Frame Selection for Producing Recipe with Pictures from an Execution Video of a Recipe

Taichi Nishimura
Graduate School of Informatics, Kyoto University
Kyoto, Japan
nishimura.taichi.43x@st.kyoto-u.ac.jp

Atsushi Hashimoto
Omron Sinic X Corporation
Tokyo, Japan
atsushi.hashimoto@sinicx.com

Yoko Yamakata
Department of Information and Communication Engineering, The University of Tokyo
Tokyo, Japan
yamakata@mi.u-tokyo.ac.jp

Shinsuke Mori
Academic Center for Computing and Media Studies, Kyoto University
Kyoto, Japan
forest@i.kyoto-u.ac.jp

ABSTRACT

In cooking procedure instruction, text format plays an important role in conveying quantitative information accurately, such as time and quantity. On the other hand, image format can smoothly convey qualitative information (e.g., the target food state of a procedure) at a glance. Our goal is to produce multimedia recipes, which have texts and corresponding pictures, for chefs to better understand the procedures. The system takes a procedural text and its unedited execution video as the input and outputs selected frames for instructions in the text. We assume that a frame suits to an instruction when they share key objects. Under this assumption, we extract the information of key objects using named entity recognizer from the text and object detection from the frame, and we convert them into feature vectors and calculate their cosine similarity. To enhance the measurement, we also calculate the scene importance based on the latest changes in object appearance, and aggregate it to the cosine similarity. Finally we align the instruction sequence and the frame sequence using the Viterbi algorithm referring to this suitability and get the frame selection for each instruction. We implemented our method and tested it on a dataset consisting of text recipes and their execution videos. In the experiments we compared the automatic alignment results with those by human annotators. The precision, recall, and F-measure showed that the proposed approach made a steady improvement in this challenging problem of selecting pictures from an unedited video.

CCS CONCEPTS

• **Computing methodologies** → **Scene understanding**; • **Applied computing** → *Multi / mixed media creation*.

KEYWORDS

procedural text, execution video, frame selection

ACM Reference Format:
Taichi Nishimura, Atsushi Hashimoto, Yoko Yamakata, and Shinsuke Mori. 2019. Frame Selection for Producing Recipe with Pictures from an Execution Video of a Recipe. In *11th Workshop on Multimedia for Cooking and Eating Activities (CEA'19), June 10, 2019, Ottawa, ON, Canada.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3326458.3326928

1 INTRODUCTION

In cooking, there is a big gap between professional and amateur chefs. Even following a recipe text, the quality of dishes by amateur chefs is largely different from those by the professional chefs. One major reason is the difference in their background knowledge and experiences. Professional chefs refer to information not written in the recipe text but from their experiences. To help amateur chefs get the idea of shapes or colors of intermediate products or final ones, multimedia recipes, in which a picture is added to each instruction sentence, are useful for better results.

Writing such multimedia recipes is, however, not easy because it requires taking photographs at each step. The chef is sometimes forced to suspend cooking (i.e. leave the tool, clean hands, and take the camera) at a good timing for photographs to explain the action. One solution is to have a camera fixed at a high point of the kitchen to record the entire cooking procedures and select frames to be attached to each instruction.

In this paper, we propose a method for automatizing frame selection from a video recording an execution of a text form recipe to produce a multimedia version of it. Given a text recipe and its execution video, our method selects a suitable frame to be attached to each instruction, which is defined as the smallest unit in recipe texts representing a single action (here an instruction means an action description in text recipes, which we define in section 3.2.1 in detail). We assume that a frame is suitable for an instruction when key objects appearing in the frame and those in the instruction are common.

To extract objects in the instruction, the recipe texts are decomposed into instructions by natural language processing tools such as tokenizer and recipe named entity recognizer. And we detect objects in each frame of the video. Then we map instructions and frames into a common feature vector space and calculate their cosine similarity. The similarity is used in the Viterbi algorithm to align the instructions and frames. Here, we assume monotonicity (the chef does not change the order of instructions). Finally we select a best suitable frame for each instruction from the aligned frames to output recipe with pictures.

To refine the cosine similarity measurement, we also consider *scene importance* based on the following assumption; A frame will be more suitable for selection if important objects changes their visual appearance significantly in the previous frames. We name it after scene importance, and calculate it using pre-trained neural network. We add this score to the cosine similarity to select frames.

We implemented our method and tested it on KUSK Dataset [6]. KUSK Dataset consists of text recipes and their execution videos. The recipes have been downloaded from an internet site named Cookpad[1]. Each video records the entire cooking actions by a chef following a specified recipe observed by three fixed cameras placed at the top of the kitchen. To evaluate the proposed method, we prepared correct frames by asking two human annotators to select frames for each instruction. In the experiments we compared the automatic alignment results with those by the annotators and evaluate the proposed method.

2 RELATED WORK

There have been some attempts at integrating natural language and image or movie. The cooking domain is considered as a good example for such attempt. In this section we first describe an overview of the research in this domain. Then we describe related work to align each step of a recipe and segments of its movie.

2.1 Integration of Language and Other Modalities in Cooking Domain

In recent years, exploring the relationship between image and natural language has attracted a great deal of attention [3, 20, 23]. In cooking domain, there are a large amount of recipes, images and videos available on the Web, so some researchers use these data. In the subsequent parts, we describe multimodal dataset in the cooking domain. Then we describe applications of these datasets.

