image. The center of the referred object is used as the training target, and the loss function is the mean squared error. During training, we use the SGD optimizer with a learning rate of 0.0001 and a batch size of 128.

For the scene image, we choose ResNet-50 as the backbone of the RetinaNet model. The RetinaNet model is pretrained with the COCO dataset [19] with 5 layers of feature pyramid ($P_3 \rightarrow P_7$). As mentioned in Sec. 3.4, the scene image feature will be enhanced by the language feature and the gaze heatmap. The enhanced feature map has a channel depth of 256, the same as the original pyramid feature. The classification branch and the bounding box regression branch on top of the feature map are the same as [18].

The scene image is resized to 960×480. During training, the center of the bounding box (normalized) is utilized as the pseudo gaze prediction. For data augmentation, the pseudo gaze prediction point is randomly shifted by (x, y), where $-0.2 \leqslant x \leqslant 0.2$, and, $-0.2 \leqslant y \leqslant 0.2$. This makes the model robust to different gaze estimation methods. We use Adam optimizer with a learning rate of 0.0001 for model training. And the training batch size is 20. A total of 4 Titan V100 GPUs are used. We learn the network for at most 100 epochs. The best model is chosen based on the performance on the validation set. During inference time, the predicted gaze from the trained gaze model is used.

4.3. Baseline and Evaluation Metric

We evaluate the performance of *ObjRef* using Accuracy@1 (Acc@1). Given the input scene image, the corresponding object description and the gaze image, only one object bounding box is predicted. If the Intersection over Union (IoU) between the predicted and the ground-truth bounding box is larger than 0.5, the prediction is a true detection. Acc@1 is defined as the percentage of the prediction being a true detection.

As far as we know, Object Referring with Gaze (ORG [31]) is the only method using gaze for *ObjRef*. Besides that, we also compared several language-based *ObjRef* methods: SimModel [25], MCB [8] and NLOR [11].

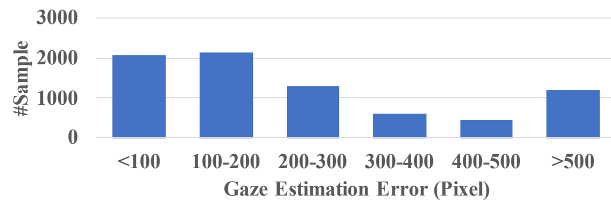


Figure 4. The distribution of the number of test samples with different gaze estimation errors. The resolution of the scene image is 2048×1024.

Modalities	Methods	w/o Gaze	w/ Gaze
S + D	SimModel [25]	35.6	-
	MCB [8]	33.4	-
	NLOR [11]	36.9	-
	ORG [31]	38.6	41.5
	Ours FRCNN	43.8	50.0
	Ours RetinaNet	**52.1**	**54.8**
S + D + O	ORG [31]	42.5	44.2
S + D + O + Dep	ORG [31]	43.8	47.0

Table 2. Acc@1 result with and without Gaze in CityScapes-OR dataset. S: Scene Image, D: Language Description, O: Optical Flow, Dep: Depth.

4.4. Result

4.4.1 Gaze Accuracy

We first evaluate the performance of our gaze estimator. The gaze prediction error is defined as the ℓ_2 distance between the predicted gaze point and the center of the referred object. The resolution of the scene image is 2048×1024. The distribution of estimation error is shown in Fig. 4, with an average error of 271 pixels.

4.4.2 Gaze for Object Referring

We show the performance of our methods and baseline methods in Tab. 2. "Ours RetinaNet" is the proposed one-stage *ObjRef* model and "Ours FRCNN" applies the proposed idea to a two-stage object detector (FRCNN, see Sec. 3.7). The proposed one-stage *ObjRef* system (Ours RetinaNet) is a clear winner overall, considering all the models using only the scene image (S), the language description (D) and the Gaze. The proposed model outperforms the previous state-of-the-art by 13.3%. Even compared to the previous state-of-the-art with two additional modalities (O: Optical flow, and Dep: Depth), the proposed method also wins by 7.8%. Compared to the two-stage model with the proposed fusion innovation (Ours FRCNN), the one-stage model outperforms by 4.8%. These numbers clearly show the outstanding performance of the proposed one-stage *ObjRef* architecture.

