Searching for Ambiguous Objects in Videos using Relational Referring Expressions

Hazan Anayurt*
hazan.anayurt@metu.edu.tr
Sezai Artun Ozyegin*
artun.ozyegin@metu.edu.tr
Ulfet Cetin
ulfet.rwth@gmail.com
Utku Atkas
utkuaktascs@gmail.com
Sinan Kalkan
skalkan@metu.edu.tr

Department of Computer Engineering,
Middle East Technical University,
Ankara, Turkey

* Equal contribution

Abstract

Humans frequently use referring (identifying) expressions to refer to objects. Especially in ambiguous settings, humans prefer expressions (called relational referring expressions) that describe an object with respect to a distinguishing, unique object. Unlike studies on video object search using referring expressions, in this paper, our focus is on (i) relational referring expressions in highly ambiguous settings, and (ii) methods that can both generate and comprehend a referring expression. For this goal, we first introduce a new dataset for video object search with referring expressions that includes numerous copies of the objects, making it difficult to use non-relational expressions. Moreover, we train two baseline deep networks on this dataset, which show promising results. Finally, we propose a deep attention network that significantly outperforms the baselines on our dataset. The dataset and the codes are available at https://github.com/hazananayurt/viref.

1 Introduction

A referring expression (RE) is a description that identifies an object among many candidates. Humans are likely to use (non-relational) RE that include absolute attributes such as color or relative attributes such as size especially when there are no objects with the same attributes [7]. However, when there are ambiguities e.g. owing to having multiple instances of the same object categories or multiple objects having the same attributes, humans prefer relational RE [7, 15], i.e., descriptions that identify an object with respect to other objects.

In the past few years, there have been many successful studies on REs for objects in images [4, 7, 8, 17]. These studies generally focus on two main tasks: generation and comprehension. In a generation task, the aim is to generate an RE given an image and an object whereas, in comprehension task, the aim is to find the object given an image and an RE.

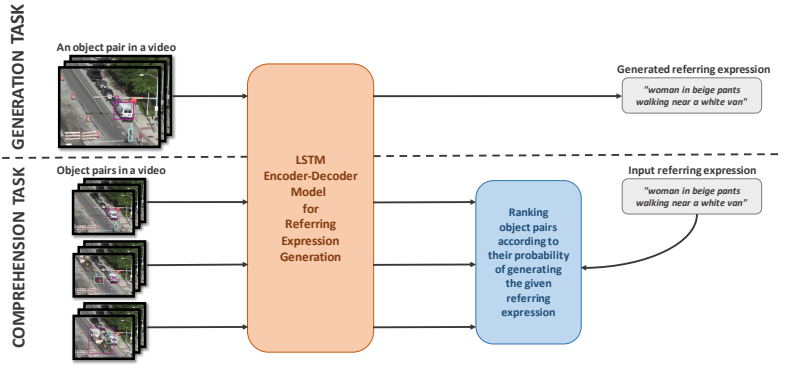


Figure 1: An overview of the generation and comprehension tasks performed by our model.

Recently, a few studies have addressed RE comprehension for object search in videos [1, 5, 16]. However, these studies (i) only focused on the comprehension of REs, and (ii) used datasets and REs that did not necessarily require relational attributes since the level of ambiguities was low.

In this work, we collected a dataset for REs describing an object using another (context) object. This means that every referring expression includes two objects: the main object and the context object. We collected the REs on the objects from the VIRAT (Video Image Retrieval and Analysis Tool) dataset [9], which is a surveillance video dataset mainly containing people and vehicles, and the subset of the ILSVRC2015 dataset [12] which contains vehicles.

Using the dataset we collected, we were able to train a model that recognizes the relationship between two objects including the classes and physical appearances of the objects and the actions they are performing. To achieve this, we used a model similar to the one used in [7] for RE comprehension and generation on images. The idea behind our model is first training an LSTM generator that, given two objects in a video, generates an RE describing the first object using the second object. Then, we can use this LSTM generator in the comprehension task. See Fig. 1 for an illustration.