2.1.1 **Dataset.** There are large image/text pair datasets collected from the Web. Cookpad Image Dataset [4] is one of the largest dataset in this domain. It contains 3.10 million images, and 1.64 million completed images. Salvador *et al.*[17] build a dataset named Recipe1M, which consists of 1.0 million cooking images. Compared with image/text datasets, there are smaller video/text pair datasets. Zhou *et al.*[24] collected data from YouTube, and divide a video into segments by actions. In [13, 15], they downloaded videos collected from [16], and annotated segments with captions and actions. In this paper, we use KUSK Dataset [6], which consists of recipes and *unedited* videos. In terms of videos, this paper differs from [24] in whether videos are raw or not. In terms of recipes, it differs from [13, 15] in whether recipes are annotated or not.

2.1.2 **Applications.** Image and video captioning is an important application in this area. Ushiku *et al.* [19] proposed a model to generate a procedural text from a cooking video. Because the number

[1]https://cookpad.com/

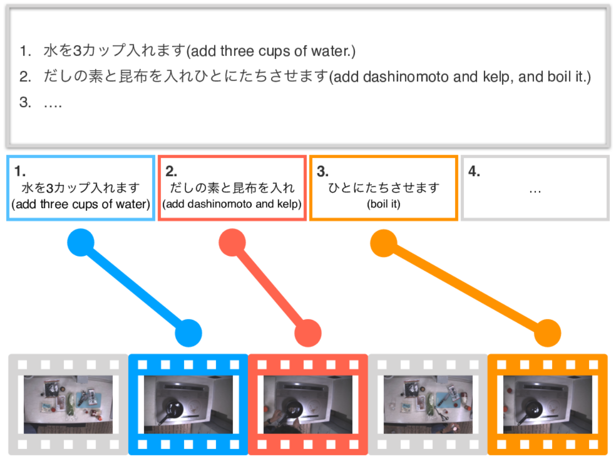

1. 水を3カップ入れます(add three cups of water.)
2. だしの素と昆布を入れひとにたちさせます(add dashinomoto and kelp, and boil it.)
3. ….
1. 水を3カップ入れます (add three cups of water)
2. だしの素と昆布を入れ (add dashinomoto and kelp)
3. ひとにたちさせます (boil it)
4. …

Figure 1: Task overview[2].

of pairs of cooking video and recipe is not large, they attempted to generate a recipe based on object detection results instead of taking so-called end-to-end approach. Salvador *et al.* [17] proposed joint embedding model between recipes, ingredients, and pictures. Their model achieved a great improvement compared with conventional methods in retrieval tasks. El *et al.* [2] proposed a model to generate a completed dish image from a recipe using StackGAN-v2 architecture [22].

2.2 Alignment of a Recipe and Cooking Video

There exists previous research which aims to align steps of procedural text with processing scenes in the video. Malmaud *et al.* [9] proposed that they obtained an alignment between recipe, narration, and video frames using hidden Markov model after they collected a large amount of data from YouTube. Their research differs in utilizing narration data. In this paper, we do not need any narration data. Another important difference is that their research utilizes edited video. In general, utilizing edited video is easier than unedited video to align because edited video is produced to teach procedure details to users. Bojanowski *et al.* [1] proposed a supervised method which predicts a range that procedure text were taking place. They input a procedure text, unedited video, and annotation which indicated range of actions, whereas, our method does not need this kind of annotation.

3 TASK OVERVIEW

3.1 Generation of Instructions with Pictures from Procedure Text and Video

Our goal is to produce recipe with pictures given a recipe and its execution video. Figure 1 shows an overview of our task.

An execution video, one of the inputs, records a sequence of activities to make or repair something from the beginning to the end. Thus in the videos only one person appears (mainly the hands only). In the beginning, there are some ingredients and tools on the

[2]All example sentences in this paper are quoted from the recipes uploaded to Cookpad, "アツアツとろ〜り白菜と鶏のスープ"(https://cookpad.com/recipe/121196), "しいたけとしめじのソテー"(https://cookpad.com/recipe/176550), and "コロッケ種の揚げ包み焼き"(https://cookpad.com/recipe/201826).

Table 1: Definition of r-NE tags.

r-NE tag	meaning
F	Food
T	Tool
D	Duration
Q	Quantity
Ac	Action by the chef
Af	Action by foods
Sf	State of foods
St	State of tools

cooking table and in the video later, some intermediate products appear in it. Finally, it finishes with a completed dish. The system selects frames for the instructions in the recipe for others to better understand them.

As an evaluation metrics, it is preferable to measure how much the output text helps another chef to produce the same dish. Thus, we have two people who cook on a daily basis select pictures from video frames. Note that the number of video frames among which they choose are approximately one hundred frames on average because they are extracted from a video as key frames (we describe this process in the subsequent section). After this, we calculate recall, precision, and F-measure comparing with automatic and manual results.

3.2 Prerequisites

3.2.1 **Instruction identification in recipe text.** First, we assume a set of terms (word sequences) called named entities (x-**NEs**) representing important object names in the target domain x. They are also objects detected by computer vision.

In the recipe case, noun phrases for ingredients and tools are important object names. In this paper, we adopt the recipe named entities (**r-NEs**) defined in [11], whose types are listed in Table 1. We use the notation "鶏肉/F" ("chicken/F") to indicate that "鶏肉 (chicken)" is an r-NE and its type is food (F). Although all of these types are important, we use only food (F) and tool (T) which can be directly detected by computer vision.

In order to extract terms for important objects from text, we must locate x-NEs in given sentences. So-called named entity recognizer (NER) is suitable for this task. In this paper, we adopt PWNER [18], NERs based on sequence labeling techniques that can be trained on an annotated corpus in the domain.

Each sentence of a recipe may consist of multiple actions, so it is not clear which actions to be aligned with a picture. As an example in our case, "鍋に 3 カップお水を入れ (Action) 味の素と昆布を入れ (Action) てひとにたちさせ (Action) ます。(Add three cups of water to a pot, and put flavors in it, and boil it.)[3]" have three actions, thus we separate this sentence into three instructions according to actions (Ac), and name it after "Action instruction (Ac instruction)".

