Visual Recipe Flow: A Dataset for Learning Visual State Changes of Objects with Recipe Flows

Keisuke Shirai[1] **Atsushi Hashimoto**[2] **Taichi Nishimura**[1] **Hirotaka Kameko**[1]
Shuhei Kurita[3] **Yoshitaka Ushiku**[2] **Shinsuke Mori**[1]
[1]Kyoto University [2]OMRON SINIC X Corporation [3]RIKEN AIP, JST PRESTO
{shirai.keisuke.64x,nishimura.taichi.43x}@st.kyoto-u.ac.jp
{atsushi.hashimoto,yoshitaka.ushiku}@sinicx.com
{kameko,forest}@i.kyoto-u.ac.jp shuhei.kurita@riken.jp

Abstract

We present a new multimodal dataset called Visual Recipe Flow, which enables us to learn each cooking action result in a recipe text. The dataset consists of object state changes and the workflow of the recipe text. The state change is represented as an image pair, while the workflow is represented as a recipe flow graph (r-FG). The image pairs are grounded in the r-FG, which provides the cross-modal relation. With our dataset, one can try a range of applications, from multimodal commonsense reasoning and procedural text generation.

1 Introduction

Our aim is to track how foods are processed and changed toward the final food product by each cooking action given a recipe text. This requires some knowledge of the actions: what foods and actions are involved and how the action changes them. Skilled chefs can easily imagine these action effects while understanding the required foods. We are interested in building an autonomous agent endowed with this ability, as illustrated in Figure 1. This example involves two cooking actions, and the agent imagines the second action result: the shredded cabbage in the bowl. This also implicates the food requirement: the shredded cabbage produced by the previous action. The prediction for the required foods and action results is indeed a natural ability for humans when they cook something. Thus, this is also crucial for intelligent autonomous agents to understand recipe texts.

Previous work on this line of research provided visual annotation for each cooking instruction (Nishimura et al., 2020; Pan et al., 2020). Nishimura et al. (2020) attached an image with bounding boxes of objects to each instruction, while Pan et al. (2020) split an instruction into sentences and attached frames to each sentence. However, their annotations are often insufficient to predict the action result for each object. A typical

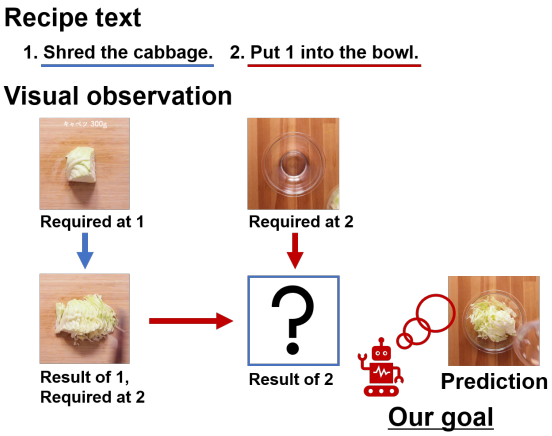

Figure 1: Our goal is to build an agent that tracks object state changes and predicts what observations can be obtained by cooking actions.

case is an instruction in a sentence that directs multiple actions. For example, the instruction of "slice the tomato and put it into the bowl" produces two action results: the sliced tomato and that put in the bowl. Therefore, an instruction-wise visual annotation is insufficient for our task, and action-wise visual annotation is required. Preparing a more dense visual annotation is one straightforward way to handle this case.

Toward the realization of an agent that predicts the result of each action, we introduce a new multimodal dataset called Visual Recipe Flow (VRF). The dataset consists of object[1] state changes caused by every action and the workflow of the text. The change is given as an image pair, while the workflow is given in the format of recipe flow graph (r-FG) (Mori et al., 2014). Each image pair is grounded in the r-FG, which gives the cross-modal relation. Figure 2 shows an example of our dataset.

We focus on recipe text involving various cooking actions, foods, and state changes, which is one of the representatives of procedural texts. Un-

[1]In our work, object refers to food or tool.

