Procedural Text Generation from a Photo Sequence

Taichi Nishimura¹, Atsushi Hashimoto², Shinsuke Mori³
¹Graduate School of Informatics, Kyoto University
²OMRON SINIC X Corporation
³Academic Center for Computing and Media Studies, Kyoto University
nishimura.taichi.43x@st.kyoto-u.ac.jp
atsushi.hashimoto@sinicx.com
forest@i.kyoto-u.ac.jp

Abstract
Multimedia procedural texts, such as instructions and manuals with pictures, support people to share how-to knowledge. In this paper, we propose a method for generating a procedural text given a photo sequence allowing users to obtain a multimedia procedural text. We propose a single embedding space both for image and text enabling to interconnect them and to select appropriate words to describe a photo. We implemented our method and tested it on cooking instructions, i.e., recipes. Various experimental results showed that our method outperforms standard baselines.

1 Introduction
A multimedia procedural text, e.g. instruction sentences with photos, inspires users to learn a new skill. Some web services, such as Cookpad and Instructables, capitalize on this characteristics allowing users to submit photos or video clips in addition to instruction sentences to explain procedures better. An automatic system outputting instruction sentences given a photo sequence supports authors of such services.

In this paper, we propose a method for generating a procedural text from a photo sequence. As shown in Figure 1, given a photo sequence, it outputs a step consisting of some instruction sentences for each photo. Among various kinds of procedural texts, we take the cooking domain for example because cooking is daily activity and recipe is one of the most familiar procedural texts. Our task may resemble visual storytelling (Huang et al., 2016) sharing the input. The main difference is, however, that the output of our task is a procedural text that should be concise and concrete allowing its readers to execute it. In cooking domain the output, a recipe consisting of multiple sentences, should have necessary and sufficient foods, tools, and actions in the correct order. For this reason procedural text generation seemed to be difficult and was initially solved by formulating it as a retrieval task (Salvador et al., 2017; Zhu et al., 2019; Chen and Ngo, 2016). Another similar task setting is recipe generation from a photo of the final dish using ingredient predictor (Salvador et al., 2019). This setting may be, however, very difficult or even impossible because a single photo of the final dish does not contain sufficient information for its production procedure.

In this background, we focus on procedural text generation from a photo sequence and, as a solution, we propose to incorporate a retrieval method into a generation model. Our method generates a procedural text in two phases. First given a photo sequence, it retrieves relating steps using a joint embedding model, which has been pre-trained on a large amount of image/step pairs available in the Web. Then it generates word sequences referring to these retrieved steps.

We conducted experiments to evaluate our method in comparison with existing methods in BLEU, ROUGE-L, and CIDEr-D. The results showed the effectiveness of the proposed method. However, as often pointed out, these metrics are not perfect because they ignore importance of each token. Thus we investigated the ratios of correctly verbalized important terms, i.e., foods, tools, and
actions in the recipe case. The result showed that the proposed method verbalizes them more correctly. Some qualitative analyses also suggested that the proposed method generates a suitable procedural text for a given photo sequence.

2 Related Work

Some researchers have been tackling problems to generate a procedural text from various inputs. In cooking domain, Salvador et al. (2019) tried to generate a recipe from an image of a complete dish. Bosselut et al. (2018) and Kiddon et al. assumed a title and ingredients as the input. It may be, however, almost impossible to generate a good recipe due to lack of information on mesomorphic states of ingredients. Mori et al. (2014a) generated a procedural text from a meaning representation taking intermediate states into account. Close look at these studies suggests the importance of the information on intermediate processes for a procedural text generator to be practical.

Thus we assume a photo sequence as the input. Since authors of multimedia procedural texts at least take a photo at each important step, this setting is realistic. Sharing the input and output media the most similar task may be the visual storytelling (Huang et al., 2016). Liu et al. (2017) proposed a joint embedding model for image and text to interconnect them. Contrary to this, we propose to generate sentences directly from the vectors in this shared space.

3 Procedural Text Generation

Figure 2 shows an overview of our method. (i) We pre-train the joint embedding model using image/text pairs. Then, given a photo sequence, our method repeats the following procedures for each photo: (ii) retrieve the top $K$ nearest steps to the photo in the embedding space, (iii) compute the vector by the encoder from the input photo and the average of the $K$ vectors of the retrieved steps, and (iv) decode a step represented by the photo.

3.1 Joint embedding model

First, (i) we train a joint embedding model based on the two branch networks (Wang et al., 2016), which transform different modality representations, i.e., text and image, into a common feature space using multiple layer perceptrons with nonlinear functions. With the resulting joint embedding model we can calculate similarity between a step and an image. In our preliminary experiment, the original networks did not achieve a good performance because there are many omissions in procedural texts (Malmaud et al., 2014). To solve this problem, we propose to insert a bi-directional LSTM (biLSTM) to the textual encoder to refer to the preceding and following steps in addition to the current one.