3.2.2 **Key frame extraction using object detection in video.** Our method requires the module that can detect appearance and disappearance of materials and tools involved in each procedure. In the cooking video case, we use the Faster R-CNN model [14] fine-tuned with relatively small set of images of foods and cooking tools.

Based on this, we extract key frames from video. We assume that frames are important when they show chef picking up or placing objects, that are provided in KUSK Object Dataset [5] with object regions. Note that the provided frames and regions can contain multiple objects because the method used in [5] is based on a background subtraction. To divide the detected region into object-wise regions, we adopted the Faster R-CNN [14]. This neural network outputs identified object region as a rectangular area while recognizing its category. It also provides confidence as a probability. An example of visualized output is shown in Figure 3, where a cutting board and a knife are in the region detected by [5].

We utilized the Faster R-CNN's ability of object region identification to suppress another type of false detection. The regions provided in [5] contain objects that are moved only slightly by coming in contact with the hands.[4] Such objects should not be related to the procedure. To suppress such detection but spot only objects obviously related to the procedure, we compare the location of object-wise regions before and after the contact, and ignore object regions if they have the same object name and have a certain score in Jaccard index, which is general method to measure the size of intersection of two regions. After the test of region intersection, only the objects with an obvious location change are regarded as procedure-related. In this manner, we extract key frames related to the procedure.

4 PROPOSED METHOD

In this section, we explain the proposed method for selecting a video frame for each instruction. The outline of this method is shown in Figure 2. We assume that a frame is suitable for an instruction when key objects appearing in them are common. Under this assumption, first, we extract the information of key objects using named entity recognizer and object detection, and map Ac instructions and key frames into a common feature space (Figure 2 A). Next, to measure how suitable they are, we calculate the matching score using these vectors (Figure 2 B). Finally, we align the Ac instructions and key frames using the Viterbi algorithm (Figure 2 C). The recurrence formula is defined for reflecting on the assumption that the chef does not change the order of instructions.

4.1 Conversion to Feature Vectors

Below we describe how to convert key frames and Ac instructions into feature vectors for the similarity calculation in Figure 2.

4.1.1 **key frame to vector.** We convert key frames into feature vectors $\boldsymbol{v}^{fr}$ by the following steps. First, based on the Faster R-CNN, we detect objects in each frame. We see these results in Figure 3. In this step, we can obtain information of important objects, such as location and probability to each object category. Let O_t^d be a set of objects detected in a key frame t and classified into d-th object category. d-th element of the feature vector $\boldsymbol{v}^{fr}$ is calculated

[3]The language resources used in our experiments are in Japanese. Thus, these sentences are just for reference. Note that the proposed method can produce a recipe with pictures in other language by preparing the the prerequisites in that language.

[4]We note that all those processes to KUSK Object Dataset were done automatically.