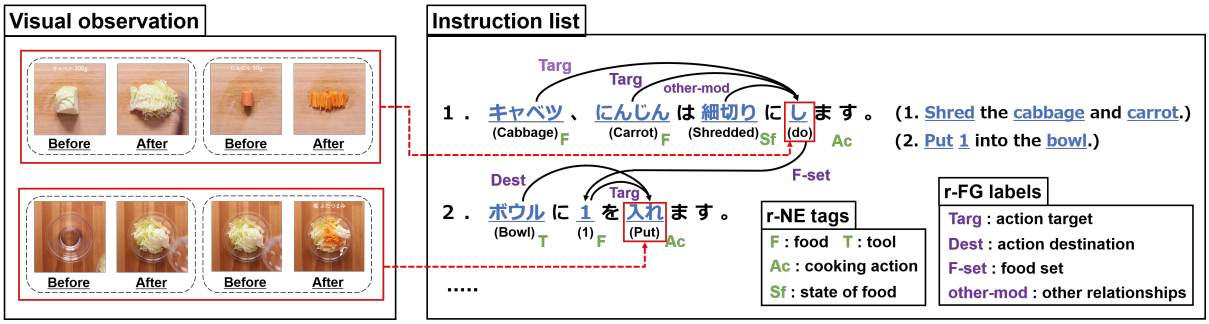

Figure 2: Example of our dataset. A pair of images in the visual observation corresponds to the states of object before and after a cooking action. They are grounded in the action in the instruction list. The black solid arrows denote recipe flows, which describe the relationships between expressions (e.g., cooking actions, foods, and tools).

derstanding these texts by tracking object state changes is one of the recent trends (Dalvi et al., 2018; Bosselut et al., 2018; Tandon et al., 2020; Nishimura et al., 2021; Papadopoulos et al., 2022). Our work also contributes to this line of research. Since images directly express object appearances in the real world (Isola et al., 2015; Zhang et al., 2021), our dataset would provide rich information for the changes. The sequential nature of our dataset can also be used to test the reading comprehension ability of large-scale language models (Srivastava et al., 2022). Furthermore, since our dataset has arbitrary interleaved visual and textual annotations, it is also possible to evaluate the few-shot capability of vision-language models on such data (Alayrac et al., 2022).

2 The VRF dataset

The Visual Recipe Flow (VRF) dataset is a new multimodal dataset. It provides visual annotations for objects in a recipe text before and after a cooking action. We identify expressions including the action in the text by using recipe named entities (r-NEs) (Mori et al., 2014), which can be extended to other procedural tasks. Based on the r-NEs, it also provides a representation of the recipe workflow as a recipe flow graph (r-FG) (Mori et al., 2014). In this section, we first explain the overview of the r-FG and then introduce our visual annotation.

2.1 Recipe flow graph (r-FG)

The r-FG represents the cooking workflow of a recipe text. It consists of a set of recipe flows. The recipe flow is expressed as a directed edge that takes two r-NEs as the starting and ending vertices. It also has a label that describes the relationship between them. It connects one cooking action with the next and expresses its dependencies. For example, in Figure 2, the first action is connected with the second one, which means that the second action requires the products of the first action: shredded cabbage and carrot. This helps us to identify what foods are required for the actions. The annotation has the flows from the ingredient lists to track foods from raw ingredients (Nishimura et al., 2021), which allows us to convert the r-FG into cooking programs (Papadopoulos et al., 2022).

2.2 Visual annotation

Our visual annotation is given as an extension of the r-FG. Each annotation consists of a pair of images which represent object state change by the action. Each image pair is linked with the action in the r-FG. In some cases, a single action can require multiple objects and change their states. Our annotation provides an image pair to all of these state changes. In Figure 2, for example, the first action is linked with two image pairs because it induces the state changes of two objects: cabbage and carrot. This dense annotation would help develop autonomous cooking agents because these images provide visual clues for each action.

3 Annotation standards

In this section, we describe our annotation standards. The annotation consists of three steps in order: (i) r-NE annotation, (ii) r-FG annotation, and (iii) image annotation. Each recipe has an ingredient list, an instruction list, and a cooking video. Figure 3 shows an example of the annotations.

r-NE annotation. First, we annotated words in the ingredient and instruction lists with r-NE tags[2].

