FlowGraph2Text: Automatic Sentence Skeleton Compilation for Procedural Text Generation

†[1]Shinsuke Mori ‡[2]Hirokuni Maeta [1]Tetsuro Sasada [2]Koichiro Yoshino
[3]Atsushi Hashimoto [1]Takuya Funatomi [2]Yoko Yamakata
[1]Academic Center for Computing and Media Studies, Kyoto University
[2]Graduate School of Informatics, Kyoto University
[3]Graduate School of Law, Kyoto University
Yoshida Honmachi, Sakyo-ku, Kyoto, Japan
†forest@i.kyoto-u.ac.jp

Abstract

In this paper we describe a method for generating a procedural text given its flow graph representation. Our main idea is to automatically collect sentence skeletons from real texts by replacing the important word sequences with their type labels to form a skeleton pool. The experimental results showed that our method is feasible and has a potential to generate natural sentences.

1 Introduction

Along with computers penetrating in our daily life, the needs for the natural language generation (NLG) technology are increasing more and more. If computers understand both the meaning of a procedural text and the progression status, they can suggest us what to do next. In such situation they can show sentences describing the next instruction on a display or speak it.

On this background we propose a method for generating instruction texts from a flow graph representation for a series of procedures. Among various genres of procedural texts, we choose cooking recipes, because they are one of the most familiar procedural texts for the public. In addition, a computerized help system proposed by Hashimoto et al. (2008) called smart kitchen is becoming more and more realistic. Thus we try to generate cooking procedural texts from a formal representation for a series of preparation instructions of a dish.

As the formal representation, we adopt the flow graph representation (Hamada et al., 2000; Mori et al., 2014), in which the vertices and the arcs correspond to important objects or actions in cooking and relationships among them, respectively. We use the flow graphs as the input and the text parts as the references for evaluation.

Our generation method first automatically compiles a set of templates, which we call the skeleton pool, from a huge number of real procedural sentences. Then it decomposes the input flow graph into a sequence of subtrees that are suitable for a sentence. Finally it converts subtrees into natural language sentences.

2 Recipe Flow Graph Corpus

The input of our LNG system is the meaning representation (Mori et al., 2014) for cooking instructions in a recipe. A recipe consists of three parts: a title, an ingredient list, and sentences describing cooking instructions (see Figure 1). The meaning of the instruction sentences is represented by a directed acyclic graph (DAG) with a root (the final dish) as shown in Figure 2. Its vertices have a pair of an important word sequence in the recipe and its type called a recipe named entity (NE)[1]. And its arcs denote relationships between them. The arcs are also classified into some types. In this paper, however, we do not use arc types for text generation, because we want our system to be capable of generating sentences from flow graphs output by an automatic video recognition system[2] or those drawn by internet users.

Each vertex of a flow graph has an NE composed of a word sequence in the text and its type such as food, tool, action, etc. Table 3

[‡] His current affiliation is Cybozu Inc., Koraku 1-4-14, Bunkyo, Tokyo, Japan.

[1] Although the label set contains verb phrases, they are called named entities.

[2] By computer vision techniques such as (Regneri et al., 2013) we may be able to figure out what action a person takes on what objects. But it is difficult to distinguish the direct object and the indirect object, for example.

Proceedings of the 8th International Natural Language Generation Conference, pages 118–122,
Philadelphia, Pennsylvania, 19-21 June 2014. ©2014 Association for Computational Linguistics

1. 両手鍋で油を熱する。
   (In a Dutch oven, heat oil.)
   セロリと青ねぎとニンニクを加える。
   (Add celery, green onions, and garlic.)
   1 分ほど炒める。
   (Cook for about 1 minute.)
2. ブイヨンと水とマカロニと胡椒を加え
   パスタが柔らかくなるまで煮る。
   (Add broth, water, macaroni, and pepper, and simmer until the pasta is tender.)
3. 刻んだセージをまぶす。
   (Sprinkle the snipped sage.)

Figure 1: A recipe example. The sentences are one of the ideal outputs of our problem. They are also used as the reference in evaluation.

lists all of the type labels along with the average numbers of occurrences in a recipe text and examples. The word sequences of verbal NEs do not include their inflectional endings. From the definition we can say that the content words are included in the flow graph representation. Thus an NLG system has to decide their order and generate the function words (including inflectional endings for verbs) to connect them to form a sentence.