Related Work on Referring to Objects in Images: With advances in deep learning, it has been possible to link language and vision with superior performance than before. One of the first to address such a link for referring to objects in images with textual descriptions is Kazemzadeh *et al*. [8], who collected a large-scale dataset using a game-like activity. Many following studies extended and proposed better ways to use referring expressions. E.g., Mao *et al*. [7] demonstrated that RE comprehension and generation can be addressed by a single model thanks to the generative and discriminative capabilities of a Recurrent Neural Network. In their model, an RE is generated using an LSTM [6] network that is initialized with VGG features [13] extracted from the input image. For comprehension, the same LSTM network is used to evaluate the compatibility with the features of the input image.

Nagaraja *et al*. [8] came up with another interesting approach by formulating RE generation and comprehension as a Multiple Instance Learning problem for learning RE generation with respect to a context object even when such objects are not labeled in the dataset. They do this by creating positive and negative bags from object pairs. The positive bags contain the referred object as the main object while the negative bags do not contain the referred

object at all. This way, there is no possibility that the referred pair is in the negative bag whereas it could be any of the pairs in the positive bag but which object is the context object is not known. They claim that, by modelling the context between objects, they were able to achieve a better performance than when they used only the main object's properties.

Similar to other problems linking computer vision and language, it has recently been shown that an attention capability improves the performance of RE comprehension [17]. This is a "module"-level attention model that modulates contributions from location, object and action/relation.

Related Work on Referring to Objects in Videos: Inspired from similar studies on images, recently, many have focused on using REs for objects in videos [1, 5, 16]. These studies have shown that REs in videos are more challenging since they also contain temporal information, and the dimensionality of the inputs is much higher.

Vasudevan *et al*. [1] have studied linking human gaze with REs. In their work, the gaze location is estimated from the user's eyes, and an object corresponding to an uttered RE is sought in a region around the gaze location. A study done by Khoreva *et al*. [5] has addressed the segmentation of a referred object in a video given an RE. They have evaluated their methods on video object segmentation datasets. In a more recent study by Wiriyathammabhum *et al*. [16], the MattNet model [17] was extended for REs in videos by including motion modules and attention on motion modules.

Our Contributions: Looking at the related studies, we can summarize our contributions as follows:

(1) A Dataset with Ambiguities: We introduce a new dataset that includes REs on pairs of objects. The dataset is highly ambiguous, including numerous instances of the same objects, therefore, relational REs are required.

The three studies [1, 5, 16] on using REs for objects in videos have introduced their own datasets since they all addressed a different setting and application for REs. However, these datasets are not suitable for our goals since (i) they require additional modalities like gaze or segmentation, and (ii) ambiguity in these datasets is limited and therefore, they do not focus on relational REs.

(2) A Deep Attention Model that can relate a relational RE with an object pair. Unlike other studies on REs on videos, our model can be used for both comprehension and generation tasks. Moreover, we provide two baseline models that are extensions of REs for objects in images (namely, [2]) to REs for objects in videos.

2 A Dataset of Relational Referring Expressions (REs) for Objects in Videos

For our dataset, we used videos from two video datasets: The VIRAT Ground [9] and ILSVRC2015 [12] datasets. Our dataset consists of videos and bounding boxes from these two datasets and REs we collected for object pairs in these videos. VIRAT is a video surveillance dataset that contains videos of places like streets and parking lots. Due to this, the main object classes are people and vehicles. Because we wanted to narrow our domain down to fit the majority of classes in VIRAT, we only used videos that contained vehicles from the ILSVRC dataset. Due to this, VIRAT makes up most of our videos. Other than this, we did not put a restriction on object categories or REs on them, which means that our dataset contains objects (and REs using these objects) such as garbage bins, traffic cones and objects

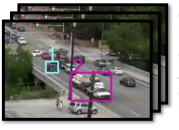
Table 1: Information about the videos in our dataset.

	Avg.	Min.	Max.
Sequence length (sec)	19.00	1	42
# objects per video	9.38	2	46
# pairs per video	19.57	2	148
# videos	125 (VIRAT) + 37 (ILSVRC)		
# objects (total)	1520		
# pairs (total)	3170		

Table 2: Information about the REs in our dataset. $\bullet$ stands for the number of words in an RE, and $\star$ denotes the number of occurences of words in the dataset.

