Supplemental Material: Connecting Vision and Language with Localized Narratives

Jordi Pont-Tuset, Jasper Uijlings, Soravit Changpinyo,
Radu Soricut, and Vittorio Ferrari

Google Research

This supplemental material is organized as follows:

Section 1 presents a demonstration of application of Localized Narratives for image generation. The user describes the image they want by means of a Localized Narrative and the method generates an image that matches the description.

Section 2 provides additional qualitative examples for the controlled image captioning application.

Section 3 provides an additional quantitative plot of the localization accuracy of the mouse trace in Localized Narratives for Open Images. It was suppressed from the main paper due to space limitations.

Section 4 provides additional technical details on the framework that we use for controlled image captioning.

1 Image Generation

Generating an image conditioned on a semantic segmentation map is a well-studied application [4, 5, 7, 10]. However, while such segmentation maps give control over the image to be synthesized, they do not provide a natural interface for the user. In this section, we show how we can use labelled mouse traces to generate images. This opens up a new and intuitive way for the user to provide guidance to the image generation process.

We start from SPADE [7], which is an existing, state-of-the-art framework for generating images conditioned on a pixel-wise segmentation map. We use their publicly available model that is pre-trained on COCO-stuff [2, 6], which features 182 semantic classes, including object and background classes (stuff). At test time, the model takes as input a segmentation map where pixels are labeled with these classes, and generates an image. In this section we exploit Localized Narratives as a natural interface for producing these segmentation maps efficiently, as the user can specify both the location and class label of the desired image elements at the same time, and can intuitively specify elements in their order of importance.

Localized Narrative to Semantic Segmentation Map. For this application we need to convert the labelled traces into an appropriate segmentation map. We found that scene elements should have a realistic shape for SPADE to