3.2 Procedural text generation assisted by vector retrieval

The input is a photo sequence $\{v_1, v_2, \ldots, v_N\}$. Each photo $v_n$ is converted into an image embedding vector $\tilde{v}_n$ through the image encoder of the joint embedding model. For each photo we execute the following procedures.

**Image vector enhancement (ii):** We retrieve the top $K$ nearest vectors $R = \{r_1, r_2, \ldots, r_K\}$ among those converted from the steps in the training dataset for the embedding space. Then we calculate their average

$$\tilde{r}_n = \frac{1}{K} \sum_{k=1}^{K} r_k,$$

and concatenate it to the image embedding vector for the photo to have $u_n = (\tilde{v}_n, \tilde{r}_n)$.

**Encoding (iii):** We provide an LSTM with the output of the encoder $o_n$ as the initial vector. It decodes repeatedly outputting a token in the vocabulary including period, beginning of step (/step/), and its ending (\step) to form a step consisting of multiple sentences. We also use the general attention mechanism (Luong et al., 2015), which helps the model to generate important terms by receiving feedback from retrieved step embedding vectors. Based on a hidden vector $h_t$ at decoding $t$-th token and the series of retrieved step embedding vectors $R$, we calculate the attention weight of $k$-th step $a_k^t$ at $t$-th token decoding as follows:

$$a_k^t = \frac{\exp(r_kW_a h_t)}{\sum_{j=1}^{K} \exp(r_jW_a h_t)}$$

$$c_t = \sum_{k=1}^{K} a_k^t r_k$$

$$\hat{h}_t = \tanh(W_c (c_t, h_t))$$

Decoding (iv): We provide an LSTM with the output of the encoder $o_n$ as the initial vector. It decodes repeatedly outputting a token in the vocabulary including period, beginning of step (/step/), and its ending (\step) to form a step consisting of multiple sentences. We also use the general attention mechanism (Luong et al., 2015), which helps the model to generate important terms by receiving feedback from retrieved step embedding vectors. Based on a hidden vector $h_t$ at decoding $t$-th token and the series of retrieved step embedding vectors $R$, we calculate the attention weight of $k$-th step $a_k^t$ at $t$-th token decoding as follows:

$$a_k^t = \frac{\exp(r_kW_a h_t)}{\sum_{j=1}^{K} \exp(r_jW_a h_t)}$$

$$c_t = \sum_{k=1}^{K} a_k^t r_k$$

$$\hat{h}_t = \tanh(W_c (c_t, h_t))$$
where \( W_a \) and \( W_c \) are trainable parameters. The probability distribution of the output tokens 
\[
p(y_t | y_{<t}, o_n) = \text{softmax}(W_o \hat{h}_t + b_o),
\]
where \( W_o \) is the weight matrix to transform the size of the vector \( \hat{h}_t \) into the vocabulary size and \( b_o \) is a bias weight. In the test phase the model outputs the token of the highest probability. After decoding, the last hidden state of the decoder is reset to the initial state of the decoder to get ready to generate the next step.

In the training phase, we minimize the sum of 
\[
L(\theta) = - \sum_{D} \sum_{t=1}^{T} \log p(y_t | y_{<t}, o_n; \theta),
\]
where \( D \) is the entire training dataset and \( \theta \) is all the parameters and \( T \) is the length of target instruction sentences.

4 Evaluation

In order to evaluate our method, we implemented it and tested it in the cooking domain.

4.1 Parameter setting

We employed ResNet-50 (He et al., 2016) trained with ImageNet (Deng et al., 2009) as the image encoder of the joint embedding model. We removed only its last softmax layer. Thus the dimension of the output vector is 2,048. In the joint embedding model, we set the dimension of the hidden layer of biLSTM to 1,024, hence the dimension of the output vector is its double (2,048) because the bi-directional output vectors are concatenated. Training procedure is the same as the two branch networks (Wang et al., 2016). In our generation model, we set the dimension size of the hidden vector to 512 in both of the biLSTM encoder and the LSTM decoder. To train the model, we freeze the joint embedding weights and all other weights were optimized by Adam (Kingma and Ba, 2015) with the initial value \( \alpha = 0.001 \). The number of retrieved steps \( K \) was set to ten.

4.2 Dataset

To prepare the dataset we selected all recipes (in Japanese) from the Cookpad Image Dataset (Harashima et al., 2017) under the condition that

<table>
<thead>
<tr>
<th></th>
<th>train</th>
<th>valid</th>
<th>test</th>
</tr>
</thead>
<tbody>
<tr>
<td>( D_{emb} )</td>
<td># recipes</td>
<td>162,463</td>
<td>18,059</td>
</tr>
<tr>
<td></td>
<td># steps</td>
<td>5.65</td>
<td>5.57</td>
</tr>
<tr>
<td></td>
<td># tokens</td>
<td>24.51</td>
<td>24.51</td>
</tr>
<tr>
<td></td>
<td>vocabulary</td>
<td>24,152</td>
<td></td>
</tr>
<tr>
<td>( D_{gen} )</td>
<td># recipes</td>
<td>21,039</td>
<td>2,281</td>
</tr>
<tr>
<td></td>
<td># steps</td>
<td>8.09</td>
<td>8.10</td>
</tr>
<tr>
<td></td>
<td># tokens</td>
<td>19.35</td>
<td>19.51</td>
</tr>
<tr>
<td></td>
<td>vocabulary</td>
<td>11,091</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Dataset statistics.