	Avg.	Min.	Max.
RE length	11.70	5	37
# RE per object	12.51	6	90
# RE	9416 ($\bullet < 25$) + 94 ($\bullet \geq 25$)		
# words in REs	708 ($\star > 1$) + 370 ($\star = 1$)		

(i) *"a man in white shirt riding a bike and passing by a woman in white shirt"*
(ii) *"a man in white t-shirt riding bicycle beside a woman in white t-shirt"*
(iii) *"a male cyclist in white clothes passing a female pedestrian in a white blouse"*



(i) *"a black car going behind of a white truck which is carrying a load"*
(ii) *"a black car driving behind a pickup hauling a loaded flatbed"*
(iii) *"a black sedan following a white mini truck on a two-way street"*

Figure 2: Two object pairs from our dataset and the three REs collected for them.

carried by people e.g. bags or boxes.

One thing we did put a restriction on while choosing object pairs was whether the pair actually had a relation or not. This was because an RE defining one object using the other could not be written if the objects did not have any meaningful "overlap" in the video. We saw that the number of pairs we obtained were still too high even after eliminating pairs that were too far apart spatially or temporally; therefore, we visually observed each pair to decide manually whether or not a meaningful RE could be written for them. The REs we have collected for this dataset are only for the pairs that remained after both eliminations.

To collect the REs for our dataset, we used the Microworkers crowdsourcing platform. The video sequences we showed the workers were only the parts of a video starting when the first object of the pair to enter the video enters and ending when the last one of the pair to exit the video exits. Each video sequence contained an object pair with one object labeled as the first object and the other object labelled as the second and we asked for the workers to provide an RE for the first object using the second object as the context object after watching the video. Therefore, the order of the labels was important. We also reversed the order of the pair, and collected REs for the reverse order as well.

We collected six REs each object pair (three for the straight and three for the reverse order). We asked the workers to give as much detail as possible and we did not have any vocabulary or grammar restrictions apart from asking the answers to be noun phrases defining the main object w.r.t. the context object without using full sentences. Differently from the previous datasets, we did not require for the REs to be unambiguous; hence, one RE could be correctly identifying more than one pair.

Two example object pairs from our dataset along with the collected REs can be seen in Fig. 2. In Tables 1 and 2 we provide some descriptive statistics about our dataset. REs with length greater than 25 are not used during training as explained in Section 5.1. Moreover, words that occur only once are removed from our vocabulary.

3 Video Object Search using Relational Referring Expressions (VIREF)

For our model, similar to [1, 5, 16], we assume that objects have been already detected or annotated. Then, given two objects in a video, our goal is to generate the most probable referring expression defining the first object using the second object, which is called the generation task. We also use the model trained on the generation task to find the most suitable object pair among labeled objects in a video given an RE as input, which is the comprehension task.

We use a method similar to [7] and approach the generation and comprehension tasks together. We first encode the objects in videos using an LSTM, then we feed the encoded object pair to a decoder LSTM. We use the hidden states of the decoder to calculate the attention weights of the objects' features, enabling our model to attend to different combinations of features according to the word it wants to predict. After the attention, we combine the encoded features that the model has attended to and the hidden state of the decoder to calculate the probability of each word in our vocabulary to try to obtain the most probable RE given an object pair in a video. We also use these probabilities while searching for objects in a video (comprehension task). An overview of our model is provided in Fig. 3.

3.1 Generation Task

The generation task can be formally defined as finding the most likely sequence of words, $\mathbf{r} =: < w_1, ..., w_n >$, (i.e., the referring expression) from the vocabulary given a video sequence $\mathbf{v} =: < I_1, ..., I_m >$ (I_i is the i^{th} frame) with annotated boxes for the main object, $\mathbf{o}^t =: < B^t_1, ..., B^t_m >$, and the context object, $\mathbf{o}^c =: < B^c_1, ..., B^c_m >$. B is simply a vector with the coordinates of the top-left and the bottom-right corners of the bounding box.

For the generation task, we use an LSTM decoder architecture similar to [7] with two main differences, the first of which is using an encoder LSTM on the object features. This helps in adapting the model to the video domain by allowing it to capture the temporal relation between the objects more accurately. The second one is the attention module attached on top of the outputs of the decoder.

The input ($\mathbf{x}_i$) of the encoder LSTM at step i are the features extracted from the i^{th} frame of the input video (sampled each second). Denoting the VGG16 (fc1) [13] features by $\phi()$, $\mathbf{x}_i$ is a concatenation of the following:

$$\mathbf{x}_i =< \phi(I_i(B^t_i)),\ \phi(I_i(B^c_i)),\ \phi(I_i),\ \phi(M(B^t_i)),\ \phi(M(B^c_i)) >, \tag{1}$$

where $I_i(B)$ denotes an image with the pixels in bounding box B; and $M(B)$ is a binary image (with the same size as I_i) where the pixels are white inside B and black elsewhere.

