Reading Between the Lines: Exploring Infilling in Visual Narratives

Khyathi Raghavi Chandu  Ruo-Ping Dong  Alan W Black

Language Technologies Institute, Carnegie Mellon University
{kchandu, awb}@cs.cmu.edu, ruopingd@alumni.cmu.edu

Abstract

Generating long form narratives such as stories and procedures from multiple modalities has been a long standing dream for artificial intelligence. In this regard, there is often crucial subtext that is derived from the surrounding contexts. The general seq2seq training methods render the models shorthanded while attempting to bridge the gap between these neighbouring contexts. In this paper, we tackle this problem by using infilling techniques involving prediction of missing steps in a narrative while generating textual descriptions from a sequence of images. We also present a new large scale visual procedure telling (ViPT) dataset with a total of 46,200 procedures and around 340k pairwise images and textual descriptions that is rich in such contextual dependencies. Generating steps using infilling technique demonstrates the effectiveness in visual procedures with more coherent texts. We conclusively show a METEOR score of 27.51 on procedures which is higher than the state-of-the-art on visual storytelling. We also demonstrate the effects of interposing new text with missing images during inference. The code and the dataset will be publicly available at https://visual-narratives.github.io/Visual-Narratives.

1 Introduction

Humans process information from their surrounding contexts from multiple modalities. These situations are often derived from a modality (source) and expressed in another modality (target). Recent advances have seen a surge of interest in vision and language as source and target modalities respectively. One such widely studied task is image captioning (Hossain et al., 2019; Liu et al., 2019) which provides a textual description $T$ given an image $I$. In contrast, visual storytelling (Huang et al., 2016) is the task of generating a sequence of textual descriptions $\{T_1, T_2, ..., T_n\}$ from a sequence of images $\{I_1, I_2, ..., I_n\}$.

A fundamental incongruity between how humans process information from multiple modalities and how we teach machines to do the same is that, humans are capable of bridging the information gap from surrounding contexts. Traditionally, the problem of missing context in long text generation is addressed using additional input such as entities, actions, etc., (Fan et al., 2019; Dong et al., 2019). In contrast, in the spirit of simplicity, we propose infilling techniques to interpolate the gap between surrounding contexts from a stream of images. We focus on two kinds of visual narratives namely, stories and procedures. We curated a large scale ViPT dataset with pairwise image and text descriptions comprising of 46k procedures and 340k images. The percentage of unique words in each step in stories datasets. This is usually because stories are more creative and diverse while procedures are in-domain. For both these reasons, we hypothe-
### Table 1: Details of the ViST and Visual Procedure Telling (ViPT) Dataset broken down into 10 categories

<table>
<thead>
<tr>
<th>Dataset</th>
<th>ViST</th>
<th>Visual Procedure Telling (ViPT)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Categories</td>
<td>stories</td>
<td>recipes</td>
</tr>
<tr>
<td>#narratives</td>
<td>50,136</td>
<td>34,138</td>
</tr>
<tr>
<td>#images or steps</td>
<td>209,651</td>
<td>203,519</td>
</tr>
<tr>
<td>avg #steps</td>
<td>5.00</td>
<td>5.96</td>
</tr>
<tr>
<td>avg #words/step</td>
<td>11.35</td>
<td>79.19</td>
</tr>
</tbody>
</table>

The main contributions of this paper are:

- We present a Visual Procedure Telling (ViPT) dataset similar to the Visual Storytelling (ViST) dataset with 46k procedures on various domains.
- We demonstrate the efficacy of our visual infilling technique on narratives that have stronger contextual dependencies on the rest of the sentences.

### 2 Related Work

#### Multimodal Language:
Language generation from visual modality has seen a steep rise in interest with the introduction of several large scale tasks such as image captioning (Hossain et al., 2019), visual question answering (Antol et al., 2015) and visual dialog (Das et al., 2017; Mostafazadeh et al., 2017; De Vries et al., 2017).