3 Recipe Text Generation

The problem in this paper is generating a procedural text for cooking (ex. Figure 1) from a recipe flow graph (ex. Figure 2).

Our method is decomposed into two modules. In this section, we explain them in detail.

3.1 Skeleton Pool Compilation

Before the run time, we first prepare a skeleton pool. A skeleton pool is a collection of skeleton sentences, or skeletons for short, and a skeleton is a sentence in which NEs have been replaced with NE tags. The skeletons are similar to the so-called templates and the main difference is that the skeletons are automatically converted from real sentences. The following is the process to prepare a skeleton pool.

1. Crawl cooking procedural sentences from recipe sites.
2. Segment sentences into words by a word segmenter KyTea (Neubig et al., 2011). Then recognize recipe NEs by an NE recognizer PWNER (Mori et al., 2012).
3. Replace the NE instances in the sentences with NE tags.

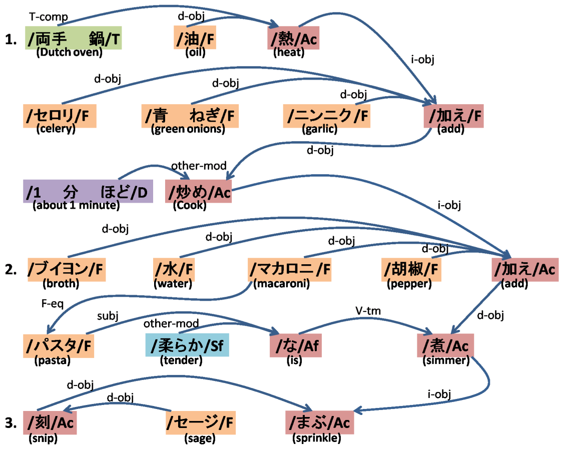

Figure 2: The flow graph of the example recipe.

Table 3: Named entity tags with average frequence per recipe.

NE tag	Meaning	Freq.
F	Food	11.87
T	Tool	3.83
D	Duration	0.67
Q	Quantity	0.79
Ac	Action by the chef	13.83
Af	Action by foods	2.04
Sf	State of foods	3.02
St	State of tools	0.30

We store skeletons with a key which is the sequence of the NE tags in the order of their occurrence.

3.2 Sentence Planning

Our sentence planner produces a sequence of subtrees each of which corresponds to a sentence. There are two conditions.

Cond. 1 Each subtree has an Ac as its root.

Cond. 2 Every vertex is included in at least one subtree.

As a strategy for enumerating subtrees given a flow graph, we choose the following algorithm.

1. search for an Ac vertex by the depth first search (DFS),
2. each time it finds an Ac, return the largest subtree which has an Ac as its root and contains only unvisited vertices.
3. set the visited-mark to the vertices contained in the returned subtree,
4. go back to 1 unless all the vertices are marked as visited.

In DFS, we choose a child vertex randomly because a recipe flow graph is unordered.

Table 1: Corpus specifications.

Usage	#Recipes	#Sent.	#NEs	#Words	#Char.
Test	40	245	1,352	4,005	7,509
NER training	360	2,813	12,101	51,847	97,911
Skeleton pool	100,000	713,524	*3,919,964	*11,988,344	22,826,496

The numbers with asterisc are estimated values on the NLP result.

Table 2: Statistical results of various skeleton pool sizes.

No. of sentences used for skeleton pool compilation	2,769 (1/256)	11,077 (1/64)	44,308 (1/16)	177,235 (1/4)	708,940 (1/1)
No. of uncovered subtrees	52	27	17	9	4
Average no. of skeletons	37.4	124.3	450.2	1598.1	5483.3
BLEU	11.19	11.25	12.86	13.12	13.76

3.3 Sentence Generation

Given a subtree sequence, our text realizer generates a sentence by the following steps.