The hidden state of the encoder LSTM at step (frame) i can be denoted as follows:

$$\mathbf{h}^e_i = LSTM^e(\mathbf{x}^{a_0}_i,\ h^e_{i-1}), \tag{2}$$

where $LSTM^e$ is the encoder LSTM, $\mathbf{x}^{a_0}_i$ is the i^{th} input scaled with the initial attention weights $a_0 =< a^1_0,\ a^2_0,\ a^3_0,\ a^4_0,\ a^5_0 >\in \mathbb{R}^5$ and $\mathbf{h}^e_i$ is the encoder hidden state at step i. $\mathbf{h}^e_0$ is a trainable parameter and it is initialized as 0. The attention weights are integrated as follows:

$$\mathbf{x}^{a_0}_i =< a^1_0 \times \phi(I_i(B^t_i)),\ a^2_0 \times \phi(I_i(B^c_i)),\ a^3_0 \times \phi(I_i),\ a^4_0 \times \phi(M(B^t_i)),\ a^5_0 \times \phi(M(B^c_i)) > . \tag{3}$$

The rest of our model resembles a regular image captioning architecture with the addition of a feature attention mechanism. As input to the decoder, we used GloVe [11] 50-dimensional vector representations of the words. The hidden state of the decoder at step j can be denoted as follows:

$$\mathbf{h}_j^d = LSTM^d(\psi(w_j),\, h_{j-1}^d), \tag{4}$$

where w_j is the word at step j and $\psi(\cdot)$ is the function that takes a word to its embedding. The last hidden state of the encoder is fed into the decoder as its initial hidden state, i.e., $\mathbf{h}_1^d = LSTM^d(\psi(w_{start}), h_m^e)$, where $\psi(w_{start})$ is the embedding vector of a pre-defined start token.

At each time step j of the decoder, the attention weight vector a_j is calculated using a network named Feature Attention Network (FAN) as $a_j = FAN(h_j^d)$. a_j is used as the attention weights to the features of the encoder (Fig. 3). The attention weights a_j are integrated into the inputs in the same way as shown for a_0 in Eq. 3. Then, with those weights, the encoder is run again. Combining the last hidden state of the newly run encoder and the $\mathbf{h}_j^d$, the output is calculated using the Word Estimation Network (WEN) as follows:

$$\mathbf{out}_j = WEN(h_j^d,\, LSTM^e(x_m^{a_j},\, LSTM^e(x_{m-1}^{a_j},\, \ldots))). \tag{5}$$

The output of WEN is passed through Softmax to obtain a probability distribution over the words in our vocabulary. With this, we obtain the probability of an RE, $p(\mathbf{r} \mid \mathbf{v}, \mathbf{o}^t, \mathbf{o}^c)$, by multiplying the word probabilities. In test time, we sample the most probable RE (i.e., $\arg\max_{\mathbf{r}} p(\mathbf{r} \mid \mathbf{v}, \mathbf{o}^t, \mathbf{o}^c)$) using beam search with beam size 3, similar to [7]. The illustration of the VIREF generation model can be seen in Fig. 3.

The attention mechanism we employ is similar to the modular attention mechanism employed in [16, 17]. The main differences are that attention in our model is performed over the extracted features at each time step, not over the modules that summarize information over time separately, and it is adapted to video domain by using an encoder LSTM.

As can be seen in Eq. 5, the encoder is run again for each time step of the decoder with its corresponding feature attention weights. It may be argued that this is too much computational overhead; however, in this way, the decoder can decide which input features it would attend to according to the word it wants to predict and this allows the features to interact in the encoding step. If we wanted to have feature-level attention but only ran the encoder once and attended to the encoded features, we would have to encode each feature separately. This would prevent the encoder from capturing the relation between the features such as an object and its location, or the motion of the objects with respect to each other.