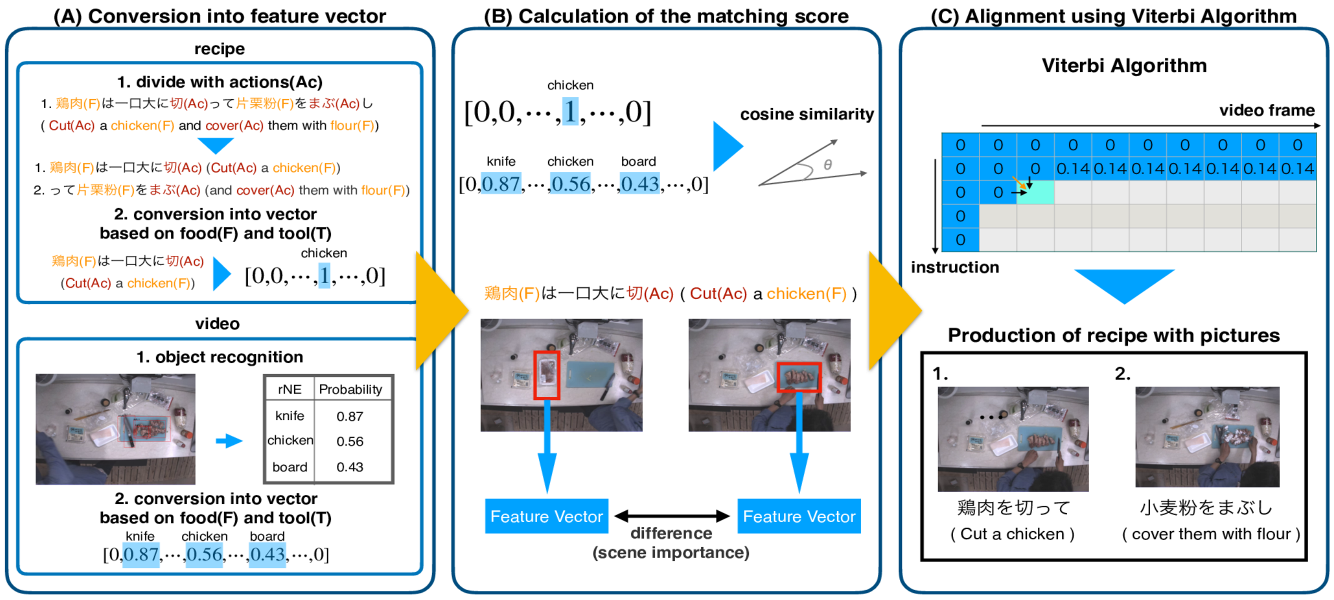

(A) Conversion into feature vector
recipe
1. divide with actions(Ac)
1. 鶏肉(F)は一口大に切(Ac)って片栗粉(F)をまぶ(Ac)し
(Cut(Ac) a chicken(F) and cover(Ac) them with flour(F))
1. 鶏肉(F)は一口大に切(Ac) (Cut(Ac) a chicken(F))
2. って片栗粉(F)をまぶ(Ac) (and cover(Ac) them with flour(F))
2. conversion into vector based on food(F) and tool(T)
chicken
[0,0,⋯,1,⋯,0]
video
1. object recognition
rNE Probability
knife 0.87
chicken 0.56
board 0.43
2. conversion into vector based on food(F) and tool(T)
knife chicken board
[0,0.87,⋯,0.56,⋯,0.43,⋯,0]
(B) Calculation of the matching score
cosine similarity
鶏肉(F)は一口大に切(Ac) (Cut(Ac) a chicken(F))
Feature Vector
difference
(scene importance)
(C) Alignment using Viterbi Algorithm
Viterbi Algorithm
video frame
instruction
Production of recipe with pictures
1. 鶏肉を切って (Cut a chicken)
2. 小麦粉をまぶし (cover them with flour)

Figure 2: Overview of the proposed method.

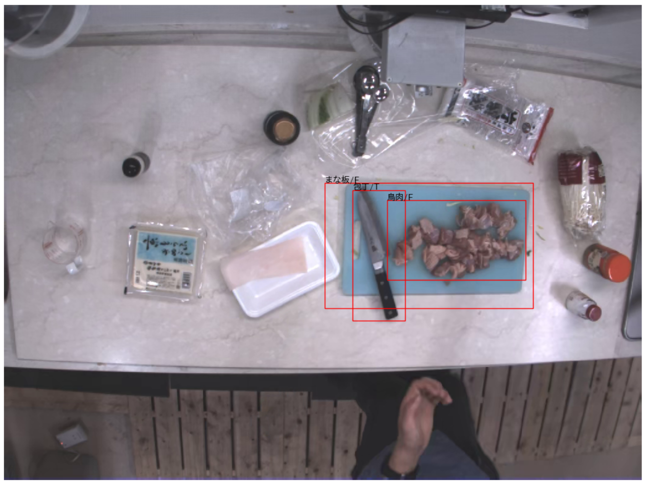


Figure 3: Example of an object detection result.

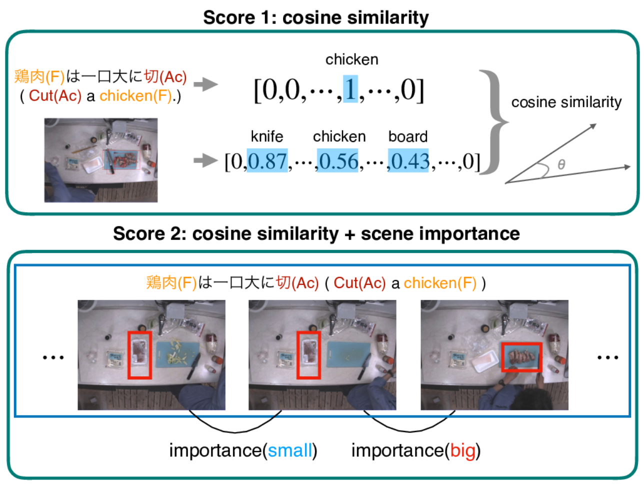


Figure 4: Outline of calculation of the matching score.

by

$$v_d^{fr} = \begin{cases} 0 & (O_t^d = \phi) \\ \max_{o \in O_t^d} \Pr(d|o) & (\text{otherwise}) \end{cases}, \tag{1}$$

where $\Pr(d|o)$ is the probability obtained by the Faster R-CNN.

4.1.2 ***Ac* instruction to vector.** We convert each *Ac* instruction into a vector $\boldsymbol{v}^{in}$ by the following steps. First, in order to identify orthographic variant, we use cooking ontology [12] and normalize words. Second, based on PWNER [18] trained with a cooking corpus [11], we extract information of important objects in the *Ac* instruction. Finally, based on the information, we set the elements in the vector to 1.0 corresponding to the objects in the instruction. And the other values are set to be 0.0. From this transformation, we can obtain n-hot vectors $\boldsymbol{v}^{in}$ from all *Ac* instructions in the recipe.

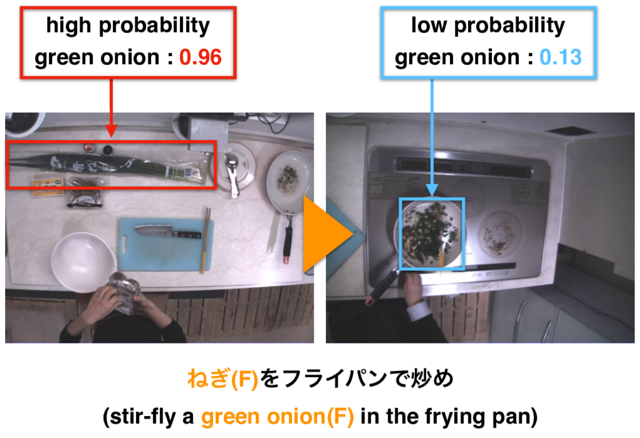
4.2 Matching Score Calculation

Based on the above feature vectors, we obtain the best alignment using the Viterbi algorithm. To use the vectors in this alignment, we need to calculate matching score for each pair of a key frame and an *Ac* instruction. The outline of the matching score calculation is shown in Figure 4, which we explain in detail below.