#### Visual Storytelling:
Huang et al. (2016) ventured into sequential step wise generation of stories by introducing visual storytelling (ViST). Chandu et al. (2019) also proposed a dataset of 16k recipes in a similar form. Recent methods have tackled ViST using adversarial learning, reinforcement learning (Wang et al., 2018; Huang et al., 2019; Hu et al., 2019), modality-fusion (Smilevski et al., 2018), traditional seq2seq models (Kim et al., 2018; Jung et al., 2020; Hsu et al., 2018) and explicit structures (Bosselut et al., 2016; Bisk et al., 2019). Though the stories in ViST demonstrate a sense of continuity, the overarching sequential context is feeble. Procedures such as cooking recipes (Salvador et al., 2019; Wang et al., 2019) on the other hand, demonstrate this characteristic inviolably. Hence, we present a large scale ViPT dataset to encourage research in this direction.

#### Infilling and Masking:
The idea is motivated by cloze tasks (Taylor, 1953) that addresses readability and understanding. However, recent advances in learning a masked language model (Devlin et al., 2019) paved way for a new trend in exploring masked contexts (Song et al., 2019; Lewis et al., 2019). Generation of meaning patches with missing portions of text is experimented by Zhu et al. (2019); Donahue et al. (2018); Fedus et al. (2018) to generate meaningful patches. Similarly, Ippolito et al. (2019) proposed a hierarchical model to generate middle span using a bag of predicted words from left and right contexts. In a similar spirit, this paper studies the effects of infilling techniques for visual narrative generation. An alternate stream of work to improve the context in stories include providing supporting information such as entities (Clark et al., 2018; Xu et al., 2018), latent templates (Wiseman et al., 2018), knowledge graphs (Yang et al., 2019), etc., explicitly. In contrast to this, infilling provides an opportune platform to implicitly learn the contextual information. Our work is positioned in the intersection of infilling and multimodal language generation.

### 3 Dataset Description

This section describes our new ViPT dataset and highlights the differences with ViST.

#### Procedures vs Stories:
Narrative properties such as content and structure in these forms are sufficiently contrastive (Gatt and Krahmer, 2018). Content in stories include characters and events while procedures include ingredients, materials and actions. While stories start from setting an era to characterizing participants, procedures start by listing ingredients followed by a step by step description.
ViPT dataset: Though stories have the potential to exhibit the properties listed above, it is challenging to observe them in ViST dataset (Huang et al., 2016) owing to the shorter sequence lengths. To facilitate multi-domain research with stronger interleaved contexts between surrounding steps, we present a large scale visual procedure telling dataset with 46k procedures comprising of 340k pairwise images and textual descriptions. It is carefully curated from a number of how-to blogging websites. Our dataset comprises of pairwise images and textual descriptions of the corresponding images, typically describing a step in a procedure. This means that each description of the step is tethered to an image. We categorized the dataset into 10 distinct domains including recipes, crafts, outdoors, lifestyle, technology, styling, fitness, hobbies, pets and miscellaneous. The category wise details of the dataset are presented in Table 1.

Differences between ViPT and ViST datasets: As observed in Table 1, the average number of steps in ViPT is higher than ViST. The average number of words per step in ViPT is also much higher, thereby presenting a more challenging long form generation task. Typically, each step in the ViPT dataset comprises of multiple sentences as opposed to ViST which is a single sentence. These long sequences also present a case for dealing with larger vocabularies as well. The recipes category alone has a vocabulary of 109k tokens while the same for stories is 25k. We also consider the diversity in vocabulary of each step by computing the average percentage of unique words in a step with respect to the rest of the narrative. While this number is a high 60% for ViST, it is 39% for ViPT. This reveals the stronger cohesive and overlapping contexts in the ViPT dataset, as compared to the ViST datasets. These overlapping contexts motivates the idea of generating a sentence by bridging the contexts from surrounding sentences. Hence it forms a suitable test bed to learn interpolation with infilling technique.

### 4 Models Description

We present infilling based techniques for learning missing visual contexts to generate narrative text from a sequence of images. As the ViST and recipes category in ViPT are of comparable sizes (both in terms of data size and the average number of steps per instance), we perform comparative experimentation on these two categories. For our ViPT category, we use 80% for training, 10% for validation and 10% for testing. An overview of infilling based training is depicted in Figure 1. The underlying encoding and decoding stages are described here.

