Connecting Vision and Language with Video Localized Narratives: Supplemental Material

Paul Voigtlaender Soravit Changpinyo Jordi Pont-Tuset Radu Soricut Vittorio Ferrari
Google Research
{voigtlaender,schangpi,jponttuset,rsoricut,vittoferrari}@google.com

Abstract

We provide here more examples of VidLN annotations. Additionally, we describe implementation details and additional experiments for ReferFormer-VNG, and more details about the VideoQA location-output questions.

1. More Video Localized Narratives Examples

In Figs. 1 to 3, we show more examples of our VidLN annotations.

2. ReferFormer-VNG

Here we describe implementation details and additional experiments for ReferFormer-VNG.

2.1. Implementation Details

For simplicity, for the ReferFormer-VNG baseline, we input only the narrative of one actor at a time (*e.g.*, only the narrative of "Parrot one" rather than the concatenation of all narratives of all actors), and the segmentation for each noun is predicted separately, both during training and inference. During training, for videos with sparse mask annotations, we only sample from the frames that have a mask annotated. For the text encoder, we use a pre-trained RoBERTa [2] model to extract both per-text-token features and whole-sentence features.

For the training hyper-parameters, we generally follow the setup of the original ReferFormer [5], and in the following we will only describe settings that are different. For fine-tuning on OVIS-VNG, we train for 6 epochs with 8 Tesla V100 GPUs and reduce the learning rate at epochs 3 and 5. For all other training setups, we train for 12 epochs with 16 Tesla V100 GPUs, and reduce the learning rate at epochs 8 and 10.

2.2. Additional Experiments

Experiments with Different Backbones. Tab. 1 shows results for ReferFormer-VNG with different visual backbones. Here we use the same backbones that are considered for the original ReferFormer [5].

The order of training is always (i) initialize the visual backbone with the checkpoint obtained on the dataset specified in the column "Initialization Dataset", (ii) pre-train for 12 epochs on COCO-PNG, (iii) optionally train for 12 epochs on UVO-VNG, and (iv) optionally fine-tune for 6 epochs on OVIS-VNG. Here, the step (iv) is only optionally used for evaluation on OVIS-VNG.

The results for ResNet-50 are the same as in the main paper and are shown here again for comparison.

Using a large Swin transformer [3] backbone ("Swin-Large") pre-trained on the larger ImageNet-21k version achieves the strongest results for all setups, reaching 40.0 $\mathcal{J}\&\mathcal{F}$ on OVIS-VNG (with fine-tuning) and 55.1 on UVO-VNG.

Using instead a Video Swin Transformer [4] ("VideoSwin"), for most setups also achieves stronger results than the ResNet-50 backbone. However, when initializing from ImageNet-1k and not training on UVO-VNG, then no video data (or only for OVIS-VNG fine-tuning) is used. In this case, the Video Swin Transformer which is specifically designed for videos, cannot unfold its full potential and yields relatively weak results. When initializing from Kinetics, and using UVO-VNG for main-training with videos, the VideoSwin results are strong, close to the results of Swin-Large.

The Video Swin Transformer result with initialization on Kinetics and evaluation on UVO-VNG needs to be taken with a grain of salt, because the Kinetics [1] training dataset overlaps with the UVO-VNG test set. However, the task for Kinetics is action classification, which is very different from VNG.

Fine-tuning on OVIS-VNG from ImageNet. We perform an additional experiment, where we initialize ReferFormer-VNG's ResNet-50 backbone on ImageNet and then directly train on the OVIS-VNG training set and afterwards evaluate on the OVIS-VNG test set. This setup achieves a $\mathcal{J}\&\mathcal{F}$ score of 25.1, which is much lower than