4.2.1 **cosine similarity.** Using feature vectors, we calculate the cosine similarity $sim(\boldsymbol{v}^{fr}, \boldsymbol{v}^{in})$ as

$$sim(\boldsymbol{v}^{fr}, \boldsymbol{v}^{in}) = \frac{\boldsymbol{v}^{fr} \cdot \boldsymbol{v}^{in}}{||\boldsymbol{v}^{fr}||_2 ||\boldsymbol{v}^{in}||_2}. \tag{2}$$

4.2.2 **scene importance.** Because foods change their appearance drastically while cooking, the probability in object detection can



high probability
green onion : 0.96
low probability
green onion : 0.13
ねぎ(F)をフライパンで炒め
(stir-fly a green onion(F) in the frying pan)

Figure 5: Example of drop in detection probability.

easily drop down. An example is given in Figure 5. The frame corresponds to an *Ac* instruction of "ねぎをフライパンで炒め (stir-fly a green onion in the frying pan)". When cut green onions are in the pan, the detection probability drops precipitously. In contrast, if they are in a typical state (e.g. before cut or in the package), the probability becomes quite stable. This can be an unwanted bias in the similarity calculation. To compensate this bias, we focus on changes in visual appearance of objects. When they are involved to a process, their appearance in a key frame will change from that in the previous key frame. This reduces the chance of selection by objects in typical state and increases that by objects with unstable appearance.

Let $z^o_{t_1}$ be a feature extracted from the region of object o detected in a key frame t_1. Let also t_2 be the frame in which o was lastly detected before the frame t_1. The scene importance si_{t_1} is calculated as an average of the Euclidean difference $||z^o_{t_1} - z^o_{t_2}||_2$ among all object detected at the frame.

$$si_{t_1} = \frac{1}{|O_{t_1}|} \sum_{o \in O_{t_1}} \frac{||z^o_{t_1} - z^o_{t_2}||_2}{Z}, \tag{3}$$

where Z is a normalizing factor given as the maximum value of $||z^o_{t_1} - z^o_{t_2}||_2$ among all the pairs t_1 and t_2 in the video and over all the objects. Using the scene importance si_t, we calculate the matching score for the Viterbi algorithm simply as

$$score = si_t + sim(\boldsymbol{v}^{fr}, \boldsymbol{v}^{in}) \tag{4}$$

4.3 Alignment using the Viterbi Algorithm

We find the best alignment between key frames and *Ac* instructions based on the matching score described above. Assuming that the text and the video are aligned monotonically (i.e. no reordering), the Viterbi algorithm, one of the dynamic programming (DP) algorithm, allows us to find the matching of the highest score between *Ac* instructions and key frames as the path from the top left to the bottom right at the speed linear to the input lengths.

Let i be the index of *Ac* instructions, and j be the index of key frames. *table* means the DP table, and $table[i][j]$ shows the similarity of a combination of i-th *Ac* instruction and j-th key frame. The following is the recurrence formula to fill each cell in the DP table.

$$table[i][j] = \begin{cases} 0 & (i = 0 \lor j = 0) \\ \max \left\{ \begin{array}{l} table[i-1][j-1] + score, \\ table[i-1][j], \\ table[i][j-1] \end{array} \right\} & (otherwise) \end{cases}$$

5 EVALUATION

We produce a recipe with pictures given a recipe text and its execution video. In our experiment, we conducted the following three methods:

(a) cos (baseline 1)
(b) cos+Viterbi (baseline 2)
(c) cos+Viterbi+scene

In (a), based on the feature vectors, we calculate the cosine similarity between each *Ac* instruction and every key frame, and select the frame with the highest similarity for the instruction. In (b), we calculate the cosine similarity and regard this as the matching score. Based on this score, we obtain alignment using the Viterbi algorithm. In (c), we add the scene importance to the cosine similarity and regard this as the matching score. Similar to (b), based on this score, we obtain the best alignment using the Viterbi algorithm.

To compare the performance of above three methods, we prepared the ground truth based on manual frame selections for each *Ac* instruction by two annotators. The two annotators selected frames which can be suitable to an *Ac* instruction based on the following rules:

- Select frames suitable for *Ac* instruction among key frames.
- Select frames in which the action happens at the center.
- They may select multiple frames. They may also select no frame if there are no suitable one.

We calculate the recall, precision, and F-measure as a measure of alignment evaluation, and compare our method with baseline as well as with manual results. Let $\hat{y}_n$ be the index of the frame selected for the n-th *Ac* instruction by the proposed method, and Y^i_n be the ordered set of the indices of frames selected by i-th human annotator for n-th *Ac* instruction. The recall, precision, and F-measure are calculated as

$$recall = \frac{1}{2} \sum_{i=\{1,2\}} \frac{\sum_{n=1}^{N} \mathbb{1}(\hat{y}_n \in Y^i_n)}{\sum_{n=1}^{N} \mathbb{1}(|Y^i_n| > 0)} \tag{5}$$

$$precision = \frac{1}{2} \sum_{i=\{1,2\}} \frac{\sum_{n=1}^{N} \mathbb{1}(\hat{y}_n \in Y^i_n)}{\sum_{n=1}^{N} \mathbb{1}(\hat{y}_n > 0)} \tag{6}$$

$$F\text{-}measure = \frac{2 recall \cdot precision}{recall + precision}, \tag{7}$$

where $\mathbb{1}$ is the indicator function, N is the number of *Ac* instructions, and $\hat{y}_n = 0$ when the method selects no frame.

5.1 Annotation Results

In our experiments, we used 12 videos in KUSK Dataset [6]. Table 2 shows the annotation results. In this table, using the notations described above, the "AND" and "OR" are counted as follow:

Table 2: Manual annotation results.