[2]We segmented sentences into words beforehand by using a Japanese tokenizer, KyTea (Neubig et al., 2011), because words in a Japanese sentence are not typically separated by whitespace.

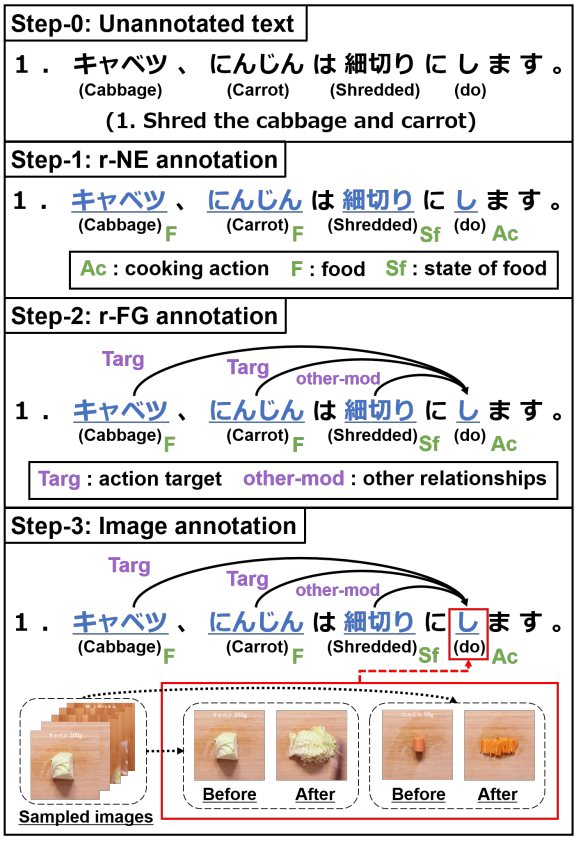

Figure 3: Example of annotation process for a single instruction. The instruction is sequentially annotated with r-NE tags, recipe flows, and images.

i

We used the eight types of r-NE tags, following Mori et al. (2014). See Appendix A for details.

r-FG annotation. Second, we annotated the r-NEs in the first step with the r-FG. We used the 13 types of r-FG labels, following Maeta et al. (2015). See Appendix A for details.

Image annotation. Third, we annotated object states with images, sampled at 3 frames per second from the videos. Each object required for any cooking action is annotated as a pair of frames of states before and after the action. When there are multiple suitable frames, we prioritize the one based on the visual clarity of the object. In some cases, objects are always heavily covered by human hands or abbreviated from the video. We treat them as missing data.

4 Annotation results

This section first describes our annotation process and the statistics for the annotation results. It then investigates the dataset quality and finally assesses our dataset by conducting experiments.

4.1 Annotation process

We started by collecting recipes and cooking videos since the existing r-FG datasets (Mori et al., 2014; Yamakata et al., 2020) are not necessarily associated with the videos. We collected 200 recipes in Japanese and videos from the Kurashiru website[3]. In the video, each cooking process is recorded in detail by a fixed camera. Thus, we can annotate the object states with a fixed viewpoint. Considering the future cooking agent developments, we focused on salad recipes, in which the procedures are simple but still contain 89 unique expressions for cooking action and 275 unique ingredients.

We asked one Japanese annotator, familiar with the r-NE and r-FG, to annotate the recipes. However, filling spreadsheets manually (Mori et al., 2014) is heavy, and it also might cause unexpected annotation errors. Therefore, we developed a web interface to help the annotation. The interface supports all three annotation steps. With this interface, the annotator can annotate recipes with r-NE tags, r-FG labels, and images by simple mouse operations. An illustration of the interface is provided in Appendix B. The whole annotation took 120 hours.

In the annotation collection process, we created annotation guidelines to check annotation errors and reproduce high-quality annotations by another annotator. Starting with a draft, we iteratively revised the guidelines when the first 10, 20, and 50 recipe annotations were finished. In the verification process, we shared the guidelines and three annotation examples with the second annotator.