1. Collect skeletons from the pool whose NE key matches the NE tag sequence specified by the subtree[3].
2. Select the skeleton that maximize a scoring function among collected ones. As the first trial we use the frequency of skeletons in the pool as the scoring function.
3. Replace each NE in the skeleton with the word sequence of the corresponding NE in the subtree.

4 Evaluation

We conducted experiments generating texts from flow graphs. In this section, we report the coverage and the sentence quality.

4.1 Experimental Settings

The recipe flow graph corpus (Mori et al., 2014) contains 200 recipes. We randomly selected 40 flow graphs as the test data from which we generate texts. The other 160 recipes were used to train the NE recognizer PWNER (Mori et al., 2012) with 200 more recipes that we annotated with NE tags. To compile the skeleton pool we crawled 100,000 recipes containing 713,524 sentences (see Table 1).

4.2 Skeleton Pool Coverage

First we counted the numbers of the skeletons that matches with a subtree (Step 1 in Subsection 3.3) for all the subtrees in the test set by changing the number of the recipe sentences used for the skeleton pool compilation.

Table 2 shows the numbers of subtrees that do not have any matching skeleton in the pool (uncovered subtrees) and the average number of skeletons in the pool for a subtree. From the results shown in the table we can say that when we use 100,000 recipes for the skeleton compilation, our method can generate a sentence for 98.4% subtrees. And the table says that we can halve the number of uncovered subtrees by using about four times more sentences. The average number of the skeletons says that we have enough skeletons in average to try more sophisticated scoring functions.

4.3 Text Quality

To measure the quality of generated texts, we first calculated the BLEU ($N = 4$) (Papineni et al., 2002) with taking the original recipe texts as the references. The unit in our case is a sequence of sentences for a dish. Table 2 shows the average BLEU for all the test set. The result says that the more sentences we use for the skeleton pool compilation, the better the generated sentences become.

The absolute BLEU score, however, does not tell much about the quality of generated texts. As it is well known, we can sometimes change the instruction order in dish preparation. Therefore we conducted a subjective evaluation in addition. We asked four evaluators to read 10 texts generated from 10 flow graphs and answer the following questions.

Q1. How many ungrammatical two-word sequences does the text contain?

Q2. How many ambiguous wordings do you find in the text?

Then we show the evaluators the original recipe text and asked the following question.

[3] This part is language dependent. Since Japanese is SOV language, the instance of Ac is placed at the last of the sentence to be generated. Languages of other types like English may need some rules to change the NE tag order specified by the subtree into the proper sentence element order.

Table 4: Result of text quality survey on 10 recipe texts.

	Evaluator 1			Evaluator 2			Evaluator 3			Evaluator 4		
BLEU	Q1	Q2	Q3	Q1	Q2	Q3	Q1	Q2	Q3	Q1	Q2	Q3
6.50	13	2	4	11	0	3	12	0	2	7	1	2
7.99	7	2	2	5	2	2	7	1	1	4	2	2
10.09	18	2	4	15	2	1	17	4	1	11	4	2
11.60	24	1	4	13	2	4	18	2	4	13	1	2
13.35	6	1	4	6	0	4	7	1	5	4	1	2
14.70	16	1	4	12	2	4	12	0	3	6	2	2
16.76	9	2	3	6	1	3	7	1	3	5	3	2
19.65	8	2	5	6	1	1	4	1	4	4	2	4
22.85	18	1	4	15	2	5	12	2	2	7	3	2
31.35	5	1	5	5	0	4	5	1	3	5	1	4
Ave.	12.4	1.5	3.9	9.4	1.2	3.1	10.1	1.3	2.8	6.6	2.0	2.4
PCC	−0.30	−0.46	+0.57	−0.24	−0.24	+0.36	−0.46	−0.04	+0.26	−0.29	−0.04	+0.70

PPC stands for Pearson correlation coefficient.

Q3. Will the dish be the same as the original recipe when you cook according to the generated text? Choose the one among 5: completely, 4: almost, 3: partly, 2: different, or 1: unexecutable.

Table 4 shows the result. The generated texts contain 14.5 sentences in average. The answers to Q1 tell that there are many grammatical errors. We need some mechanism that selects more appropriate skeletons. The number of ambiguous wordings, however, is very low. The reason is that the important words are given along with the subtrees. The average of the answer to Q3 is 3.05. This result says that the dish will be partly the same as the original recipe. There is a room for improvement.