**Encoding:** Models 1, 2 and 3 here show different variants of encoding with and without infilling.

1. **XE (baseline):** We choose a strong performing baseline model based on sequence to sequence modeling with cross entropy (XE) loss inspired from Wang et al. (2018). The visual features are extracted from the penultimate layer of ResNet-152 by passing the resized images ($\{I_1, I_2, ..., I_n\}$) of size 224 X 224. These represent the image specific local features ($\{l_1, l_2, ..., l_n\}$). These features are then passed through a bidirectional GRU layer to attain narrative level global features ($\{g_1, g_2, ..., g_n\}$) constituting the narrative context layer in Figure 1.

2. **V-Infill:** We introduce an infilling indicator function on the underlying XE model by randomly sampling an infilling index ($in_{idx}$). This is used to construct the final infilled local features as follows.

$$l_k(\forall k, s.t. 0 < k \leq n) = \begin{cases} \text{zero tensor} & \text{if } k = in_{idx} \\ l_k & \text{otherwise} \end{cases}$$

Other than the sampled $in_{idx}$, the rest of the local features for other indices remain the same. The local features for $in_{idx}$ are all masked to a zero tensor. The dropout of an entire set of local features from an image forces the model to learn to bridge the context from the left and the right images of $in_{idx}$. In this way, the infilling mechanism encourages our underlying seq2seq model to learn the local

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Stories</th>
<th>Recipes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>XE</td>
<td>V-Infill</td>
</tr>
<tr>
<td>BLEU-1</td>
<td>62.15</td>
<td>61.75</td>
</tr>
<tr>
<td>BLEU-2</td>
<td>38.31</td>
<td>37.27</td>
</tr>
<tr>
<td>BLEU-3</td>
<td>22.68</td>
<td>21.70</td>
</tr>
<tr>
<td>BLEU-4</td>
<td>13.74</td>
<td>12.96</td>
</tr>
<tr>
<td>METEOR</td>
<td>35.01</td>
<td>34.53</td>
</tr>
<tr>
<td>ROUGE_L</td>
<td>29.66</td>
<td>29.12</td>
</tr>
</tbody>
</table>

Table 2: Performance of different models on stories (from ViST) and recipes (from ViPT) datasets.
representation from contextual global features in the narrative context layer.

3. **V-Infill**: This model varies the rates in which local features are masked as training proceeds based on the indicator function above in the V-Infill model. Scheduling the number of missing features itself is a hyperparameter and we used the following setting. In the first quarter of training epochs, none are masked, then increasing it to 1 local feature for the next quarter and leaving it at 2 for the last two quarters.

**Decoding**: In all the above models, \( g_k \) are fed into a GRU decoder to predict each word (\( \hat{w}_t \)) of the step \( k \). We perform beam search with a beam size of 3 during inference. Here \( \tau \) is the number of words in each step and \( t \) is the current time step.

\[
\hat{w}_t \sim \prod_{\tau} P(\hat{w}_t^{\tau}\mid \hat{w}_t^{<\tau}, g_k)
\]

4. **INF**: We reimplemented the model achieving the state of the art results (Hu et al., 2019) on the visual storytelling dataset. Additionally, they use a relational embedding layer that captures relations across spatio-temporal sub-spaces. Our replication of their model is close, achieving a 35.5 METEOR and 63.3 BLEU-1 in comparison to the scores reported in their paper which are 35.6 and 64.4.

**Hyperparameter Setup**: We use a GRU with hidden dimension of 256 for encoder and 512 for decoder. The word embedding dimension is 512. The learning rate is 4e-4 optimized with Adam and smoothing of 1e-8. We use a dropout of 0.2 and momentum of 0.9 with a gradient clipping of 10.

The performance when experimented with a transformer based encoder along with autoregressive decoding is comparatively lesser and hence we proceed with a GRU based model. Based on the average number of steps in recipes from Table 1 which is 5.96, we truncate the recipes to 6 steps.

## 5 Results and Discussion

In this section, we present the effects of infilling both during both training and inference on ViST and ViPT datasets.

