Procedural Text Generation from a Photo Sequence

Taichi Nishimura[1], **Atsushi Hashimoto**[2], **Shinsuke Mori**[3]
[1]Graduate School of Informatics, Kyoto University
[2]OMRON SINIC X Corporation
[3]Academic 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.

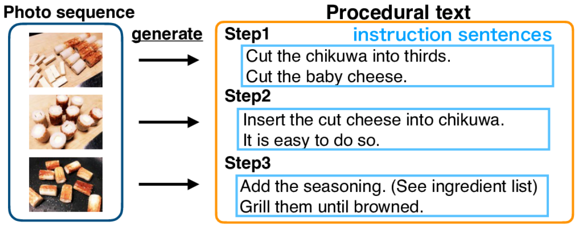


Photo sequence
generate
Procedural text
instruction sentences
Step1
Cut the chikuwa into thirds.
Cut the baby cheese.
Step2
Insert the cut cheese into chikuwa.
It is easy to do so.
Step3
Add the seasoning. (See ingredient list)
Grill them until browned.

Figure 1: An overview of our task. The input is a photo sequence (left). The task is to output a step consisting of instruction sentences (right) for each photo.

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

Proceedings of The 12th International Conference on Natural Language Generation, pages 409–414,
Tokyo, Japan, 28 Oct - 1 Nov, 2019. ©2019 Association for Computational Linguistics

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 encorder 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 $(\boldsymbol{v}_1,\ \boldsymbol{v}_2,\ \ldots,\ \boldsymbol{v}_N)$. Each photo $\boldsymbol{v}_n$ is converted into an image embedding vector $\hat{\boldsymbol{v}}_n$ 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 $R = (\boldsymbol{r}_1,\ \boldsymbol{r}_2,\ \ldots,\ \boldsymbol{r}_K)$ among those converted from the steps in the training dataset for the embedding space. Then we calculate their average

$$\bar{\boldsymbol{r}}_n = \frac{1}{K}\sum_{k=1}^{K}\boldsymbol{r}_k, \tag{1}$$

and concatenate it to the image embedding vector for the photo to have $\boldsymbol{u}_n = (\hat{\boldsymbol{v}}_n, \bar{\boldsymbol{r}}_n)$.

Encoding (iii): We provide the enhanced image embedding vector to a biLSTM

$$\boldsymbol{o}_n = \mathrm{biLSTM}(\boldsymbol{u}_n). \tag{2}$$

Decoding (iv): We provide an LSTM with the output of the encoder $\boldsymbol{o}_n$ 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 recieving feedback from retrieved step embedding vectors. Based on a hidden vector $\boldsymbol{h}_t$ 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:

$$\begin{aligned} a_k^t &= \frac{\exp(\boldsymbol{r}_k \boldsymbol{W}_a \boldsymbol{h}_t)}{\sum_{j=1}^{K}\exp(\boldsymbol{r}_j \boldsymbol{W}_a \boldsymbol{h}_t)} && (3) \\ \boldsymbol{c}_t &= \sum_{k=1}^{K} a_k^t \boldsymbol{r}_k && (4) \\ \tilde{\boldsymbol{h}}_t &= \tanh(\boldsymbol{W}_c(\boldsymbol{c}_t, \boldsymbol{h}_t)), && (5) \end{aligned}$$

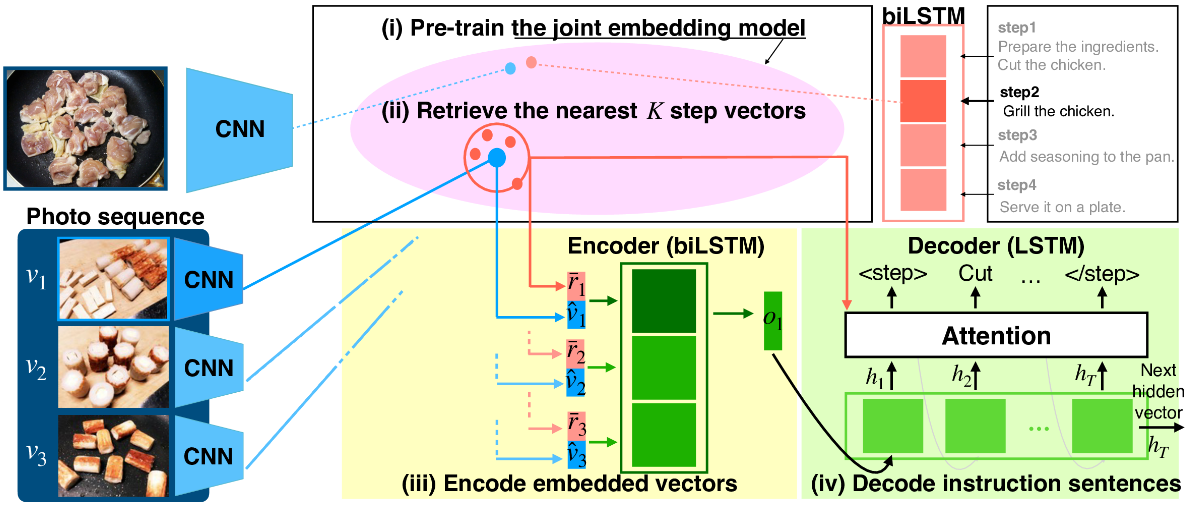


Figure 2: The outline of the proposed method.