4.2 Statistics

The recipes contained 1,701 ingredients, 1,077 instructions, and 33,400 words in total. The average number of ingredients and instructions per recipe was 8.51 and 5.31, respectively. The r-NE annotation resulted in 11,686 r-NEs, while the r-FG annotation resulted in 11,291 recipe flows. We provide the detailed statistics for them in Appendix A.

Table 1 shows the statistics for the image annotation results. We annotated 3,705 objects in the r-FGs with images. Among them, 2,551 had both pre-action and post-action images, 485 had only a post-action image, 72 had only a pre-action image, and 597 had no image. In total, 5,659 images (3,824 unique images) were used.

[3] https://www.kurashiru.com, accessed on 2021/12/14.

Annotated image		# objects
Pre-action state	Post-action state	
		597
✓		72
	✓	485
✓	✓	2,551
Total		3,705

Table 1: Statistics for the image annotation results. Objects have image annotation of a pre-action or post-action state if it is checked.

Annotation	Precision	Recall	F-measure
r-NE	97.93	98.88	98.40
r-FG	86.18	86.04	86.11
Image	75.13	70.60	72.80

Table 2: Inter-annotator agreements of the annotations.

4.3 Dataset quality

To investigate the correctness and consistency of the annotation results, we asked another annotator to re-annotate 10 recipes, which were randomly sampled from the collected recipes and contained 623 named entity tags, 616 recipe flows, and 199 visual state changes. We then measured the inter-annotator agreements in precision, recall, and F-measure. The agreements were calculated between the two sets of annotations by taking the first one as the ground truth.

Table 2 lists the results. The F-measure for the r-NE was 98.40, which was almost perfect agreement. The F-measure for the r-FG was 86.11, which was also quite high considering that all the r-NEs were presented as candidate vertices. The F-measure for the images was 72.80, which was smaller than the former steps. However, this was still high, considering that annotation differences in the former steps affected this step.

4.4 Experiments

We conducted multimodal information retrieval experiments to assess our dataset. The experiments aimed to find a correct post-action image from a set of candidate images by using the cooking action verb and pre-action image information. We used a joint embedding model (Miech et al., 2019) and briefly explain the calculation here[4]. We calculated a vector for an estimated post-action object state from the action verb and pre-action image information. This vector is mapped into a shared embedding space. On the other hand, the candidate post-action images are mapped into vectors and mapped them into the embedding space. We searched for the correct post-action image from the estimated post-action state based on their similarities in the embedding space.

[4]See details in Appendix C

Used input		R@5 (↑)	MedR (↓)
Action verb	Pre-action image		
		2.37	149.00
✓		21.24	26.70
	✓	33.77	12.60
✓	✓	37.01	10.40

Table 3: R@5 and MedR for the models with different inputs. The model uses action verb or pre-action image if it is checked. The first line denotes random search.

Our model was trained with different input configurations. We used the Recall@5 (R@5) and the median rank (MedR) as evaluation metrics. Table 3 shows the results. The second and third lines' scores show that the image provides more information than the text. The fourth line's scores imply that the textual and visual modalities provide different information, and using them together is more effective. These results demonstrate that the visual modality provides critical information for finding post-action images. These also indicate the usefulness of our annotation.

5 Application

5.1 Multimodal commonsense reasoning

Multimodal commonsense reasoning in recipe text is one of the recent trends (Yagcioglu et al., 2018; Alikhani et al., 2019). With our dataset, one can try reasoning about the food state changes from a raw ingredient to the final dish with the visual modality (Bosselut et al., 2018; Nishimura et al., 2021). One can also use our dataset for analyzing the cooking action effects throughout a recipe.

5.2 Procedural text generation

Generating procedural text from vision is an important task (Ushiku et al., 2017; Nishimura et al., 2019). To correctly reproduce procedures, the generated instructions should be consistent. The r-FG has the potential to make them more consistent as it represents the flow of the instructions. Since our

recipes are associated with cooking videos, one can use our dataset for that purpose.

6 Conclusion

We have presented a new multimodal dataset called Visual Recipe Flow. The dataset provides dense visual annotations for object states before and after a cooking action. The annotations allows us to learn each cooking action result. Experimental results demonstrated the effectiveness of our annotations for a multimodal information retrieval task. With our dataset, one can also try various applications, including multimodal commonsense reasoning and procedural text generation.