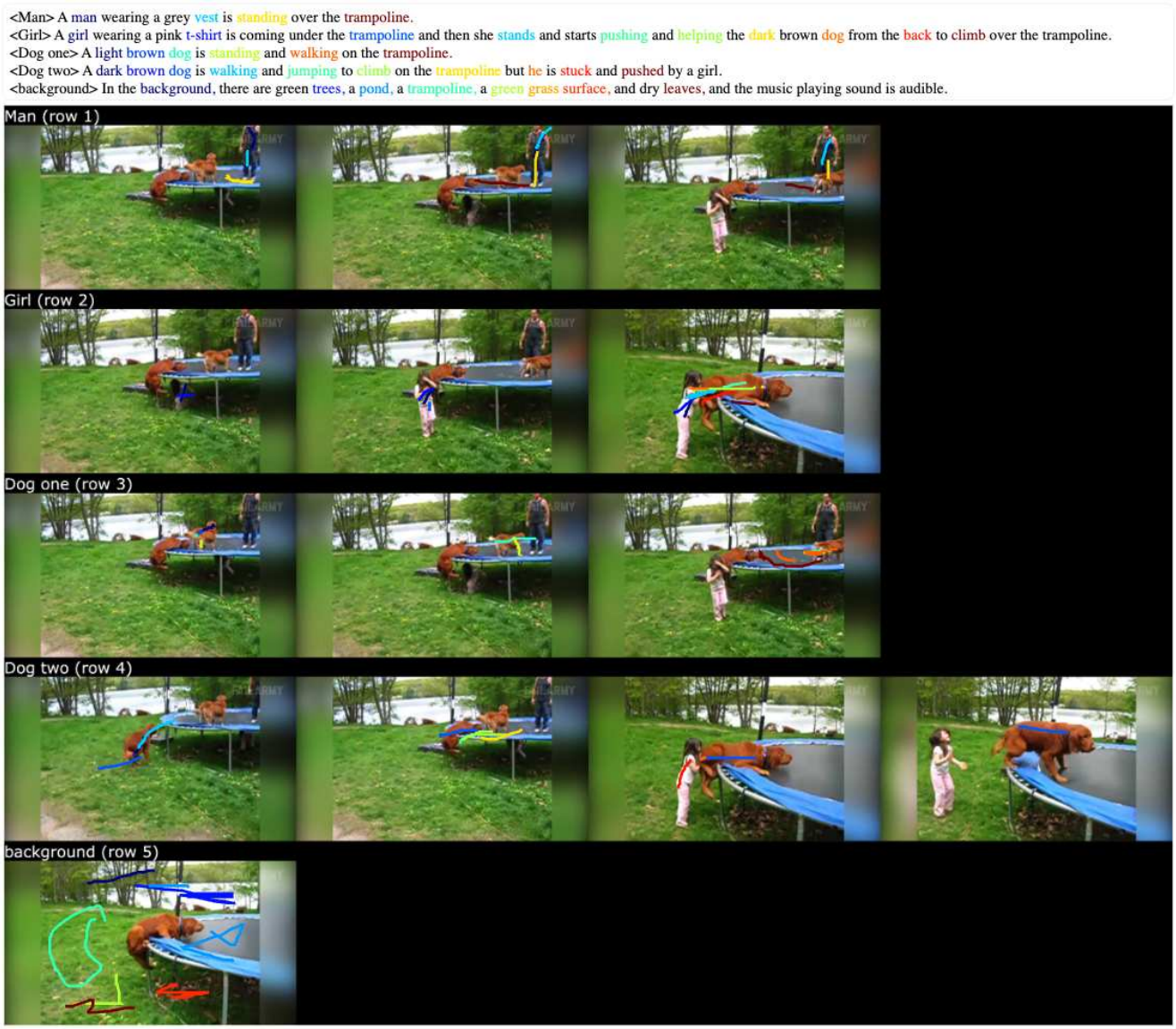

Figure 1. A VidLN example with four actors and background.

the scores of 32.4 with COCO-PNG pre-training and 32.7 with COCO-PNG pre-training and UVO-VNG main training (see Tab. 1). This result shows that without any other VNG-related training data, the result greatly suffers.

3. Video Question Answering: Location-output Questions

Here we explain for the VideoQA location-output questions, how the approximate square bounding boxes are estimated, and how we estimated the precision of the mouse trace answers.

3.1. Estimation of Approximate Square Bounding Boxes

For the evaluation of location-output questions, we start from a location ground truth in the form of a annotator-verified mouse trace on the object of interest. In order to evaluate the precision criterion of predicted bounding boxes, we estimate approximate square bounding boxes based on the mouse trace segment as follows.