		train	valid	test
D_{emb}	# recipes	162,463	18,059	20,104
	# steps	5.65	5.57	5.66
	# tokens	24.51	24.51	24.40
	vocabulary	24,152		
D_{gen}	# recipes	21,039	2,281	2,598
	# steps	8.09	8.10	8.10
	# tokens	19.35	19.51	19.32
	vocabulary	11,091		

Table 1: Dataset statistics.

	image2step	step2image
w/o biLSTM	23	24
w/ biLSTM	**6**	**6**

Table 2: MedR results.

where $\boldsymbol{W}_a$ and $\boldsymbol{W}_c$ are trainable parameters. The probability distribution of the output tokens $p(y_t|y_{<t}, \boldsymbol{o}_n)$ is calculated as follows:

$$p(y_t|y_{<t}, \boldsymbol{o}_n) = \text{softmax}(\boldsymbol{W}_o \tilde{\boldsymbol{h}}_t + \boldsymbol{b}_o), \qquad (6)$$

where $\boldsymbol{W}_o$ is the weight matrix to transform the size of the vector $\tilde{\boldsymbol{h}}_t$ into the vocabulary size and $\boldsymbol{b}_o$ 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 the negative log likelihood over all the tokens in the training set

$$L(\boldsymbol{\theta}) = -\sum_{\mathcal{D}} \sum_{t=1}^{T} \log p(y_t|y_{<t}, \boldsymbol{o}_n; \boldsymbol{\theta}), \qquad (7)$$

where $\mathcal{D}$ is the entire training dataset and $\boldsymbol{\theta}$ 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.

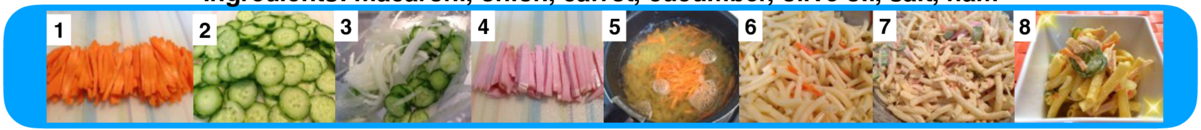
4.1 Parameter setting

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 doubble (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

Title: ひと味ちがう♪＊うちのマカロニサラダ (A bit difference♪ Macaroni salad)
Ingredients: macaroni, onion, carrot, cucumber, olive oil, salt, ham



Baseline

1: Prepare ingredients.
2: Slice up the onion.
3: Slice up the onion.
4: Slice up the onion.
5: Add water and consommé, and turn off the heat when boiled.
6: Serve it on a plate.
7: Add shichimi if you want.
8: Add shichimi if you want.

Proposed method

1: **Cut the carrot into thin strips.**
2: **Slice up the cucumber** and add salt.
3: Cut the vegetables.
4: Cut the bacon.
5: Add the egg and mix them.
6: Add the pasta and **cover it with the olive oil.**
7: Add olive oil and **salt.**
8: **Serve it on a plate.**

Reference

1: **Cut the carrot into thin strips.**
2: **Slice up the cucumber** and onion.
3: Squeeze them with salt.
4: Cut the ham.
5: Add boiled water and salt, and boil the macaroni and carrot.
6: Drain it using a strainer, and **cover it with the olive oil.**
7: Add dressings and **salt.**
8: **Serve it on a plate.**

Figure 3: Output examples. **Word sequences in bold green** are correct instructions, while those in underlined red are incorrect ones. **Those double underlined** are correctly verbalized ingredients.

		BLEU1	BLEU4	ROUGE-L	CIDEr-D
Baseline	Image	27.3	4.2	18.3	13.2
	Image + Title	28.6	5.4	17.6	13.1
	Image + Title + Ingredient	28.8	6.1	19.4	14.6
Proposed method w/o biLSTM	Image embedding + top1 step embedding	26.7	4.1	17.7	13.8
	Image embedding + top$K$ step embedding	31.4	6.8	21.5	11.7
Proposed method w/ biLSTM	Image embedding	31.0	6.5	21.6	14.9
	Image embedding + top1 step embedding	32.9	6.7	**21.8**	**16.4**
	Image embedding + top$K$ step embedding	**33.4**	**7.2**	20.7	14.9

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

an image is attached to all the steps in a recipe[1]. To obtain reliable results, we extracted recipes consisting of reasonable length (7-10 steps), which are denoted by D_{gen}, for text generation test. We used the rest, D_{emb}, as the training set for the joint embedding model. The size of D_{gen} is not enough to train the joint embedding model and generation model jointly, thus we train each model using D_{gen} and D_{emb} independently. All tokens appearing less than three times were replaced with the unknown word symbol. Table 1 shows statistics of the datasets.