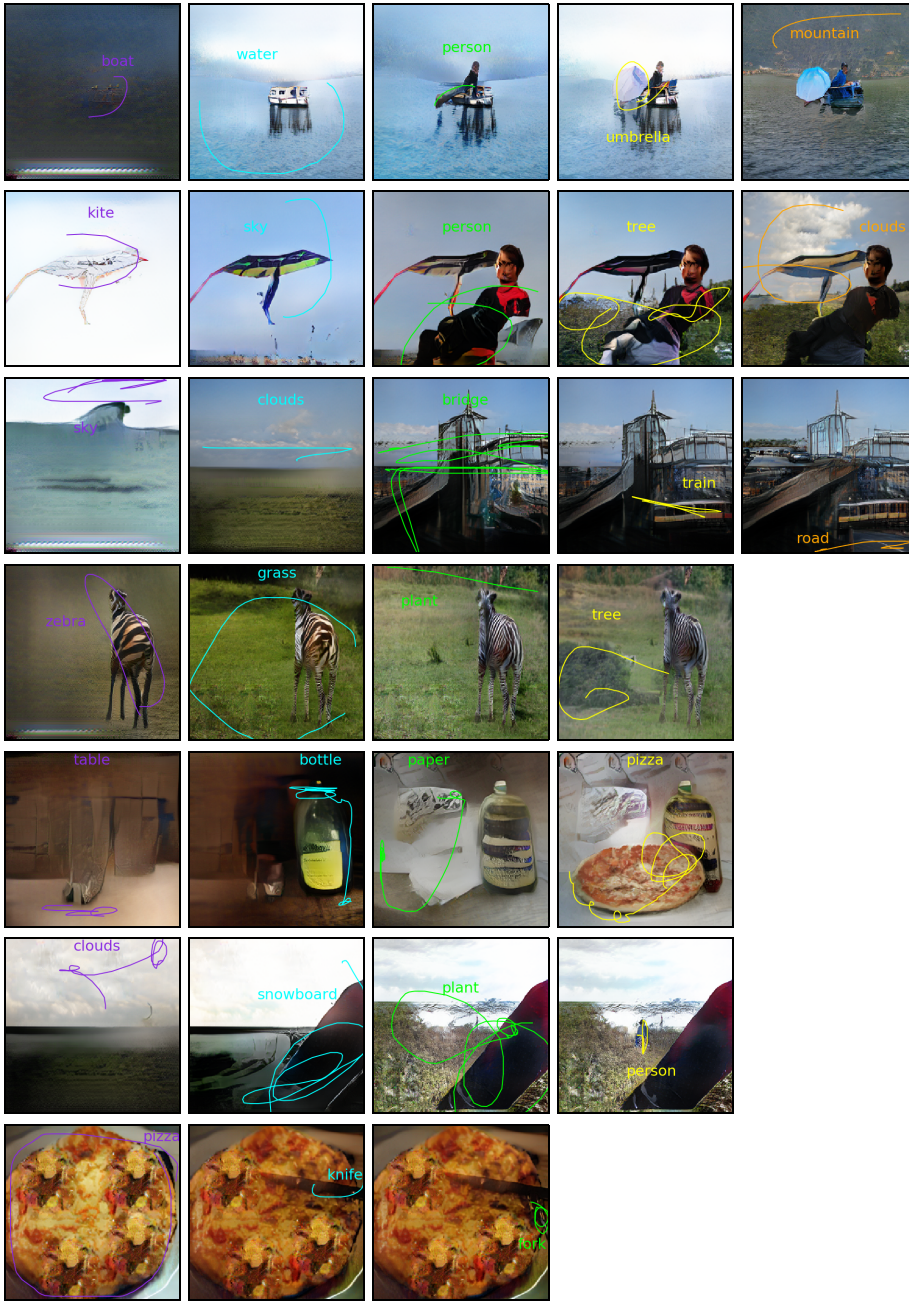


Fig. 1: **Seven examples (one per row) of image generation using mouse traces.** New image elements are iteratively added (from left to right) using a noun and its associated trace segment.

produce a pleasing image. Furthermore, SPADE deals poorly with maps which consist mostly of unlabelled pixels. To overcome this, we first collect masks for 1000 instances of each class from the COCO-stuff training set (both object and background classes). Given a trace segment with a class label, we first create its convex hull. We then compare it to all training instances of the same class and select the mask with the highest spatial overlap. This mask has a natural shape since it comes from a real instance.

Equipped with these retrieved masks, we construct a semantic segmentation map. We start from an empty map where all pixels are unlabelled, and iteratively add masks in the same order as the trace segments. An object mask is pasted on top of the current map, overwriting any previously labelled pixels. A background mask only overwrites pixels labeled as another background class. This approach results in using masks that cover more surface compared to the input trace segments, which helps reducing the surface of unlabelled pixels.

Results. Figure 1 shows seven examples created based on Localized Narratives. In both examples, the images get increasingly complex as the Localized Narrative continues, while also preserving previous details. In the first example, the closed boat becomes open once the user indicated that a person should be visible on the boat. Moreover, the addition of the mountain alters the appearance of the water. In the second example, adding the clouds effectively changes the weather conditions and therefore the illumination. The other examples follow similar patterns.

Conclusions. We demonstrated that Localized Narratives can be used for image generation. Since we kept the pre-trained SPADE [7] model unmodified and only used traces to create segmentation maps, we do not believe our framework generates better images. Instead, we demonstrated that we can generate images incrementally with an intuitive interface. More importantly, while we now only generated *nouns*, Localized Narratives opens up the possibility to also consider adjectives such as *red* or *old* and verbs such as *holding* and *riding*. We feel this presents exciting and challenging new research opportunities.

2 Additional Qualitative Examples for Controlled Image Captioning

Figure 2 and Figure 3 show additional qualitative examples of controlled versus classical image captioning on our data.

3 Localization accuracy on Open Images

Figure 4 shows the histograms of mouse trace segment locations on COCO (left) and Open Images (right) with respect to the closest box of the relevant class (The main paper also shows the histogram for COCO).

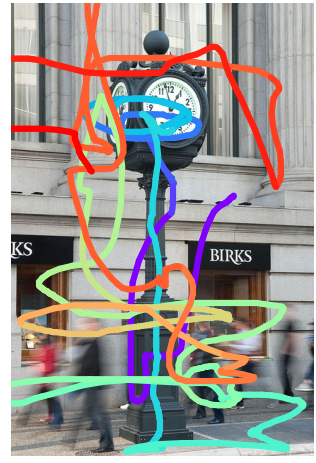
Traditional Captioning	Controlled Captioning
	
This is a black and white picture. Here we can see clocks on the pole. In the background there is a building and this is sky.	In the center of the image there is a black pole to which clocks are placed. At the bottom of the image, we can see a group of people walking on the road. In the background, there is a building.
	