	Annotator 1	Annotator 2	AND	OR
# of *Ac* instructions w/ frames	130	125	112	143
# of *Ac* instruction w/o frames	68	73	86	55
total	198			

Table 3: Evaluation results.

Method	Recall	Precision	F-measure
cos	0.046	0.059	0.052
cos+Viterbi	0.099	0.134	0.114
cos+Viterbi+scene	0.135	0.176	0.153
manual	0.749	0.872	0.805

$$AND = \sum_{n=1}^{N} \mathbb{1}(|Y_n^1 \cap Y_n^2| > 0) \tag{8}$$

$$OR = \sum_{n=1}^{N} \mathbb{1}(|Y_n^1 \cup Y_n^2| > 0) \tag{9}$$

When calculating the recall, precision, and F-measure, we only consider the match of *Ac* instructions with frames by the proposed method and the annotators. From "OR" results, It amounts for 72% (=143/198) that there are *Ac* instructions with frames in all annotated *Ac* instructions. Note that most of the *Ac* instructions without frames are auxiliary (for example, indisputable instructions such as "火を止め (turn off the heat)", or options such as "ご飯を入れても美味しいですよ (it is also delicious if you add rice)". This finding indicates that it is possible to evaluate the proposed method comparing the annotation results as the ground truth data.

5.2 Implementation Details

We fine-tuned the Faster R-CNN using the KUSK Object Dataset [5], which gives annotation of bounding boxes and object names in videos of the dataset. For the scene importance score calculation, we used ResNet-50 [7] as a pre-trained neural network.

5.3 Results

Table 3 shows the recall, precision, and F-measure of the methods. To evaluate a variation in manual alignments done by different annotators, their intersection is hired as the metric. Further detail of the recall and precision scores for each video-recipe pair are given in Figures 6 and 7, respectively. It consists of 12 videos and recipes, which is used in our experiment. From this result, we can see that the method considering the scene importance outperforms other automatic methods. We show that it is effective to calculate the scene importance.

In our experiment, we allow annotators to choose attach multiple frames or no frame to each *Ac* instructions. To clarify the relation between accuracy and the number of attached frames, we measure the change of accuracy in increment of the number of attached frames to each *Ac* instruction in Figure 8.

Note that there is the small number of attached frames to each *Ac* instruction, so the graph is not smooth. We set the max number to 15 because the graph changes as the same way when the number of attached frame is more than 8.

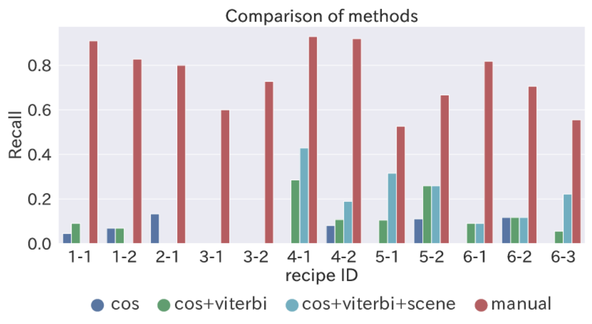


Figure 6: Detailed comparison in recall.

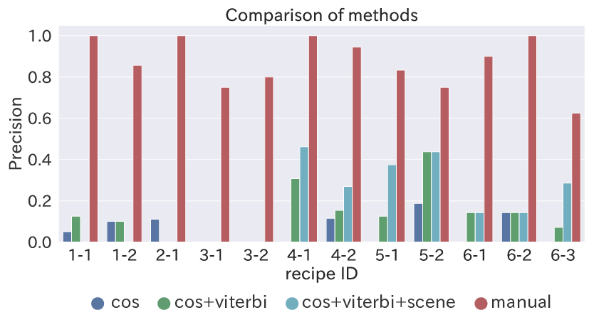


Figure 7: Detailed comparison in precision.

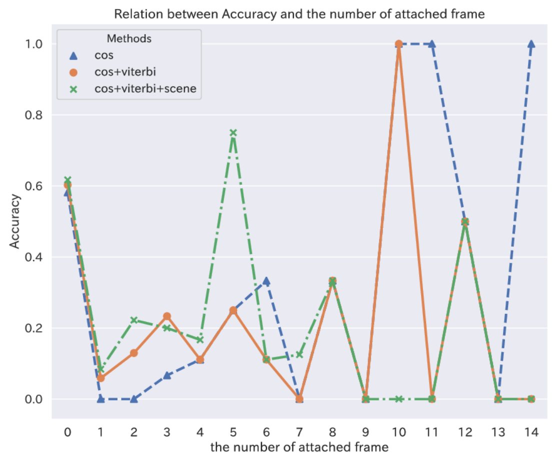


Figure 8: Variation in accuracy against the number of attached frames.

6 DISCUSSION

In this section, we give a more detailed discussion for comparing the results with the manual selections and clarify pros and cons of our methods qualitatively.

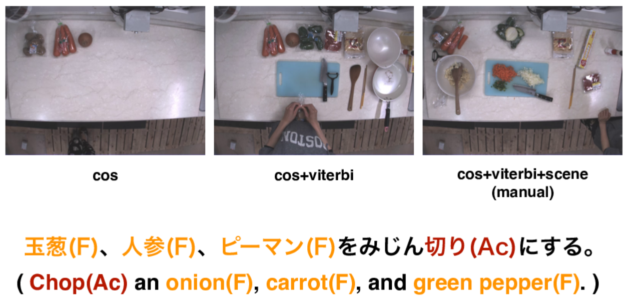

cos
cos+viterbi
cos+viterbi+scene (manual)
玉葱(F)、人参(F)、ピーマン(F)をみじん切り(Ac)にする。
(Chop(Ac) an onion(F), carrot(F), and green pepper(F).)

Figure 9: Example of influence of considering the scene importance.