### Infilling during training: The overall performance of the models is presented in Table 2. Both the infilling model variants achieve higher scores on the recipes while not decreasing their performances on stories. We also observed that increasing the number of masked local features beyond 2 drastically decreases the performance on both datasets.

### Infilling during inference: Acquiring parallel pairwise image and narrative data in the wild is
You will need the following ingredients : - 1 cup of water , salt and pepper.

Cut the chicken into small pieces .
The first thing you will need to do is put the meat in the pan . Heat the oil in a pan and add the olive oil .
Add the salt and pepper to the pan .
Once the chicken is cooked , remove the chicken from the oven and let it cool for about 10 minutes .
Now that you have a hot pan , it ’ s time to cook the chicken.

GT

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Analysis</strong></td>
<td><strong>GT</strong></td>
<td>There are only a few components necessary to cook meat and make a pan sauce.</td>
<td>Trim the fat around the meat until almost none is left .</td>
<td>Dry the meat by blotting it with paper towels .</td>
<td>Heat a little oil on a stainless steel or cast iron skillet on high heat and wait for the oil to start smoking .</td>
</tr>
<tr>
<td><strong>XE</strong></td>
<td><strong>Analysis</strong></td>
<td>There are only a few components necessary to make a pan sauce .</td>
<td>Trim the fat around the meat until almost none is left .</td>
<td>Dry the meat by blotting it with paper towels .</td>
<td>Heat a little oil on a stainless steel or cast iron skillet on high heat and wait for the oil to start smoking .</td>
</tr>
</tbody>
</table>

| **V-Infill** | **Analysis** | The ingredients are prepared . | Cut the chicken into small pieces . | Cut the chicken into small pieces . | Cut the chicken into small pieces . | Heat a little oil on a stainless steel or cast iron skillet on high heat and wait for the oil to start smoking . |

**Figure 3** : Comparison of V-Infill and XE dealing with infilling context during inference (for making chicken roast). GT corresponds to the ground truth step. The index in each row corresponds to the index of the missing image.

Often not feasible. Hence, we perform infilling not only at train time but also at inference time to evaluate the ability of the model to bridge contexts when the corresponding image is absent and deal with real-world data imputation scenarios. Figure 3 demonstrates an example of generated samples by infilling different indices. As observed, the XE model either generates generic sentences to infill content or copies the content from the surrounding contexts. In contrast, the V-Infill model is able to generate new sentences with small phrases of sentences being copied. Underspecification of content is an issue in both these models where there is a room for improvement.

**Lengths of generated sequences** : We compare infilling during inference between baseline XE model and our V-Infill model in Table 3. While the METEOR scores remain comparable, the BLEU scores steadily increase as we move the instance to the right. Specifically, these jumps are bigger after step 3. Quantitatively, this is the result of the model being able to produce longer sequences as we move to the right as BLEU gets penalized for short sentences. Qualitatively, this implies that the initial steps like specifying the ingredients are more crucial compared to later ones. A similar observation emerges by analyzing the effects of infilling during training. The average length of generated recipes by XE is 71.26 and by V-Infill is 76.49. A similar trend is observed for stories in Table 4.

**Human Evaluation** : We also conducted human evaluation (interface in Figure 2) in the form of preference testing comparing our V-Infill model with XE by randomly sampling 10 recipes. Among 20 evaluators, our V-Infill model is preferred 64% of the times over the XE model on average.

**6 Conclusions and Future Work**

We demonstrate that infilling is a simple yet effective technique and a step towards maximizing the utilization of surrounding contexts in visual narratives. Infilling is the strategy of enabling the model to learn surrounding contextual information by masking spans of input while the decoding attempts in generating the entire text. To experimentally support our hypothesis, we collect a new large scale ViPT dataset of 46k procedures comprising 10 categories. We compare the performance of our model and conclusively show the higher significance of infilling based techniques in visual procedures compared to visual stories. In future, we plan to explore the following two directions: (1) interpolating the contexts between consecutive steps by introducing a new infilled image, and (2) addressing the underspecification problem by controlling the content in infilled image with explicit guidance. These infilling techniques are also immensely useful when dealing with data imputation with missing contexts and collaborative authoring in real-world scenarios.
References


7 Supplemental Material

1. Human Evaluation: Figure 2 depicts a screenshot of our human evaluation interface. A sequence of images are presented on top of the screen. This evaluation is conducted to compare between XE and V-Infill model. The generated sentences from the 2 models, in this case XE and V-Infill are presented after the images. Note that the generated outputs are presented in arbitrarily random order for each example to ensure there is no bias while performing preference testing. Human subjects are asked to pick one of the generated recipes for the given sequence of images. 10 such recipes are presented for each user and we averaged the preference scores among 20 users.