In this image, there is snow on the ground which is in white color, in the middle there is a person standing on the ski board and wearing a red color jacket, in the background there are some green color plants.	Here in this picture we can see a person skiing on snow with ski board on her legs and she is also wearing gloves, goggles and a helmet on her and we can see the ground is covered with snow over there.

Fig. 2: **Controlled Captioning Qualitative Examples 1**: Traditional captioning where the input is only the image (left) versus our captioning controlled by mouse traces where the mouse traces are also an input to the model (right). Gradient indicates time.

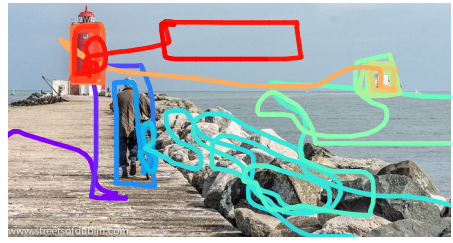
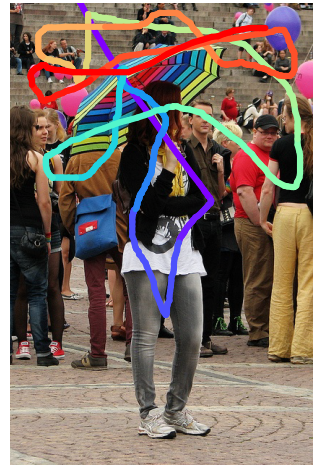
Traditional Captioning	Controlled Captioning
	
In this picture we can see a man wore jacket holding bicycle with his hand and beside to him we can see rocks, water, ship and in the background we can see sky.	In this image we can see a man standing on the left side. He is holding a bicycle in his hand. Here we can see stones on the right side. Here we can see a ship on the top right side. Here we can see a tower on the left side. This is a sky.
	
In this picture we can see a group of persons standing on the ground and in the background we can see a building, trees, sky.	A person is standing wearing a black dress and holding a umbrella. Behind her there are other people standing. At the left and right there are kites. There are trees at the back.

Fig. 3: **Controlled Captioning Qualitative Examples 2**: Traditional captioning where the input is only the image (left) versus our captioning controlled by mouse traces where the mouse traces are also an input to the model (right). Gradient indicates time.

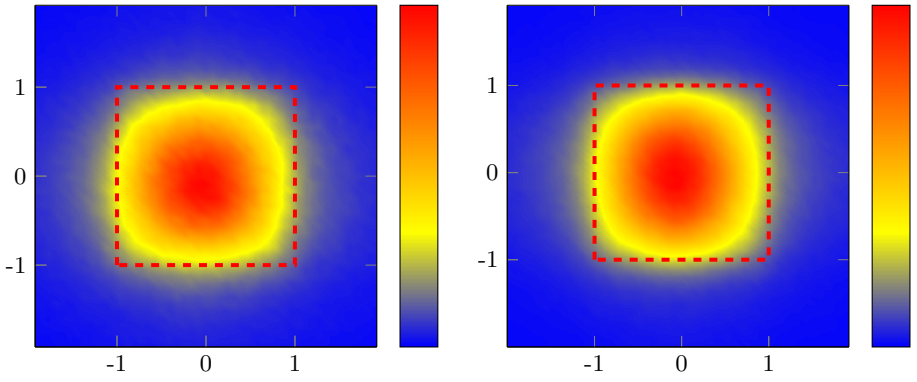


Fig. 4: **Histograms of mouse trace segment locations** on COCO (left) and Open Images (right) with respect to the closest box of the relevant class (- - -).

4 Controlled Image Captioning Details

4.1 Method and Training Details

Our transformer-based encoder-decoder image captioning model follows the architecture in [3] with a few minor differences. First, we set the number of Transformers' layers for both the encoder and the decoder to 2 instead of 6. Second, our projection layers also consist of layer normalization [1] (Sec. 4.2). Third, we set the maximum number of iterations to 150k, much smaller than the 2M used in that work. Finally, we allow the maximum number of target captions to be as long as 225 to account for the longer nature of the narration.

Besides above, our input features are standard regional Faster R-CNN [8] features: no ultra-finegrained, global, or entity features are involved. We will describe how we represent these and additional features in Section 4.2.