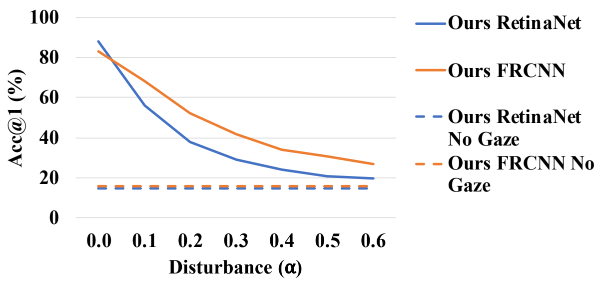


Figure 5. *ObjRef* performance with simulated gaze signal. X-axis: disturbance parameter α, Y-axis: Acc@1. No language description is used.

By comparing the "w/o Gaze" column to the "w/ Gaze" column, we find that the gaze information helps improve the *ObjRef* performance for all the models. It proves the usefulness of the gaze signal in the *ObjRef* problem. Interestingly, although using a similar gaze fusion method, the performance gains of RetinaNet and FRCNN by gaze are very different. For FRCNN, the performance gain is 6.2% (43.8% $\rightarrow$ 50.0%). While for RetinaNet, the performance gain is "only" 2.7% (52.1% $\rightarrow$ 54.8%). One possible reason is that for the two-stage model, the object proposal step helps remove a lot of candidates, such that the remaining bounding box is very sparse. Therefore, even if the gaze is not very accurate, the model can still figure out the right bounding box. However, for the one-stage model, since the model is creating dense anchor boxes, the error in the gaze estimation often leads to incorrect selection of the anchor.

4.4.3 Impact of Gaze Prediction Accuracy

We conduct experiments to show how the accuracy of the gaze prediction affects the performance of *ObjRef* . To separate the impact of the language description, we only take the scene image and gaze image as input in this experiment.

Instead of using the predicted gaze as the gaze signal, we use the center of the object bounding box (normalized) as the oracle gaze signal in both training and testing. To simulate gaze estimators with different accuracy numbers, an offset/disturbance (x, y) is added to the center point, where x and y are uniformly randomly sampled from $[-\alpha, \alpha]$ and α is chosen from $\{0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6\}$, where 0 means a perfect gaze estimator and a larger α means a worse estimator. The gaze point gets clipped at the boundary of the scene image.

The result is shown in Fig. 5. We can clearly see that, for both models, the more accurate the gaze estimator, the better the final performance. However, we can also find that a perfect gaze prediction does not always predict a perfect *ObjRef* result. The main reason is that the final result is a combined prediction of multimodality input. The prediction result will be false when the predicted size of the bounding box is inaccurate even if the center position is correct. Also, the performance drops drastically once the gaze estimation is unreliable. As expected, since RetinaNet generates dense anchors, it suffers more once the gaze prediction is unreliable.

Original Description	Extracted Tuple
A white *van* in front of us *is parked* on the left side of the road near the yellow building.	Van is parked.
A tall *tree* in front of us *is standing tall* on left side of the road.	Tree is standing tall.
A *man in black dress* walking in front of a building on the right side of the road.	Man is in black dress.

Table 3. Examples of original descriptions in CityScapesOR and corresponding extracted tuples.

Modalities	Methods	w/o Gaze	w/ Gaze
S (No Description)	Ours FRCNN	15.9	**31.4**
	Ours RetinaNet	14.9	29.2
S + D (Tuple)	Ours FRCNN	36.1	**48.8**
	Ours RetinaNet	40.5	46.9
S + D	Ours FRCNN	43.8	50.0
	Ours RetinaNet	52.1	**54.8**

Table 4. Acc@1 result with different language descriptions. S: Scene Image, D: Language Description, D (Tuple): Tuple extracted from the description.

4.4.4 Impact of Ambiguous Description

In the CityScapes-OR dataset, the language description is designed to be informative and unambiguous for the observers. It may not be the case in the real world. To show how the gaze information works with ambiguous description, we use the Stanford Open Information Extraction package [1] to extract relation tuples from the original detailed description. Some examples of the original description and its extracted tuple are shown in Tab. 3.