<table>
<thead>
<tr>
<th></th>
<th>image2step</th>
<th>step2image</th>
</tr>
</thead>
<tbody>
<tr>
<td>( D_{emb} )</td>
<td>w/o biLSTM</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td>w/ biLSTM</td>
<td>6</td>
</tr>
</tbody>
</table>

Table 2: MedR results.
Table 3: Results of overlap metrics for generated procedural texts by the models and the baselines.

<table>
<thead>
<tr>
<th></th>
<th>BLEU1</th>
<th>BLEU4</th>
<th>ROUGE-L</th>
<th>CIDEr-D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Image</td>
<td>27.3</td>
<td>4.2</td>
<td>18.3</td>
<td>13.2</td>
</tr>
<tr>
<td>Image + Title</td>
<td>28.6</td>
<td>5.4</td>
<td>17.6</td>
<td>13.1</td>
</tr>
<tr>
<td>Image + Title + Ingredient</td>
<td>28.8</td>
<td>6.1</td>
<td>19.4</td>
<td>14.6</td>
</tr>
<tr>
<td>Proposed method</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w/o biLSTM</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Image embedding</td>
<td>26.7</td>
<td>4.1</td>
<td>17.7</td>
<td>13.8</td>
</tr>
<tr>
<td>Image embedding + top1 step embedding</td>
<td>31.4</td>
<td>6.8</td>
<td>21.5</td>
<td>11.7</td>
</tr>
<tr>
<td>Proposed method</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w/ biLSTM</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Image embedding</td>
<td>31.0</td>
<td>6.5</td>
<td>21.6</td>
<td>14.9</td>
</tr>
<tr>
<td>Image embedding + top1 step embedding</td>
<td>32.9</td>
<td>6.7</td>
<td>21.8</td>
<td>16.4</td>
</tr>
<tr>
<td>Image embedding + topK step embedding</td>
<td><strong>33.4</strong></td>
<td><strong>7.2</strong></td>
<td><strong>20.7</strong></td>
<td><strong>14.9</strong></td>
</tr>
</tbody>
</table>

Table 3: Results of overlap metrics for generated procedural texts by the models and the baselines.

To evaluate our method, we measured overall generation qualities as well as ratios of important terms. We also present some generated examples.
which are calculated by word2vec (Mikolov et al., 2013). The results, Table 3, show that the proposed method achieves a higher performance than the baselines in these metrics.

### 4.4.2 Important term verbalization

Traditional overlap metrics do not measure verbalization of important terms in the generated procedural text. In the cooking domain, they are foods (F), tools (T), and actions (Ac) as the statistics on the flow graph corpus (Mori et al., 2014b) indicate. Thus we calculated the ratios of correctly verbalized ones in these categories. Although this is more important than the ordinary overlap metrics, synonyms and spelling variants prevent us from automatic calculation. Therefore we selected 50 generated recipes randomly from the test set and manually counted numbers of important terms occurring in the generated recipes, in their references, and both. Table 4 shows the results. We see that clearly Top1 retrieval outperforms the baseline and TopK is far better than top1 for all the term categories, showing advantages of our image vector enhancement in procedural text generation.

### 4.4.3 Qualitative analysis

In Figure 3 we present example generated sentences by the baseline, those by the proposed method, and their reference. It can be seen that the proposed method is capable of generating recipes which contain the ingredients really shown in the photos, while the baseline tends to just enumerate frequent ingredients in the training set.

### 5 Conclusion

In this paper, we proposed a method for generating a procedural text from a photo sequence and tested it in the cooking domain. Our main ideas are (1) biLSTM to overcome omissions in the text side for the joint embedding space, (2) image vector enhancement by top K retrieval, and (3) overall design for procedural text generation from a photo sequence. Various analyses on experimental results, which are also important contributions of this paper, showed that our method outperforms standard baselines and each one of our ideas contributes to it.

The generated sentences have the correspondence to the source photos allowing us to generate multimedia procedural texts as a natural extension of our method.

### Acknowledgments

We would like to be grateful to Hirokuni Maeta, Yoko Yamakata, and Cookpad Inc. for supporting our work. We also thank anonymous reviewers for useful comments that greatly improved the manuscript.

### References


Ting-Hao Kenneth Huang, Francis Ferraro, Nasrin Mostafazadeh, Aishwarya Agrawal Ishan Misra, Jacob Devlin, Ross Girshick, Xiaodong He, Pushmeet Kohli, Dhruv Batra, C. Lawrence Zitnick, Devi Parikh, Lucy Vanderwende, Michel Galley, and