2. Data Collection Process: We manually examined around 10 blogging websites with various user written text on several how-to activities. Among these we found that snapguide and instructables are consistent in the form of pairs of textual descriptions along with their images. We are going to release the scripts used to collect this data as well as preprocess them. We removed all the procedures in which at least one image in each step is absent. Once all this preprocessing is done, the data contained the following categories in both the websites. These categories are based on the tags given by the bloggers to the articles they have written from among the categories that each website offers. These categories for each of these websites are:

- **snapguide**: recipes, games-tricks, sports-fitness, gardening, style, lifestyle, outdoors, beauty, arts-crafts, home, music, photography, pets, automotive, technology
- **instructables**: crafts, cooking, teachers, circuits, living, workshop, outside

In union, they are a total of 18 categories. We manually examined a few procedures in each of the categories and regrouped them into 10 broad categories that are presented in Table 1. A list of urls corresponding to the data is submitted along with the paper.

3. Visualization of topics: Each of the categories in our Visual Procedure Telling (ViPT) are analyzed for the topics present in them. A screenshot highlighting one of the topics in recipes category is presented in Figure 4. To get a more detailed understanding of these topics in the dataset, we hosted the topic visualizations here: visual-narratives.github.io/Visual-Narratives/.

4. Performance of infilling during inference for Visual Story Telling: Table 4 demonstrates the effects of infilling various indices during inference. This table is analogous to Table 3 for stories. As we can see, a similar trend in the increase in all the automatic metrics are present as we move the infill index to the right of the story. While that is still the case, a very interesting observation is that the difference between the performance of XE and Infill models for any given index is much higher for recipes compared to stories. The infilling technique is bringing much more value to the task when the nature of the text is procedural and dependent more on the surrounding contexts.

5. Reproducibility Checklist:

1. For all reported experimental results:

   a. A clear description of the mathematical setting, algorithm, and/or model: Yes. There are two important definitions to understand in the paper. The first is the task definition which is outlined in Section 1. The second is the infilling technique definition which is presented in Section
<table>
<thead>
<tr>
<th>Hyperparameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Num of Epochs</td>
<td>100</td>
</tr>
<tr>
<td>Batch size</td>
<td>64</td>
</tr>
<tr>
<td>Learning rate</td>
<td>4e-4</td>
</tr>
<tr>
<td>Momentum</td>
<td>0.9</td>
</tr>
<tr>
<td>Gradient clipping</td>
<td>10</td>
</tr>
<tr>
<td>Dropout</td>
<td>0.2</td>
</tr>
<tr>
<td>Optimizer</td>
<td>Adam</td>
</tr>
<tr>
<td>alpha for Adam</td>
<td>0.8</td>
</tr>
<tr>
<td>beta for Adam</td>
<td>0.99</td>
</tr>
<tr>
<td>Adam smoothing</td>
<td>1e-8</td>
</tr>
<tr>
<td>Word Embedding dim</td>
<td>512</td>
</tr>
<tr>
<td>Visual feature dim (ResNet)</td>
<td>2048</td>
</tr>
<tr>
<td>Num of GRU layers</td>
<td>1</td>
</tr>
<tr>
<td>GRU Hidden dim</td>
<td>512</td>
</tr>
<tr>
<td>Beam width</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 6: Hyperparameters of the best performing model

4.