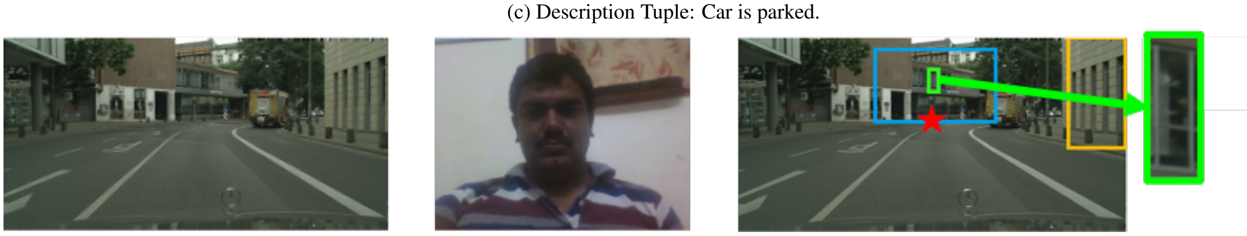
The experiment result is shown in Tab. 4. The model trained with the extracted tuple is listed as "S + D (Tuple)". We also remove the description and only use the scene image and the gaze image for *ObjRef* . The setting is listed as "S (No Description)". We observe two things: 1) the less ambiguous the language, the better the OR performance; And 2) the more ambiguous the language, the more gaze information helps.

Overall, from Sec. 4.4.3 and Sec. 4.4.4, we can see that, with accurate gaze estimation and clear language description, the RetinaNet gives better performance. However, for inaccurate gaze and ambiguous description, the two-stage FRCNN may perform better.

(a) Description: A huge tree is standing on the right side of the road.

(b) Description Tuple: Car is stopping for traffic signal.

(c) Description Tuple: Car is parked.



(d) (Failed) Description: A white window on left side of a building right in front of us.

Figure 6. Qualitative result for *ObjRef* with gaze. The first column shows the scene image, the second column shows the gaze image, and the final column shows the referring result and the ground truth. The green box shows the ground-truth, the yellow box shows the result for our *ObjRef* method without gaze, the blue box shows the result for our *ObjRef* method with gaze, and the red star shows the predicted gaze location. Small objects are highlighted.

4.4.5 Qualitative Result

We show three successful results (Fig. 6a (with full description), 6b and 6c (with extracted tuple)) and one failed result (Fig. 6d) in Fig. 6. The most interesting case is Fig. 6b. The description is "Car is stopping for traffic signal". There is a car close to the camera (to the left) and several cars on the opposite side of the crossing. And the audience is looking at the car to the left. Without gaze information, the model finds the car on the opposite side (yellow box). However, with the gaze information, the model can successfully find out the car to the left. For the failure case in Fig. 6d, it might be due to the model misunderstands the subject of the sentence.

5. Conclusion

Object referring is an important topic for computer vision and natural language understanding. In this paper, we systematically study how adding the additional gaze signal can improve the performance of object referring. Unlike the previous state-of-the-art method, which requires a separate object proposal step, the proposed method directly utilizes the one-stage object detector to find the referred target. We propose to build a gaze heatmap via anchor box position encoding and the predicted gaze to help select the object close to the predicted gaze. The proposed method achieves significantly better performance compared to the state-of-the-art method in the CityScapes-OR dataset. We further show that the gaze signal helps more when 1) the gaze prediction is accurate, and 2) the language is ambiguous.

Our future work includes improving the performance of the gaze prediction, and better integration of the gaze signal into the object referring system. We would also like to extend this framework to handle other types of modalities that can be used to localize a region in an image, e.g., gestures. Finally, we would like to explore model accuracy/latency/size trade-offs when we have to consider on-device compact versions of these models.

References

[1] Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D Manning. Leveraging linguistic structure for open domain information extraction. In *Proceedings of the Association of Computational Linguistics*, 2015. 7

[2] Dario Cazzato, Marco Leo, Cosimo Distante, and Holger Voos. When I look into your eyes: A survey on computer vision contributions for human gaze estimation and tracking. *Sensors*, 20(13):3739, 2020. 2

[3] Eunji Chong, Nataniel Ruiz, Yongxin Wang, Yun Zhang, Agata Rozga, and James M Rehg. Connecting gaze, scene, and attention: Generalized attention estimation via joint modeling of gaze and scene saliency. In *ECCV*, 2018. 3

[4] Eunji Chong, Yongxin Wang, Nataniel Ruiz, and James M. Rehg. Detecting attended visual targets in video. In *CVPR*, 2020. 3

[5] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016. 5

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019. 5

[7] Tobias Fischer, Hyung Jin Chang, and Yiannis Demiris. RT-GENE: Real-time eye gaze estimation in natural environments. In *ECCV*, 2018. 2

[8] Akira Fukui, Dong Huk Park, Daylen Yang, Anna Rohrbach, Trevor Darrell, and Marcus Rohrbach. Multimodal compact bilinear pooling for visual question answering and visual grounding. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2016. 6