3.2 Comprehension Task

For the comprehension task, we do not train a separate model but use the model we trained for the generation task. This becomes possible if we assume uniform prior for the probability of the video and the places of the objects in it (meaning they can occur anywhere in any video). Under this assumption, using the Bayes' Rule, the probability of a video and an object pair given a referring expression becomes directly proportional to the probability of a referring expression given an object pair since the evidences are the same:

$$\arg\max_{\mathbf{v},\mathbf{o^t},\mathbf{o^c}} p(\mathbf{v}, \mathbf{o^t}, \mathbf{o^c} | \mathbf{r}) = \arg\max_{\mathbf{v},\mathbf{o^t},\mathbf{o^c}} p(\mathbf{r} | \mathbf{v}, \mathbf{o^t}, \mathbf{o^c}). \tag{6}$$

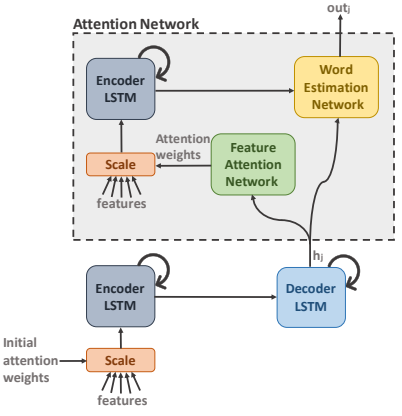


Figure 3: The VIREF model.

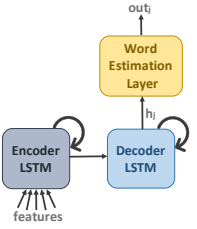


Figure 4: The VIREF-a model (VIREF without attention).

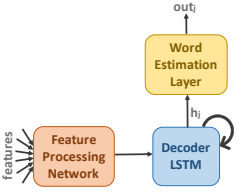


Figure 5: The VIREF-e model (VIREF without LSTM encoder).

Therefore, when an RE is taken as input, we calculate the probability of that RE for each object pair in each video, $p(\mathbf{r}|\mathbf{v},\mathbf{o^t},\mathbf{o^c})$, using the generator model as explained in Section 3.1. Then, we rank the object pairs according to the probability of them giving the input RE. Mao *et al*. [7] have used a similar approach.

3.3 Architecture and Loss

For both encoder and decoder, we use a six-layer LSTM, whereas Feature Attention Network and Word Estimation Network both consist of 3 fully-connected layers. Both the LSTMs and the fully-connected networks in the attention module use Dropout with dropping probability of 0.2. The initial attention weights are trainable parameters and all attention weights are used after being passed through the Softmax function. The input features are 4096-dimensional each and add up to 20480 when concatenated after scaling(see Eq. 3). The output of the Word Estimation Network is of size 1024, which is the size of our vocabulary, and is also passed through Softmax. For training the networks, we use cross-entropy loss.

4 Baseline Models

To compare our method to, we provide two baseline models. The architectures are similar to VIREF in certain parts with the main differences being in the encoder module. Essentially, these models are simplified versions of the VIREF model to show the performance improvements provided in both the generation and comprehension tasks by the LSTM encoder and attention modules.

VIREF without Attention (VIREF-a): VIREF-a is the same model as VIREF without the extra attention module, as shown in Fig. 4. This model can be described as a straightforward extension of the model used in [7] to the video domain, without the usage of video-related

feature extractors. The feature extractors in image domain are still more powerful than the ones in the video domain, therefore, to efficiently use the information in the time dimension, an LSTM encoder is utilized.

VIREF without LSTM encoder (VIREF-e) Another straightforward extension of [7] is directly using video feature extractors and feeding them into decoder LSTM (as illustrated in Fig. 5). Using this kind of approach would remove the necessity of using a recurrent network to capture the temporal information.

We used six features that could replace the LSTM-encoded VGG16 features. The first two are the averages of VGG16 (fc1) [13] features for the main and the context objects over time. The other four are the C3D (fc6) [14] features for the main object, the context object, and the main & context objects together (formed by blackening regions outside of their bounding boxes), and the whole scene. We believe that VGG16 features of the objects could capture the fine details of the objects and C3D features could contain motion related information.

The model takes these six features, processes them using fully-connected layers and uses the same decoder as VIREF-a. The only difference is that encoder part uses different kind of feature extraction methods instead of the LSTM encoder used in the other two models. Each of the features is of the size 4096 and the features add up to 24576 when concatenated. After the concatenation, features are passed through Feature Processing Network which consists of 3 fully-connected layers.

5 Experiments

In this section, we evaluate our method and compare it against the baselines on generation and comprehension tasks.