4.2 Representations of visual and trace features

Recall from the main text that our model consumes up to four types of features: (i) Faster R-CNN features of the automatically-detected top object proposals, representing their semantic information; (ii) the coordinate and size features of these proposals, representing the location of the detected objects. (iii) the total time duration of the mouse trace, capturing information about the expected length of the full image description. (iv) the position of the mouse trace as it moves over the image, representing the visual grounding. To create this representation, we first divide the mouse trace evenly into pseudo-segments based on the prior median word duration (0.4 sec over the whole training set). We then represent each pseudo-segment by its encapsulating bounding box, resulting in a set of features which take the same form as (ii).

Visual Features. Faster R-CNN features (i) are represented by a sequence of R=16 2,048D vectors: $f_1, f_2, \ldots, f_R$ (output by the Faster R-CNN), which are later projected onto a 512D vector and followed by layer normalization.

We represent the location of detected objects (ii) with a sequence of 5D vectors: $p_1, p_2, \ldots, p_R$. Each vector contains numbers between 0 and 1 corresponding to the top-left x and y coordinates, the bottom-right x and y coordinates, and the area with respect to the whole image. We project it to a 512D space as for (i) above.

To construct a representation of (i + ii), we add the projected and normalized vectors from each source and apply another layer normalization to the resulting vector, leading to a sequence of R 512D vectors.

For the visual features above, we do not use "time" positional encoding such that the model is permutation-invariant to the sequence vectors.

Trace Features. As mentioned in the main text, the mouse trace coordinates are uniformly divided into a 0.4-second pseudo-segments of trace coordinates and then converted into a series of corresponding bounding boxes. Thus, we now have a sequence of 5D vectors: $q_1, q_2, \ldots, q_T$, where q_j has the same form as (ii).

Each box corresponds to the smallest region that covers the trace, which might potentially not cover the whole object. To mitigate this, we extend the box in each direction by the offset δ. To represent (iii), we set $\delta = 1.0$ such as q_j's become all [0, 0, 1, 1, 1] (the region of the whole image). In other words, all the trace position information is dismissed, leaving only the total time duration of the mouse trace. On the other hand, setting $\delta = 0.1$ gives the (iii + iv) signal. After the transformation, we have a sequence of *transformed* 5D vectors: $q'_1, q'_2, \ldots, q'_T$, which are later projected onto a 512D vector and followed by layer normalization.

Different from the visual features, each trace comes with the notion of "time" — the order of the regions that are derived from traces matters. Thus, we construct such a time representation $\text{sinusoids}(1), \text{sinusoids}(2), \ldots, \text{sinusoids}(T)$, where $\text{sinusoids}(j)$ is a 512D vector of j based on sine and cosine functions of different frequencies [9]. Similarly to (i + ii), when combining the the trace features with sinusoids, we add the vectors from each source and apply another layer normalization to the resulting vector. In the end, we have a sequence of T 512D vectors.

Combining Visual and Trace Features. To construct (i + ii + iii) or (i + ii + iii + iv), we simply concatenate the "visual feature" sequence and the "trace feature" sequence and use the result as the input to the model.

References

1. Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization. arXiv preprint arXiv:1607.06450 (2016)
2. Caesar, H., Uijlings, J., Ferrari, V.: COCO-Stuff: Thing and stuff classes in context. In: CVPR (2018)

3. Changpinyo, S., Pang, B., Sharma, P., Soricut, R.: Decoupled box proposal and featurization with ultrafine-grained semantic labels improve image captioning and visual question answering. In: EMNLP-IJCNLP (2019)
4. Chen, Q., Koltun, V.: Photographic image synthesis with cascaded refinement networks. In: ICCV (2017)
5. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. In: CVPR (2016)
6. Lin, T.Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C.L., Dollár, P.: Microsoft COCO: Common objects in context. In: ECCV (2014)
7. Park, T., Liu, M.Y., Wang, T.C., Zhu, J.Y.: Semantic image synthesis with spatially-adaptive normalization. In: CVPR (2019)
8. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards real-time object detection with region proposal networks. In: NeurIPS (2015)
9. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: NeurIPS (2017)
10. Wang, T.C., Liu, M.Y., Zhu, J.Y., Tao, A., Kautz, J., Catanzaro, B.: High-resolution image synthesis and semantic manipulation with conditional gans. In: CVPR (2018)