Acknowledgments

We would like to thank anonymous reviewers for their insightful comments. This work was supported by JSPS KAKENHI Grant Number 20H04210, 21H04910, 22H00540, 22K17983 and JST PRESTO Grant Number JPMJPR20C2.

References

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, et al. 2022. Flamingo: a visual language model for few-shot learning. *arXiv preprint arXiv:2204.14198*.

Malihe Alikhani, Sreyasi Nag Chowdhury, Gerard de Melo, and Matthew Stone. 2019. CITE: A corpus of image-text discourse relations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 570–575.

Vassileios Balntas, Edgar Riba, Daniel Ponsa, and Krystian Mikolajczyk. 2016. Learning local feature descriptors with triplets and shallow convolutional neural networks. In *Proceedings of the British Machine Vision Conference*, pages 119.1–119.11.

Antoine Bosselut, Omer Levy, Ari Holtzman, Corin Ennis, Dieter Fox, and Yejin Choi. 2018. Simulating action dynamics with neural process networks. In *Proceedings of the 6th International Conference on Learning Representations*.

Bhavana Dalvi, Lifu Huang, Niket Tandon, Wen-tau Yih, and Peter Clark. 2018. Tracking state changes in procedural text: a challenge dataset and models for process paragraph comprehension. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1595–1604.

Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural networks*, 18(5-6):602–610.

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

Phillip Isola, Joseph J. Lim, and Edward H. Adelson. 2015. Discovering states and transformations in image collections. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *Proceedings of the 7th International Conference on Learning Representations*.

Hirokuni Maeta, Tetsuro Sasada, and Shinsuke Mori. 2015. A framework for procedural text understanding. In *Proceedings of the 14th International Conference on Parsing Technologies*, pages 50–60.

Antoine Miech, Ivan Laptev, and Josef Sivic. 2018. Learning a text-video embedding from incomplete and heterogeneous data. *arXiv preprint arXiv:1804.02516*.

Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. 2019. Howto100m: Learning a text-video embedding by watching hundred million narrated video clips. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2630–2640.

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

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: Human Language Technologies*, pages 529–533.

Taichi Nishimura, Atsushi Hashimoto, and Shinsuke Mori. 2019. Procedural text generation from a photo sequence. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 409–414.

Taichi Nishimura, Atsushi Hashimoto, Yoshitaka Ushiku, Hirotaka Kameko, and Shinsuke Mori. 2021. *State-Aware Video Procedural Captioning*, page 1766–1774. Association for Computing Machinery, New York, NY, USA.

Taichi Nishimura, Suzushi Tomori, Hayato Hashimoto, Atsushi Hashimoto, Yoko Yamakata, Jun Harashima,

Yoshitaka Ushiku, and Shinsuke Mori. 2020. Visual grounding annotation of recipe flow graph. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4275–4284.

Liang-Ming Pan, Jingjing Chen, Jianlong Wu, Shaoteng Liu, Chong-Wah Ngo, Min-Yen Kan, Yugang Jiang, and Tat-Seng Chua. 2020. *Multi-Modal Cooking Workflow Construction for Food Recipes*, page 1132–1141. Association for Computing Machinery.

Dim P Papadopoulos, Enrique Mora, Nadiia Chepurko, Kuan Wei Huang, Ferda Ofli, and Antonio Torralba. 2022. Learning program representations for food images and cooking recipes. *arXiv preprint arXiv:2203.16071*.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. 2015. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. 2022. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*.

Niket Tandon, Keisuke Sakaguchi, Bhavana Dalvi, Dheeraj Rajagopal, Peter Clark, Michal Guerquin, Kyle Richardson, and Eduard Hovy. 2020. A dataset for tracking entities in open domain procedural text. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6408–6417.

Atsushi Ushiku, Hayato Hashimoto, Atsushi Hashimoto, and Shinsuke Mori. 2017. Procedural text generation from an execution video. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 326–335.

Semih Yagcioglu, Aykut Erdem, Erkut Erdem, and Nazli Ikizler-Cinbis. 2018. RecipeQA: A challenge dataset for multimodal comprehension of cooking recipes. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1358–1368.