A mouse trace segment only has a notion of a single scale dimension (its length), not two. Hence, we estimate a squared box rather than the usual rectangular one. We start by first fitting a square around the mouse trace segment (the trace-box, cyan in Fig. 4) by setting its center to

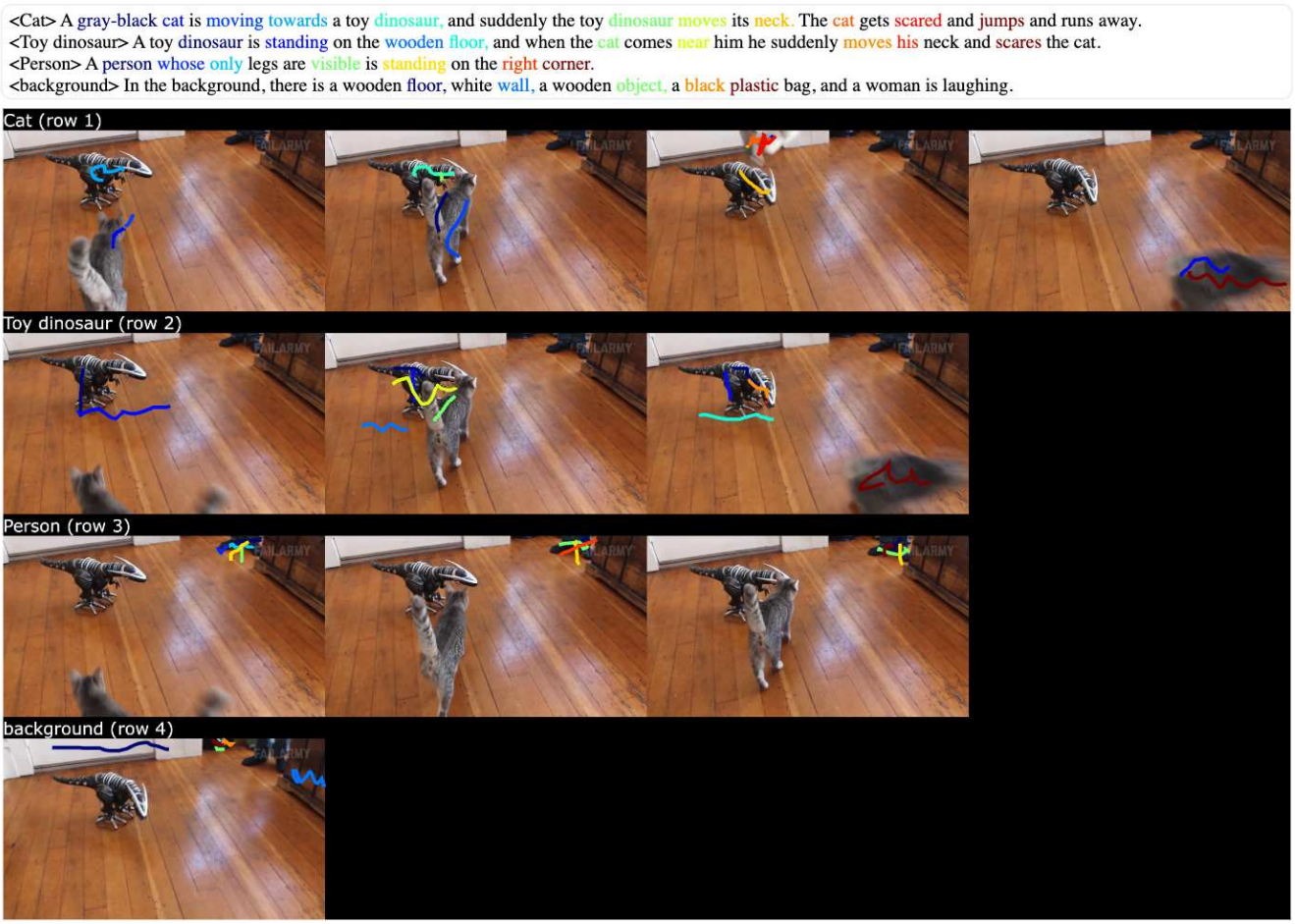

Figure 2. Another example VidLN annotation.