Finally, let us take a look at the correlation of the result of three Qs with BLEU. The numbers of grammatical errors, i.e. the answers to Q1, has a stronger correlation with BLEU than those of Q2 asking the semantic quality. These are consistent with the intuition. The answer to Q3, asking overall text quality, has the strongest correlation with BLEU on average among all the questions. Therefore we can say that for the time being the objective evaluation by BLEU is sufficient to measure the performance of various improvements.

5 Related Work

Our method can be seen a member of template-based text generation systems (Reiter, 1995). Contrary to the ordinary template-based approach, our method first automatically compiles a set of templates, which we call skeleton pool, by running an NE tagger on the real texts. This allows us to cope with the coverage problem with keeping the advantage of the template-based approach, ability to prevent from generating incomprehensible sentence structures. The main contribution of this paper is to use an accurate NE tagger to convert sentences into skeletons, to show the coverages of the skeleton pool, and to evaluate the method in a realistic situation.

Among many applications of our method, a concrete one is the smart kitchen (Hashimoto et al., 2008), a computerized cooking help system which watches over the chef by the computer vision (CV) technologies etc. and suggests the chef the next action to be taken or a good way of doing it in a casual manner. In this application, the text generation module make a sentence from a subtree specified by the process supervision module.

There are some other interesting applications: a help system for internet users to write good sentences, machine translation of a recipe in a different language represented as a flow graph, or automatic recipe generation from a cooking video based on CV and NLP researches such as (Regneri et al., 2013; Yamakata et al., 2013; Yu and Siskind, 2013).

6 Conclusion

In this paper, we explained and evaluated our method for generating a procedural text from a flow graph representation. The experimental results showed that our method is feasible especially when we have huge number of real sentences and that some more sophistications are possible to generate more natural sentences.

Acknowledgments

This work was supported by JSPS Grants-in-Aid for Scientific Research Grant Numbers 26280084, 24240030, and 26280039.

References

Reiko Hamada, Ichiro Ide, Shuichi Sakai, and Hidehiko Tanaka. 2000. Structural analysis of cooking preparation steps in japanese. In *Proceedings of the fifth international workshop on Information retrieval with Asian languages*, number 8 in IRAL '00, pages 157–164.

Atsushi Hashimoto, Naoyuki Mori, Takuya Funatomi, Yoko Yamakata, Koh Kakusho, and Michihiko Minoh. 2008. Smart kitchen: A user centric cooking support system. In *Proceedings of the 12th Information Processing and Management of Uncertainty in Knowledge-Based Systems*, pages 848–854.

Shinsuke Mori, Tetsuro Sasada, Yoko Yamakata, and Koichiro Yoshino. 2012. A machine learning approach to recipe text processing. In *Proceedings of Cooking with Computer workshop.*

Shinsuke Mori, Hirokuni Maeta, Yoko Yamakata, and Tetsuro Sasada. 2014. Flow graph corpus from recipe texts. In *Proceedings of the Nineth International Conference on Language Resources and Evaluation.*

Graham Neubig, Yosuke Nakata, and Shinsuke Mori. 2011. Pointwise prediction for robust, adaptable japanese morphological analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics.*

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318.

Michaela Regneri, Marcus Rohrbach, Dominikus Wetzel, Stefan Thater, Bernt Schiele, and Manfred Pinkal. 2013. Grounding action descriptions in videos. *Transactions of the Association for Computational Linguistics*, 1(Mar):25–36.

Ehud Reiter. 1995. Nlg vs. templates. In *Proceedings of the the Fifth European Workshop on Natural Language Generation*, pages 147–151.

Yoko Yamakata, Shinji Imahori, Yuichi Sugiyama, Shinsuke Mori, and Katsumi Tanaka. 2013. Feature extraction and summarization of recipes using flow graph. In *Proceedings of the 5th International Conference on Social Informatics*, LNCS 8238, pages 241–254.

Haonan Yu and Jeffrey Mark Siskind. 2013. Grounded language learning from video described with sentences. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics.*