Yoko Yamakata, Shinsuke Mori, and John A Carroll. 2020. English recipe flow graph corpus. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5187–5194.

Yixin Zhang, Yoko Yamakata, and Keishi Tajima. 2021. Mirecipe: A recipe dataset for stage-aware recognition of changes in appearance of ingredients. In *Proceedings of the 3rd ACM International Conference on Multimedia in Asia*, pages 1–7. Association for Computing Machinery.

Tag	Meaning	# tags
F	Food	5,098
T	Tool	758
D	Duration	129
Q	Quantity	1,778
Ac	Action by chef (cooking action)	2,532
Af	Action by food	353
Sf	State of food	971
St	State of tool	67
Total	—	11,686

Table 4: r-NE tags and the number of annotated tags of each type.

Label	Meaning	# labels
Agent	Action agent	330
Targ	Action target	2,961
Dest	Action destination	1,025
T-comp	Tool complement	157
F-comp	Food complement	20
F-eq	Food equality	2,397
F-part-of	Food part-of	330
F-set	Food set	987
T-eq	Tool equality	4
T-part-of	Tool part-of	0
A-eq	Action equality	1
V-tm	Head of clause for timing	112
other-mod	Other relationships	2,967
Total	—	11,291

Table 5: r-FG labels and the number of annotated labels of each type.

A Detailed statistics for the textual annotation

This section provides the detailed statistics for the annotated r-NE tags and r-FG labels.

A.1 r-NE tags

Table 4 shows the statistics for the annotated r-NE tags with the explanation of each tag. Among the tags, Ac, F, and T are specially important in our work. Ac denotes human cooking action, which is distinguished from action by food (Af). For example, in the instruction of "leave the salad to cool," "leave" is tagged with Ac, while "cool" is tagged with Af. F denotes foods including raw ingredients, intermediate products after cooking action, and the final dish. T denotes tools used for cooking. In our work, objects refer to the foods or tools. Our image annotation targeted the states of these objects.

A.2 r-FG labels

Table 5 shows the statistics for the annotated r-FG labels with the explanation of each tag. The cooking action (Ac) requires the objects (F or T). Targ describes this relationship taking the action and object as the starting and ending vertices, respectively. During the image annotation, we identified the required objects by using the flows labeled with Targ.

B Web interface

Our developed web interface is illustrated in Figure 4. In the first step (r-NE annotation), the annotator can annotate words in the ingredient and instruction lists with an r-NE tag by clicking the words and tag. In the second step (r-FG annotation), the annotator can annotate the r-NEs with a recipe flow by clicking starting and ending vertices and a label for them. In the final step (image annotation), the annotator can annotate the pre-action and post-action object states with images by clicking a frame and the button for the state. All objects for annotation are automatically prepared by tracing the recipe flows.

C A joint embedding model

In this section, we provide the detailed calculation of our model and experimental settings.

C.1 Model description

We first calculate a vector for an estimated post-action object state baesd on an action verb a, an object information o, and a pre-action image i_{pre}. The object is obtained by tracing a recipe flow labeled with Targ. a and o are converted to d_t-dimensional vectors h_a and h_o, respectively, by first embedding words into d_v-dimensional representations via a lookup table and then encoding them into d_t-dimensional vectors by using a bidirectional LSTM (BiLSTM) (Graves and Schmidhuber, 2005). For i_{pre}, we extract its feature $h_i^{pre} \in \mathbb{R}^{d_i}$ by using a pre-trained convolutional neural network (CNN) and transform it into $\hat{h}_i^{pre} \in \mathbb{R}^{d_t}$ as follows:

$$\hat{h}_i^{pre} = W_1^T h_i^{pre} + b_1^T, \quad (1)$$

where $W_1^T \in \mathbb{R}^{d_t \times d_i}$ and $b_1^T \in \mathbb{R}^{d_t}$ are learnable parameters. Given these fixed-size vectors, we then compute the vector for the estimated post-action object state $\hat{h}_o$ as:

$$\hat{h}_o = W_3^T(\text{ReLU}(W_2^T[h_a; h_o; h_i^{pre}] + b_2^T)) + b_3^T, \quad (2)$$