4.3 Effect on the joint embedding space

First we check the effect of the biLSTM insertion. We calculated the cosine similarity in the common space for ranking the relevant steps and relevant images and measured image2step and step2image retrieval performance in median rank (MedR). Table 2 shows the results on a subset of randomly selected 1,000 step-image pairs from the test set. From this result, we see that the insertion of the biLSTM improves the original two branch networks enabling to refer to the context.

4.4 Results and Discussion

To evaluate our method, we measured overall generation qualities as well as ratios of important terms. We also present some generated examples.

4.4.1 Overlap metrics

To evaluate the proposed method, we calculated BLEU1, BLEU4, ROUGE-L, and CIDEr-D scores over all the recipes in the test set. As the baselines, we train the model to output texts using an LSTM from multiple images (Huang et al., 2016) and mean word vectors of a title and ingredients,

[1] Cookpad Image Dataset contains 3.10 million images and steps, but some steps lack images.

		F	T	Ac	Total
Baseline	Recall	7.9	22.6	19.2	14.8
	Precision	12.3	15.8	17.0	15.4
	F1	9.6	18.6	18.0	15.1
Top1 w/ biLSTM	Recall	18.5	24.7	31.6	25.2
	Precision	23.8	21.0	21.1	21.9
	F1	20.8	22.7	25.3	23.4
Top$K$ w/ biLSTM	Recall	40.5	29.8	35.9	37.2
	Precision	43.6	26.8	32.4	36.1
	F1	**42.0**	**28.2**	**34.0**	**36.6**

Table 4: The verbalization ratios of important terms.

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 Top$K$ 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 Mr. Hirokuni Maeta, Prof. Yoko Yamakata, and Cookpad Inc. for supporting our work. We also thank anonymous reviewers for useful comments that greatly improved the manuscript.

References

Antoine Bosselut, Asli Celikyilmaz, Xiaodong He, Jianfeng Gao, Po-Sen Huang, and Yejin Choi. 2018. Discourse-aware neural rewards for coherent text generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 173–184.

Jingjing Chen and Chong-Wah Ngo. 2016. Deep-based ingredient recognition for cooking recipe retrieval. In *Proceedings of the 24th ACM international conference on Multimedia*, pages 32–41.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 248–255.

Jun Harashima, Yuichiro Someya, and Yohei Kikuta. 2017. Cookpad image dataset: An image collection as infrastructure for food research. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1229–1232.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 770–778.

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

Margaret Mitchell. 2016. Visual storytelling. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1233–1239.

Chloé Kiddon, Luke Zettlemoyer, and Yejin Choi. Globally coherent text generation with neural checklist models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 329–339.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the International Conference for Learning Representations*.

Yu Liu, Jianlong Fu, Tao Mei, and Chang Wen Chen. 2017. Let your photos talk: Generating narrative paragraph for photo stream via bidirectional attention recurrent neural networks. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pages 1445–1452.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421.

Jonathan Malmaud, Earl J. Wagner, Nancy Chang, and Kevin Murphy. 2014. Cooking with semantics. In *Proceedings of the ACL Workshop on Semantic Parsing*, pages 33–38.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *In Advances in Neural Information Processing Systems*.

Shinsuke Mori, Hirokuni Maeta, Tetsuro Sasada, Koichiro Yoshino, Atsushi Hashimoto, Takuya Funatomi, and Yoko Yamakata. 2014a. Flowgraph2text: Automatic sentence skeleton compilation for procedural text generation. In *Proceedings of the 8th International Natural Language Generation Conference*, pages 118–122.

Shinsuke Mori, Hirokuni Maeta, Yoko Yamakata, and Tetsuro Sasada. 2014b. Flow graph corpus from recipe texts. In *Proceedings of the 9th International Conference on Language Resources and Evaluation*, pages 2370–2377.

Amaia Salvador, Michal Drozdzal, Xavier Giro i Nieto, and Adriana Romero. 2019. Inverse cooking: Recipe generation from food images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10453–10462.

Amaia Salvador, Nicholas Hynes, Yusuf Aytar, Javier Marin, Ferda Ofli, Ingmar Weber, and Antonio Torralba. 2017. Learning cross-modal embeddings for cooking recipes and food images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3020–3028.

Liwei Wang, Yin Li, Jing Huang, and Svetlana Lazebnik. 2016. Learning two-branch neural networks for image-text matching tasks. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 5005–5013.

Bin Zhu, Chong-Wah Ngo, Jingjing Chen, and Yanbin Hao. 2019. R2GAN: cross-modal recipe retrieval with generative adversarial network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11477–11486.