5.1 Training and Implementation Details

We split our dataset into training, validation and test sets respectively with 60%, 10%, 30% percentages. For training the networks, we used Adam optimizer with a learning rate of 10^{-4} and a batch size of 10. We used the validation set to perform early-stopping and observed this to be a sufficient measure against overfitting in our experiments.

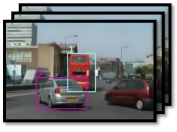
The decoder LSTMs output a probability distribution over the words in our vocabulary. The vocabulary includes (i) the words that appear at least twice in our training set (634 words), and (ii) the words most frequently used in the Google RefExp dataset [7] (386 words). In total, with <*start*>, <*end*>, <*unk*> and <*nil*> tokens, our vocabulary contains 1024 words.

During training, we use padding (the <*nil*> token) so that the referring expressions will have the same length. However, the number of very long REs only make up a very small percentage of the dataset and to be able to use less padding for evening out the length of all REs, we exclude every RE that has a length of 25 or greater in training. Out of the 9,510 REs we collected in total, only 94 exceed this length constraint and we are able to avoid higher training times at the cost of only 1% of our data.

Table 3: RE generation performance of the models. Higher is better.

Method	Average BLEU-4 Score	Average METEOR Score	# of words used in the output
VIREF-e	0.1197	0.4054	**233**
VIREF-a	0.1498	0.4580	75
VIREF	**0.2365**	**0.5492**	91

VIREF: *"a white motorcycle passed by a silver car"*
VIREF-a: *"a white van parked near a black motorcycle"*
VIREF-e: *"a white bike left behind by a silver van"*



VIREF: *"a red bus driving in front of a yellow car"*
VIREF-a: *"a white colour small truck running behind the red colour truck"*
VIREF-e: *"pass"*

VIREF: *"a black scooter being on a silver car"*
VIREF-a: *"a white colour scooter running behind the white colour suv"*
VIREF-e: *"crossed"*

VIREF: *"a man in white shirt walking behind a man"*
VIREF-a: *"a man in black shirt riding a bicycle"*
VIREF-e: *"a man in white shirt riding a bicycle passing by man in black shirt"*

Figure 6: Sample generated REs. Object with label 1 is the main object, and the one with label 2 is the context object.

5.2 Generation Results

To evaluate the results of the generator, we calculated average scores using the BLEU [10] and METEOR [6] metrics. The averaged BLEU-4 and METEOR scores of the models on the test set can be seen in Table 3. We observe that VIREF outperforms both baseline models in terms of the BLEU and METEOR metrics. However, even though VIREF is the best model in terms of BLEU and METEOR scores, it can be seen that it generates REs from a more narrow set of words compared to VIREF-e. According to our observations, VIREF-e's vast usage of different words is sometimes accompanied by the generation of REs which do not meaningfully describe an object. We believe that this has affected its average BLEU and METEOR scores. Some of the generation results of the models can be observed in Fig. 6.

5.3 Comprehension Results

The comprehension part of the model, as mentioned in Section 3.2, uses the generator model. To evaluate the comprehension model, for each video, we selected an RE from the test set and calculated a matching score with every pair in that video. We assumed that only the object pair it was written for is the correct answer. This may have, however, led to a score that is less than the actual score, since our dataset consists of ambiguous examples and an RE may actually represent more than one object pair.

We used two different retrieval task evaluation measures: mean average precision (mAP) and rank-k accuracy. AP is defined simply as $1/k$ if the correct pair is retrieved at rank k (since we have only a single "relevant document", average precision is directly equal to precision). On the other hand, rank-k accuracy is the percentage of results in which the correct pair is retrieved at top k object pairs.

Table 4 lists the comprehension results of the methods in terms of mAP and rank-1, rank-2, and rank-3 accuracies. We observe that VIREF again performs better than the other models, thanks to its attention model. See also Fig. 7 for some sample results.

Table 4: RE comprehension results. Higher is better.