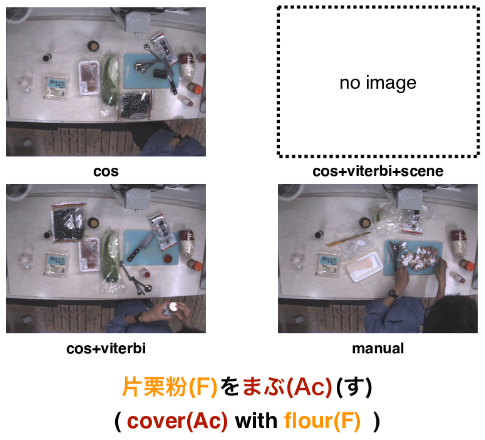


Figure 10: Example of mistake of the Faster R-CNN. Note that we convert the *Ac* instruction into end-form using the subsequent *Ac* instruction.

6.1 Variation in Accuracy against the Number of Manually Selected Frames

In the manual selection, annotators were allowed to select several frames for each *Ac* instruction. In other words, the difficulty of frame selection is different depending on the number of manually selected frames; the more frames are selected, the more the method can output one of them by chance. Figure 8 plots the difference in accuracy against the number of manually selected frames. If the method selected frames randomly, we would find a tendency of increasing accuracy along to the number of frames. We see such a tendency in the results of cos and cos+Viterbi. In contrast, the results of cos+Viterbi+scene achieve higher accuracy even at smaller numbers of manually selected frames. This indicates that the method cos+Viterbi+scene selects frames more accurately than the two baselines.

6.2 Impact of Scene Importance

Table 3 shows an improvement in performance by introducing both the Viterbi algorithm and the scene importance quantitatively. Hence, we can safely say that it is effective to use these ideas in this task.

In cooking, foods often change their appearance by being cut (or processed in various ways). Hence, it is difficult even with recent object detection method like the Faster R-CNN to detect them stably. In such cases, the object detector can give a low detection probability to processed foods. In other words, a naive method, such as the baselines, may select frames with unprocessed food more than processed one. The scene importance has an effect to correct for the influence of such low detection probability at processed foods.

Figure 9 shows an example of the error correction. Only the method of cos+Viterbi+scene could select the correct frame, while the other two baselines could not. In this case, any of selected frames contain onions, green pepper, and carrots; however only non-processed foods are in the frames selected by cos and cos+Viterbi, while cos+Viterbi+scene selected one with processed foods.

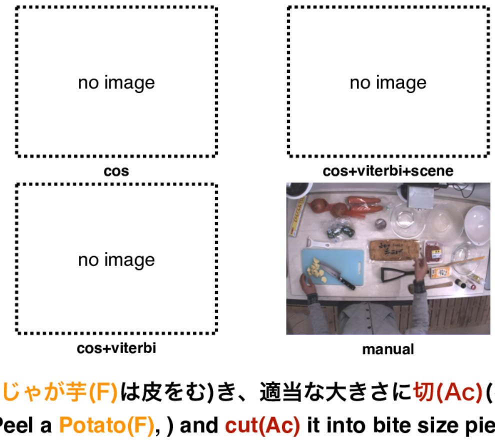

no image
cos
no image
cos+viterbi+scene
no image
cos+viterbi
manual
(じゃが芋(F)は皮をむ)き、適当な大きさに切(Ac)(る)
((Peel a Potato(F),) and cut(Ac) it into bite size pieces)

Figure 11: Example of omission in a recipe. Note that we convert the *Ac* instruction into end-form using the subsequent *Ac* instruction.

6.3 Limitation of the Method

From Table 3, one can say that there is still a large gap between automatic and manual frame selection. One considerable cause of the gap is the sparseness of vectors that the proposed method uses for suitability calculation. Because we calculate the vector from two different sources, videos and texts, we can see two typical failures from them.

Figure 10 shows a typical mistake caused in object detection. Flour changes their appearance drastically when it comes out from the pack. Hence, such foods can be easily missed by the detector, and flour does not appear in the vector that represents the frame. To overcome this problem, first, we should have a larger dataset than KUSK Object Dataset [5], which is used to fine-tune the Faster R-CNN. Second, we should get a dense feature space that represents not only independent objects, but their relationship together (e.g. something powder on chicken). Such a feature space does not necessarily detect the exact object name, but would match a frame to instruction under the context given by the Viterbi algorithm.

The second typical case is caused in extracting the vector representation from text. In the case shown in Figure 11, the word "じゃが芋/F" ("potato/F") highlighted in yellow, is included in the preceding *Ac* instruction, but not in the target *Ac* instruction of "cut (*Ac*)". The proposed method yet does not take such omissions and zero anaphora into account. In [10], they analyzed recipes and found that there are a lot of omissions and zero anaphora in recipes because the preceding objects are used in the following steps and we do not need to write it. There are many approaches to tackle this problems [10]. In our task, it would be effective to use the flow graph [8]. It gives a relationship of an *Ac* instruction and its preceding *Ac* instructions. Hence, using flow graph, we would be able to complement all objects involved to the target instruction. This is also reserved as future work.

7 CONCLUSION

In this paper, we proposed a problem of selecting frame from its execution video for each instruction to produce a recipe with pictures. We assumed that a frame is suitable to an *Ac* instruction when key objects appearing in the frame and instruction are common. Under this assumption, using named entity recognizer for text, and object detection for video, we convert them into feature vectors which represent appearance of objects. Then, the method selects a suitable frame for each instruction based on the cosine similarity of those feature vectors and the Viterbi algorithm.