where ; denotes concatenation, and $W_2^T \in \mathbb{R}^{3d_t \times 3d_t}$, $W_3^T \in \mathbb{R}^{d_t \times 3d_t}$, $b_2^T \in \mathbb{R}^{3d_t}$, and $b_3^T \in \mathbb{R}^{d_t}$ are learnable parameters. $\hat{h}_o$ is then mapped to the joint embedding space as:

$$\begin{aligned} h_t &= (W_4^T \hat{h}_o + b_4^T) \circ \\ &\quad \sigma(W_5^T(W_4^T \hat{h}_o + b_4^T) + b_5^T), \quad (3) \\ \tilde{h}_t &= \frac{h_t}{||h_t||_2}, \quad (4) \end{aligned}$$

where $W_4^T \in \mathbb{R}^{d_e \times d_t}$, $W_5^T \in \mathbb{R}^{d_e \times d_e}$, $b_4^T, b_5^T \in \mathbb{R}^{d_e}$ are learnable parameters.

The post-action image i_{post} is fed to the pre-trained CNN to extract its feature $h_i^{post} \in \mathbb{R}^{d_i}$. Based on this feature, we compute $\hat{h}_i$ as:

$$\hat{h}_i = W_2^I(\text{ReLU}(W_1^I h_i^{post} + b_1^I)) + b_2^I, \quad (5)$$

where $W_1^I, W_2^I \in \mathbb{R}^{d_i \times d_i}$, and $b_1^I, b_2^I \in \mathbb{R}^{d_i}$ are learnable parameters. Following Miech et al. (2018), the feature vector $\hat{h}_i$ is then mapped to the joint embedding space as follows:

$$\begin{aligned} h_v &= (W_3^I \hat{h}_i + b_3^I) \circ \\ &\quad \sigma(W_4^I(W_3^I \hat{h}_i + b_3^I) + b_4^I), \quad (6) \\ \tilde{h}_v &= \frac{h_v}{||h_v||_2}, \quad (7) \end{aligned}$$

where σ is the sigmoid function, $\circ$ denotes the element-wise multiplication, $W_3^I \in \mathbb{R}^{d_e \times d_i}$, $W_4^I \in \mathbb{R}^{d_e \times d_e}$, and $b_3^I, b_4^I \in \mathbb{R}^{d_e}$ are learnable parameters.

Loss function. After mapping the inputs to the joint embedding space, we calculate the distance between these vectors as:

$$D(\tilde{h}_t, \tilde{h}_v) = ||\tilde{h}_t - \tilde{h}_v||_2. \quad (8)$$

Given n examples of $((\tilde{h}_{t,1}, \tilde{h}_{v,1}), \cdots, (\tilde{h}_{t,n}, \tilde{h}_{v,n}))$, we minimize the following triplet loss (Balntas et al., 2016):

$$\begin{aligned} \mathcal{L} = \sum_{i=1}^{n} &\{\max(D_{i,i} - D_{i,j} + \delta, 0) \\ &+ \max(D_{i,i} - D_{k,i} + \delta, 0)\}, \quad (9) \end{aligned}$$

where $D_{i,j} = D(\tilde{h}_{t,i}, \tilde{h}_{v,j})$, and δ denotes a margin. In Equation (9), $D_{i,i}$ is the distance for a positive pair, and $D_{i,j}$ and $D_{k,i}$ are the distances for pairs with negative text and image feature vectors, respectively. For negative sampling, we simply sample negative examples from a mini-batch.