[9] Agostino Gibaldi, Mauricio Vanegas, Peter J Bex, and Guido Maiello. Evaluation of the Tobii EyeX eye tracking controller and Matlab toolkit for research. *Behavior research methods*, 49(3):923–946, 2017. 2

[10] Jian Guan, Liming Yin, Jianguo Sun, Shuhan Qi, Xuan Wang, and Qing Liao. Enhanced gaze following via object detection and human pose estimation. In *Proceedings of the International Conference on Multimedia Modeling*, 2020. 3

[11] Ronghang Hu, Huazhe Xu, Marcus Rohrbach, Jiashi Feng, Kate Saenko, and Trevor Darrell. Natural language object retrieval. In *CVPR*, 2016. 2, 6

[12] Petr Kellnhofer, Adria Recasens, Simon Stent, Wojciech Matusik, and Antonio Torralba. Gaze360: Physically unconstrained gaze estimation in the wild. In *ICCV*, 2019. 2

[13] Kyle Krafka, Aditya Khosla, Petr Kellnhofer, Harini Kannan, Suchendra Bhandarkar, Wojciech Matusik, and Antonio Torralba. Eye tracking for everyone. In *CVPR*, 2016. 2, 5, 6

[14] Hanhan Li, Ariel Gordon, Hang Zhao, Vincent Casser, and Anelia Angelova. Unsupervised monocular depth learning in dynamic scenes. In *the 4th Annual Conference on Robot Learning*, 2020. 1

[15] Dongze Lian, Zehao Yu, and Shenghua Gao. Believe it or not, we know what you are looking at! In *ACCV*, 2018. 3

[16] Yue Liao, Si Liu, Guanbin Li, Fei Wang, Yanjie Chen, Chen Qian, and Bo Li. A real-time cross-modality correlation filtering method for referring expression comprehension. In *CVPR*, 2020. 2

[17] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017. 3

[18] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017. 2, 3, 5, 6

[19] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, 2014. 6

[20] Jingyu Liu, Liang Wang, and Ming-Hsuan Yang. Referring expression generation and comprehension via attributes. In *ICCV*, 2017. 2

[21] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. SSD: Single shot multibox detector. In *ECCV*, 2016. 2

[22] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 3

[23] Seonwook Park, Shalini De Mello, Pavlo Molchanov, Umar Iqbal, Otmar Hilliges, and Jan Kautz. Few-shot adaptive gaze estimation. In *ICCV*, 2019. 2

[24] Siddharth Patki, Andrea F Daniele, Matthew R Walter, and Thomas M Howard. Inferring compact representations for efficient natural language understanding of robot instructions. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2019. 1

[25] Bryan A. Plummer, Arun Mallya, Christopher M. Cervantes, Julia Hockenmaier, and Svetlana Lazebnik. Phrase localization and visual relationship detection with comprehensive image-language cues. In *ICCV*, 2017. 6

[26] Adria Recasens, Carl Vondrick, Aditya Khosla, and Antonio Torralba. Following gaze in video. In *ICCV*, 2017. 3

[27] Joseph Redmon and Ali Farhadi. YOLOv3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. 2

[28] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. *NeurIPS*, 2015. 2, 5

[29] Arka Sadhu, Kan Chen, and Ram Nevatia. Zero-shot grounding of objects from natural language queries. In *ICCV*, 2019. 2, 4

[30] Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. VL-BERT: Pre-training of generic visual-linguistic representations. In *ICLR*, 2020. 1

[31] Arun Balajee Vasudevan, Dengxin Dai, and Luc Van Gool. Object referring in videos with language and human gaze. In *CVPR*, 2018. 1, 2, 3, 5, 6

[32] Licheng Yu, Zhe Lin, Xiaohui Shen, Jimei Yang, Xin Lu, Mohit Bansal, and Tamara L Berg. MAttNet: Modular attention network for referring expression comprehension. In *CVPR*, 2018. 1, 2

[33] Xucong Zhang, Yusuke Sugano, Mario Fritz, and Andreas Bulling. MPIIGaze: Real-world dataset and deep appearance-based gaze estimation. *IEEE TPAMI*, 41(1):162–175, 2017. 2

[34] Yiyi Zhou, Rongrong Ji, Gen Luo, Xiaoshuai Sun, Jinsong Su, Xinghao Ding, Chia-wen Lin, and Qi Tian. A real-time global inference network for one-stage referring expression comprehension. *arXiv preprint arXiv:1912.03478*, 2019. 2