In addition to it, to obtain a more sophisticated result, we proposed the scene importance; an amount of changes in the appearance of key objects from previous scenes. By adding this score to the cosine similarity, the ablation study confirmed that the method successfully improve the accuracy against the cases without the scene importance.

However, there still exists a large gap between manual and automatic frame selection. The gap would be derived from poor object detection and named entity recognition, or omissions and zero anaphora in an *Ac* instruction. One future direction is a use of verb information. Currently, verbs can be extracted from text using the current preprocessing of named entity recognition, but not from video. We will make use of action information in video via deep-learning based approaches in order to involve information of verbs into the similarity [17, 21, 23].

ACKNOWLEDGEMENTS

In this paper, we used recipe data provided by Cookpad and the National Institute of Informatics. The work is supported by JSPS Grants-in-Aid for Scientific Research Grant Number 18K11425 and Cookpad Inc.

REFERENCES

[1] Piotr Bojanowski, RÃĺmi Lajugie, Edouard Grave, Francis Bach, Ivan Laptev, Jean Ponce, and Cordelia Schmid. 2015. Weakly-supervised alignment of video with text. *In Proceedings of the International Conference on Computer Vision* (2015), 4462–4470.
[2] Ori Bar El, Ori Licht, and Netanel Yosephian. 2019. GILT: Generating images from long text. *arXiv preprint arXiv:1901.02404* (2019).
[3] Zhao Guo, Lianli Gao, Jingkuan Song, Xing Xu, Jie Shao, and Heng Tao Shen. 2016. Attention-based LSTM with semantic consistency for videos captioning. *In Proceedings of the ACM international conference on Multimedia.* (2016), 357–361.
[4] Jun Harashima, Yuichiro Someya, and Yohei Kikuta. 2017. Cookpad Image Dataset: An image collection as infrastructure for food Research. *In Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval* (2017), 1229–1232.
[5] Atsushi Hashimoto, Masaaki Iiyama, Shinsuke Mori, and Michihiko Minoh. 2016. KUSK Object Dataset: Recording access to objects in food preparation. *In the Proceedings of IEEE International Conference on Multimedia and Expo Workshops* (2016), 1–6.
[6] Atushi Hashimoto, Tetsuro Sasada, Yoko Yamakata, Shinsuke Mori, and Michihiko Minoh. 2014. KUSK dataset: Toward a direct understanding of recipe text and human cooking activity. *In the Proceedings of the ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication* (2014), 583–588.
[7] 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* (2016), 770–778.
[8] Hirokuni Maeta, Tetsuro Sasada, and Shinsuke Mori. 2015. A framework for procedural text understanding. *In Proceedings of the International Conference on Parsing Technologies* (2015), 50–60.
[9] Jonathan Malmaud, Jonathan Huang, Vivek Rathod, Nick Johnston, Andrew Rabinovich, and Kevin Murphy. 2015. What's cookin'? interpreting cooking videos using text, speech and vision. *In Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics* (2015), 143–152.
[10] Jonathan Malmaud, Earl J. Wagner, Nancy Chang, and Kevin Murphy. 2014. Cooking with semantics. *In Proceedings of the ACL Workshop on Semantic Parsing* (2014), 33–38.
[11] Shinsuke Mori, Hirokuni Maeta, Yoko Yamakata, and Tetsuro Sasada. 2014. Flow graph corpus from recipe texts. *In Proceedings of the International Conference on Language Resources and Evaluation* (2014), 2370–2377.
[12] Hidetsugu Nanba, Yoko Doi, Miho Tsujita, Toshiyuki Takezawa, and Kazutoshi Sumiya. 2014. Construction of a cooking ontology from cooking recipes and patents. *In the Proceedings of the ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication* (2014), 37–42.
[13] 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* (2013), 25–36.
[14] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. *In Advances in Neural Information Processing Systems* (2015), 91–99.
[15] Anna Rohrbach, Marcus Rohrbach, Wei Qiu, Annemarie Friedrich, Manfred Pinkal, and Bernt Schiele. 2014. Coherent multi-sentence video description with variable level of detail. *In Proceedings of the German Conference on Pattern Recognition* (2014), 184–195.
[16] Marcus Rohrbach, Sikandar Amin, Mykhaylo Andriluka, and Bernt Schiele. 2012. A database for fine grained activity detection of cooking activities. *In Proceedings of the Computer Vision and Pattern Recognition* (2012), 1194–1201.
[17] 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 Conference on Computer Vision and Pattern Recognition* (2017), 3020–3028.
[18] Tetsuro Sasada, Shinsuke Mori, Tatsuya Kawahara, and Yoko Yamakata. 2015. Named entity recognizer trainable from partially annotated data. *In Proceedings of the Conference of the Pacific Association for Computational Linguistics* (2015), 148–160.
[19] Atsushi Ushiku, Hayato Hashimoto, Atsushi Hashimoto, and Shinsuke Mori. 2017. Procedural text generation from an execution video. *In Proceedings of the International Joint Conference on Natural Language Processing* (2017), 326–335.
[20] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. *In Proceedings of the Conference on Computer Vision and Pattern Recognition* (2015), 3156–3164.
[21] 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* (2016), 5005–5013.
[22] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris Metaxas. 2017. Stackgan++: Realistic image synthesis with stacked generative adversarial networks. *arXiv preprint arXiv:1710.10916* (2017).
[23] Ying Zhang and Huchuan Lu. 2018. Deep cross-Modal projection learning for image-text matching. *In Proceedings of the European Conference on Computer Vision* (2018), 686–701.
[24] Luowei Zhou, Chenliang Xu, and Jason J. Corso. 2018. Towards automatic learning of procedures from web instructional videos. *In Proceedings of the Advancement of Artificial Intelligence* (2018), 7590–7598.