Backbone	Initialization Dataset	COCO-PNG Pre-training	UVO-VNG Training	OVIS-VNG no ft	OVIS-VNG ft	UVO-VNG
Swin-Large	ImageNet-21k	yes	yes	30.7	36.7	42.8
		yes	no	36.9	40.0	55.1
VideoSwin	ImageNet-1k	yes	yes	25.4	28.3	35.9
		yes	no	26.9	28.5	41.4
	Kinetics*	yes	yes	31.5	35.9	42.9
		yes	no	35.5	38.2	53.5
ResNet-50	ImageNet-1k	yes	yes	32.0	32.7	46.4
		yes	no	28.5	32.4	39.6

Table 1. VNG results with other visual backbones. All numbers are $\mathcal{J}\&\mathcal{F}$ scores. The ResNet-50 result of the main paper is also shown for comparison. "no-ft" and "ft" indicate whether we fine-tuned on the OVIS-VNG training set before evaluating on the OVIS-VNG test set. *: Note that the videos of the UVO-VNG test set overlap with the Kinetics training set, but the task is different.

the center of mass of the trace and setting its side-length to the smallest value with which the whole mouse trace segment is covered by the box. Because the mouse trace only covers part of the object, this trace-box will likely be too small. Hence, to obtain the approximate bounding box we enlarge the side length by a learned transformation.

We learn a quadratic function that performs this transformation on the OVIS-VNG train dataset, as it has ground

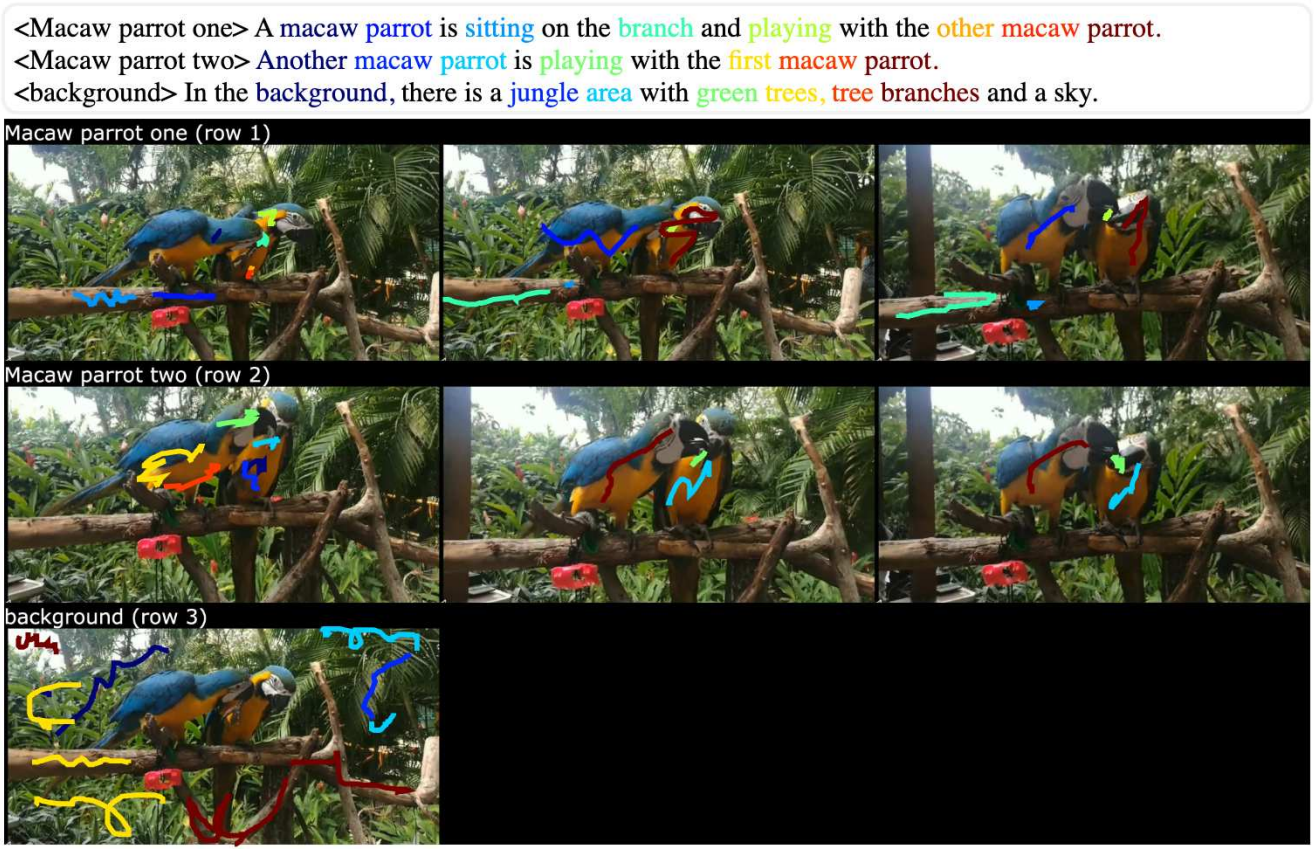

Figure 3. Another example VidLN annotation.