(b) Submission of a zip file containing source code, with specification of all dependencies, including external libraries, or a link to such resources (while still anonymized): No. Though they are not submitted along with the paper, we plan to release both the code and the data along with the scripts to preprocess the data here: visual-narratives.github.io/Visual-Narratives/

(c) Description of computing infrastructure used: The models are trained on a GPU with driver version 430.26 and CUDA version 10.2. The GPU RAM being used have a maximum size of 8GB and 12 GB for different experiments. A few of the experiments are run on GeForce GTX 1070 with 1920 cores and GeForce GTX 1080 Ti with 3584 cores.

(d) Average runtime for each approach: On average the experiments on the visual procedure telling dataset took around 8 hours and the same on the visual story telling dataset took around 11 hours. These numbers are the averages of the 4 different models on each dataset comprising of XE, V-Infill, V-InfillR and INet.

(e) Number of parameters in each model: For the models pertaining to XE, V-Infill and V-InfillR, the number of parameters are listed below. For the ViPT dataset, the number of trainable parameters are 15,665,896. For the ViST dataset, the number of trainable parameters are 14,390,894. The INet model for the ViPT dataset has 19,345,128 and on the stories data is 18,070,126. As we can see, the infilling variants are able to achieve better performance on procedural texts with comparatively fewer parameters. This justifies our hypothesis that infilling is a simple yet effective technique to perform text generation where there are strong contextual dependencies.

2. For all experiments with hyperparameter search:

(a) Bounds for each hyperparameter: For the scheduling of the infilling, we tried the following settings. We experimented with the following settings. The first setting is: (i) 0 images for first quarter (ii) 1 image for second quarter (iii) 2 images for the last half number of epochs. The second setting is: (i) 0 images for first quarter (ii) 1 image for 2 quarters (iii) 2 images for the last quarter number of epochs. The third setting is: (i) 1 images for first quarter (ii) 2 image for second quarter (iii) 3 images for the last half number of epochs. The fourth setting is: (i) 0 images for first quarter (ii) 1 image for second quarter (iii) 2 images for the third quarter (iv) 3 images for the last quarter number of epochs. For the rest of
the hyperparameters, we used the same as the baseline. We contacted the authors of the INet model, which is the state-of-the-art on visual storytelling for the hyperparameter values they have used and used the same in our work to replicate their results and perform reliable comparisons.

(b) **Hyperparameter configurations for best-performing models:** The hyperparameters for the best performing model are described in Section 4. A more detailed set of the hyperparameters are presented in Table 6.

(c) **Number of hyperparameter search trials:** The important hyperparameter that we tuned is the scheduling of infilling. The hyperparam that performed best is described in Section 4. In total, we have tried 4 settings to arrive pick the best setting.

(d) **The method of choosing hyperparameter values (e.g., uniform sampling, manual tuning, etc.) and the criterion used to select among them (e.g., accuracy):** We performed manual tuning in various settings for the rate of infilling in the V-InfillR model. The criterion used to select the best model is METEOR score, which is the automatic metric to evaluate our model.

(e) **Expected validation performance, or the mean and variance as a function of the number of hyperparameter trials:**

3. **For all datasets used:**

(a) **Relevant statistics such as number of examples:** The statistics of the ViPT dataset along with category wise distribution for the number of procedures, number of steps, average number of steps and average number of words per step are presented in Table 1. The same table also compares these distributions with the ViST dataset.

(b) **Details of train/validation/test splits:** The splits to perform training, validation and testing are readily available for ViST dataset. For our newly collected ViPT dataset, we conduct experiments on the recipes category. These splits are done for this dataset as 80% for training, 10% for validation and 10% for testing.

(c) **Explanation of any data that were excluded, and all pre-processing steps:** The data is cleaned of any html tags as they are collected from the blogging websites.

(d) **A link to a downloadable version of the data:** We plan to release the pre-processed data along with visual features. It will be available for download here: visual-narratives.github.io/Visual-Narratives/

(e) **For new data collected, a complete description of the data collection process, such as instructions to annotators and methods for quality control:** We collected the data from how-to blogging websites. From the collected procedures, we perform an additional step of removing the ones that do not have any images. We keep the procedures in the final dataset that have at least one image per step. Additionally, we manually inspected the different types of procedures and categorized them into the 10 categories as mentioned in the paper.