Method	mAP	rank-1 accuracy	rank-2 accuracy	rank-3 accuracy
VIREF-e	0.55	0.35	0.61	0.69
VIREF-a	0.46	0.26	0.49	0.57
VIREF	**0.65**	**0.47**	**0.69**	**0.78**

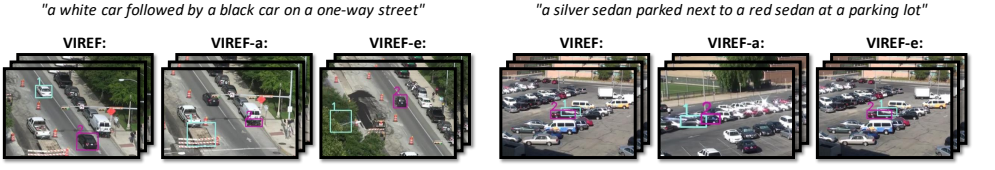


Figure 7: Sample comprehension results. Object with label 1 is the main object, and the one with label 2 is the context object.

5.4 Size and Running-time of the Models

Table 5 lists the number of parameters and the GPU time used by the models. We can see that VIREF-e has the most number of parameters since the encoding part is a fully-connected network. However, this provides a gain in terms of running speed.

Table 5: # params. and speed (avg. of 100 samples on Nvidia GeForce GTX 1060).

Method	# of parameters	Generation time (sec)	Comprehension time (sec)
VIREF-e	85.8M	0.402	0.007
VIREF-a	27.1M	0.434	0.014
VIREF	28.2M	1.403	0.252

6 Conclusion

In this work, we addressed linking objects in videos with relational referring expressions (REs). For this, we first collected a dataset of REs for a subset of videos from VIRAT and ILSVRC datasets. The videos we specifically have chosen included highly ambiguous settings with numerous occurrences of objects.

Moreover, we proposed an encoder-decoder recurrent architecture with which we can both comprehend an RE (i.e. identify the matching object pair in a video) and generate an RE identifying a pair of objects in a video. At each decoding stage, our model can attend to the features in the encoding stage. Compared with the two baselines that we have developed, our model performs significantly better in both generation and comprehension tasks.

Acknowledgments

This work was partially supported by the Scientific and Technological Research Council of Turkey (TÜBİTAK) through the project titled “Object Detection in Videos with Deep Neural Networks” (project no 117E054). The authors would like to thank Dr. Emre Akbas for discussions and suggestions on the paper.

References

[1] Arun Balajee Vasudevan, Dengxin Dai, and Luc Van Gool. Object referring in videos with language and human gaze. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4129–4138, 2018.

[2] Robert Dale. Cooking up referring expressions. In *Proceedings of the 27th Annual Meeting on Association for Computational Linguistics*, 1989.

[3] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[4] Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara Berg. Referitgame: Referring to objects in photographs of natural scenes. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 787–798, 2014.

[5] Anna Khoreva, Anna Rohrbach, and Bernt Schiele. Video object segmentation with referring expressions. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 7–12. Springer, 2018.

[6] Alon Lavie and Abhaya Agarwal. Meteor: An automatic metric for mt evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 228–231. Association for Computational Linguistics, 2007.

[7] Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L Yuille, and Kevin Murphy. Generation and comprehension of unambiguous object descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11–20, 2016.

[8] Varun K Nagaraja, Vlad I Morariu, and Larry S Davis. Modeling context between objects for referring expression understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 792–807. Springer, 2016.

[9] Sangmin Oh, Anthony Hoogs, Amitha Perera, Naresh Cuntoor, Chia-Chih Chen, Jong Taek Lee, Saurajit Mukherjee, J. K. Aggarwal, Hyungtae Lee, Larry Davis, Eran Swears, Xioyang Wang, Qiang Ji, Kishore Reddy, Mubarak Shah, Carl Vondrick, Hamed Pirsiavash, Deva Ramanan, Jenny Yuen, Antonio Torralba, Bi Song, Anesco Fong, Amit Roy-Chowdhury, and Mita Desai. A large-scale benchmark dataset for event recognition in surveillance video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3153–3160. IEEE, 2011.

[10] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 311–318. Association for Computational Linguistics, 2002.

[11] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[12] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

[13] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proceedings of 3rd International Conference on Learning Representations (ICLR)*, 2015.

[14] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 4489–4497, 2015.

[15] Jette Viethen and Robert Dale. The use of spatial relations in referring expression generation. In *Proceedings of the Fifth International Natural Language Generation Conference*, pages 59–67. Association for Computational Linguistics, 2008.

[16] Peratham Wiriyathammabhum, Abhinav Shrivastava, Vlad I Morariu, and Larry S Davis. Referring to objects in videos using spatio-temporal identifying descriptions. *arXiv preprint arXiv:1904.03885*, 2019.

[17] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.