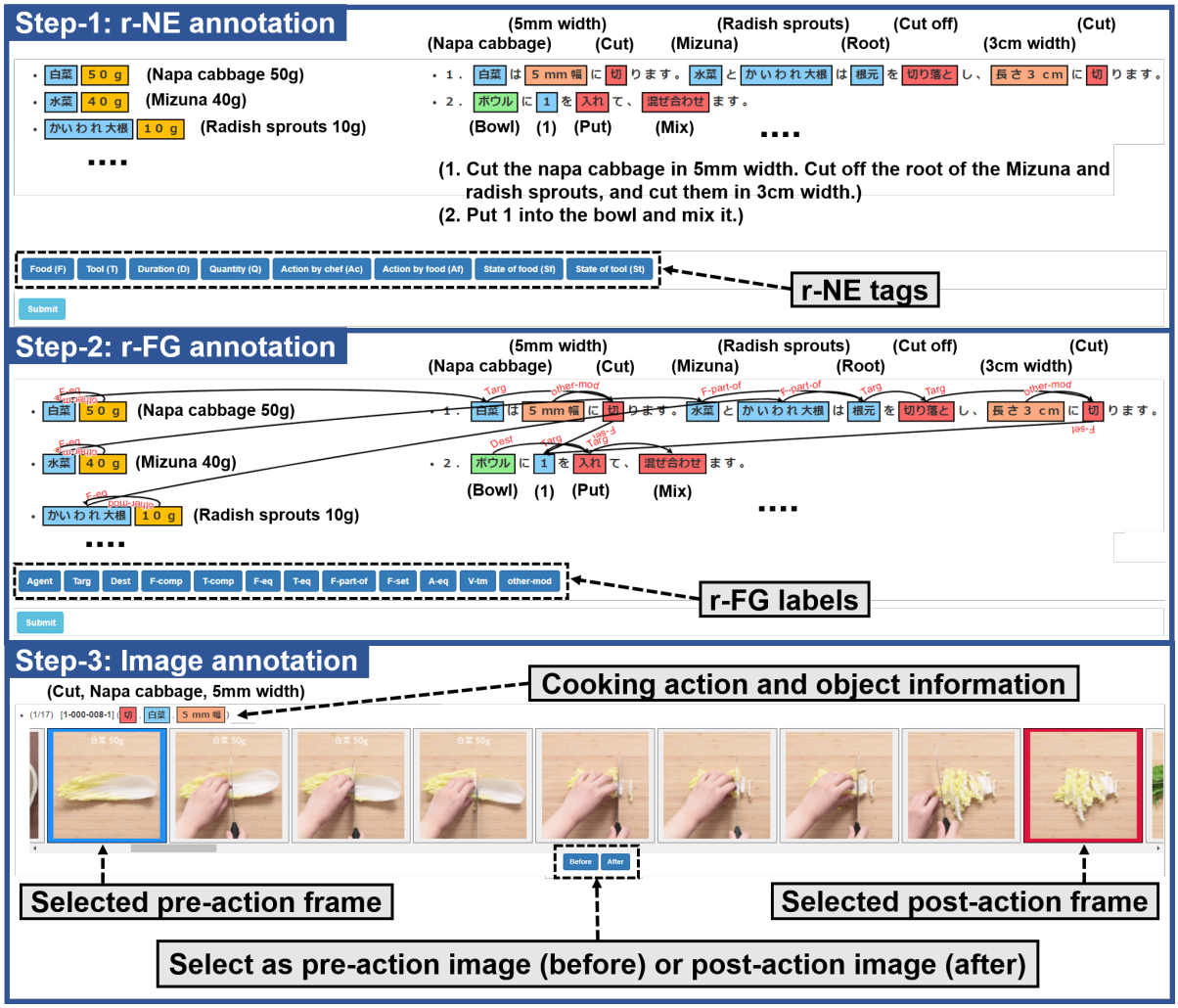

Figure 4: Our web annotation interface. The annotator can complete annotations only by mouse operations. The web page is written in Japanese.

C.2 Settings

Model parameters. We used a 1-layer 256-dimensional BiLSTM to encode words. We set the dimensions as $(d_v, d_t, d_i, d_e) = (496, 512, 2048, 128)$. We used ResNet-152 (He et al., 2016), which was pre-trained on ImageNet (Russakovsky et al., 2015), to extract a feature vector of 2048 dimensions from an image.

Optimization. We used AdamW (Loshchilov and Hutter, 2019) with an initial learning rate of 1.0×10^{-5} to tune the parameters. During training, we froze only the parameters of the CNN. Each model was trained for 350 epochs, and we created a mini-batch with 4 recipes at each step. We set δ in Equation (9) to 0.1. We evaluated the model performance through 10-fold cross-validation by splitting the dataset into 90% for training and 10% for testing.