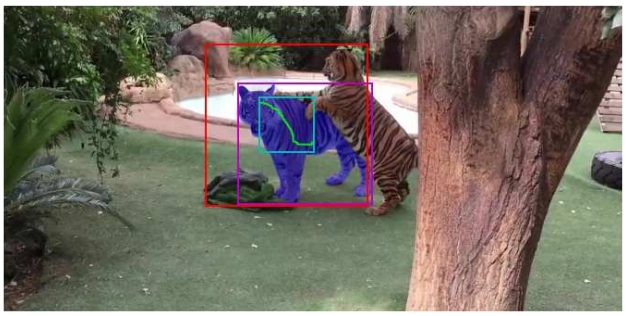

Figure 4. The estimated square bounding box (red) is obtained by scaling the (square) trace-box (cyan) of the mouse trace (green). The (unknown) ground truth mask is overlaid in blue and the (rectangular) ground truth bounding box is shown in magenta.

truth segmentation masks. The side length x of the trace-box is the input to the transformation, and the side length of the perfect square bounding box around the ground truth segmentation mask is the target output $\hat{y}$.

For each noun that has both an associated ground truth segmentation mask, and a mouse trace, we determine both the input side length x and the desired output side length $\hat{y}$. Our experiments revealed that the relationship between x and $\hat{y}$ is best captured with a quadratic transformation, as larger x need to be enlarged by a smaller factors than small x to reach their corresponding $\hat{y}$:

$$f(x) = \lambda_0 + \lambda_1 x + \lambda_2 x^2 \tag{1}$$

with three free parameters $\lambda_0, \lambda_1, \lambda_2 \in \mathbb{R}$.

As a loss function for a single training example, we consider the ratio of the estimated side length $f(x)$ to the ground-truth one $\hat{y}$. This is a scale-invariant measure, and thus more suitable for the task than an L2 loss:

$$L(x, \hat{y}) = \max\left\{\frac{f(x)}{\hat{y}}, \frac{\hat{y}}{f(x)}\right\}. \tag{2}$$

Intuitively, the ratio of the predicted side length and the desired side length should be as close to 1 as possible. The maximum is necessary to account both for cases where the prediction is too small and where it is too big. The theoretical minimum of this loss is 1.0.

Finally, we take the geometric mean over the whole training set with N inputs $x_1, \ldots, x_N$, and desired outputs

$\hat{y}_1, \ldots, \hat{y}_N$, *i.e.*,

$$L_{total} = \left(\prod_{n=1}^{N} L(x_n, \hat{y}_n) \right)^{\frac{1}{N}}. \quad (3)$$

We optimize this loss on the OVIS-VNG training set with batch gradient descent, *i.e.*, in each step the loss is calculated over the whole dataset.

3.2. Precision of Mouse Trace Answers

For the Oops dataset, we do not annotate any segmentation masks. Hence, we instead use the OVIS-VNG dataset as a proxy to measure mouse trace precision for the location-output questions of Sec. 5.2 of the main paper. Recall from Sec. 3.2 of the main paper that the trace precision, without manual verification, on OVIS-VNG is 77.3%, *i.e.*, 77.3% of the trace points are on the correct object mask on average. However, in Sec. 5.2 the candidate location-output questions are manually verified by two annotators to ensure the trace segments are correct (among other criteria). This increases the average accuracy of the questions that pass verification. To emulate this verification process on OVIS-VNG (instead of Oops), we discard trace segments with a precision of less than 25%. This filtering step removes roughly 9% of the mouse trace segments and the average precision of the remaining trace segments is very high at 92.9%. Note that the threshold was chosen conservatively and our manual verification by two annotators would likely lead to even higher precision.

References

[1] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017. 1

[2] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019. 1

[3] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021. 1

[4] Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. Video swin transformer. In *CVPR*, 2022. 1

[5] Jiannan Wu, Yi Jiang, Peize Sun, Zehuan Yuan, and Ping Luo. Language as queries for referring video object segmentation. In *CVPR*, 2022. 1