Towards Flow Graph Prediction of Open-Domain Procedural Texts

Keisuke Shirai[1] **Hirotaka Kameko**[2] **Shinsuke Mori**[2]
[1]Graduate School of Informatics, Kyoto University
[2]Academic Center for Computing and Media Studies, Kyoto University
shirai.keisuke.64x@st.kyoto-u.ac.jp {kameko,forest}@i.kyoto-u.ac.jp

Abstract

Machine comprehension of procedural texts is essential for reasoning about the steps and automating the procedures. However, this requires identifying entities within a text and resolving the relationships between the entities. Previous work focused on the cooking domain and proposed a framework to convert a recipe text into a flow graph (FG) representation. In this work, we propose a framework based on the recipe FG for flow graph prediction of open-domain procedural texts. To investigate flow graph prediction performance in non-cooking domains, we introduce the wikiHow-FG corpus from articles on wikiHow, a website of how-to instruction articles. In experiments, we consider using the existing recipe corpus and performing domain adaptation from the cooking to the target domain. Experimental results show that the domain adaptation models achieve higher performance than those trained only on the cooking or target domain data.

1 Introduction

A procedural text guides a human to complete daily activities like cooking and furniture assembly. Machine comprehension of these texts is essential for reasoning about the steps (Zhang et al., 2020b) and automating the procedures (Bollini et al., 2013). However, it needs to identify entities within a text and resolve relationships between the entities. Converting the text into an actionable representation (e.g., flow graph (Momouchi, 1980)) is an approach for solving these problems.

There are several works on converting a procedural text into an action graph (Mori et al., 2014; Kulkarni et al., 2018; Kuniyoshi et al., 2020). In the cooking domain, various approaches (Mori et al., 2014; Kiddon et al., 2015; Pan et al., 2020; Papadopoulos et al., 2022) have been taken because there are a rich amount of available resources on the web. Among them, recipe flow graph (FG) (Mori et al., 2014) has the advantage of capturing fine-grained relationships at entity-level. While the original work (Mori et al., 2014) introduced a framework and corpus in Japanese, the recent work (Yamakata et al., 2020) proposed those in English. The current FG framework has two issues. Since FG is designed to represent the flow of actions in a procedural text, it should be applicable to other procedural domains such as crafting. One is that the framework has only been applied to the cooking domain. The other is that preparing a large number of annotations (e.g., thousands of articles) is unrealistic due to its complex annotation procedures.

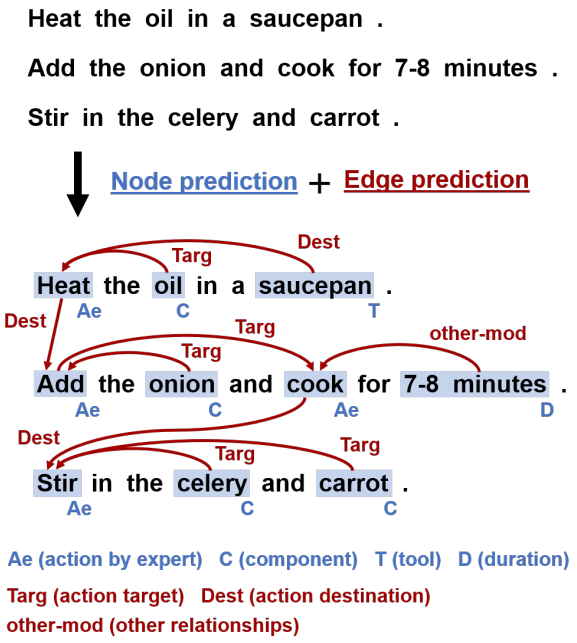

Figure 1: Example of flow graph prediction with our framework. The prediction is in two stages: node prediction (colored in blue) and edge prediction (colored in red). In this work, we use this framework to predict flow graphs of open-domain procedural texts.

In this work, we propose a framework based on the English recipe flow graph (English-FG) (Yamakata et al., 2020) for FG prediction of open-

Proceedings of the 8th Workshop on Representation Learning for NLP (RepL4NLP 2023), pages 87–96

Tag	Meaning
C (F)	Component (Food)
T	Tool
D	Duration
Q	Quantity
Ae (Ac)	Action by expert (chef)
Ae2 (Ac2)	Discontinuous Ae (Ac)
Ac (Af)	Action by component (food)
At	Action by tool
Sc (Sf)	State of component (food)
St	State of tool

Table 1: Tags and their meanings. The inside of the parenthesis represents a tag and its meaning in English-FG.

Label	Meaning
Agent	Action agent
Targ	Action target
Dest	Action destination
T-comp	Tool complement
C-comp (F-comp)	Component (Food) complement
C-eq (F-eq)	Component (Food) equality
C-part-of (F-part-of)	Component (Food) part-of
C-set (F-set)	Component (Food) set
T-eq	Tool equality
T-part-of	Tool part-of
A-eq	Action equality
V-tm	Head of clause for timing
other-mod	Other relationships

Table 2: Labels and their meanings. The inside of the parenthesis represents a label and its meaning in English-FG.

domain procedural texts. We show the overview of our framework in Figure 1. Our motivation is to expand the scope of the recipe FG to non-cooking domains by treating food ingredients as final product components. Our framework predicts an FG in two stages: node prediction and edge prediction, following Maeta et al. (2015). Our framework is compatible with the English-FG, and we can jointly learn representations from data in the cooking and non-cooking domains. To investigate FG prediction performance in non-cooking domains, we introduce the wikiHow-FG corpus from wikiHow articles. The corpus was constructed by selecting four domains from the wikiHow categories and annotating 30 articles for each domain.

In experiments, we assume a low-resource scenario in which we can access only a few training examples in the target domain. This is a realistic scenario considering the huge annotation cost for FG. To tackle this issue, we consider domain adaptation from the existing cooking domain to the target domain. Experimental results show that domain adaptation models obtain higher performance than those trained only on the cooking or target domain data. We also considered two data augmentation techniques to boost performance. From the results, we found that they improve performance in particular domains.

Our contributions are three-fold:

- We propose a framework based on the English-FG for flow graph prediction of open-domain procedural texts.
- We introduce the wikiHow-FG corpus, a new corpus from wikiHow articles. This corpus is based on four wikiHow domains and has 30 annotated articles for each domain.
- We assume a low-resource scenario in the target domain and consider domain adaptation from the cooking to the target domain. Experimental results show that domain adaptation models outperform those trained only on the cooking or target domain data.

2 Recipe flow graph

In this section, we provide a brief description of the recipe flow graph (FG) (Mori et al., 2014). A recipe FG is a directed acyclic graph $G(V, E)$, where V represents entities as nodes, while E represents the relationships between the nodes as labeled edges. Currently, FG annotations are available in Japanese (Mori et al., 2014) and English (Yamakata et al., 2020), and these corpora provide annotations of hundreds of recipes. Note that Japanese and English frameworks for FG are not compatible since the English FG uses additional tags to handle English-specific expressions. As we focus on texts in English, we consider the English-FG framework (Yamakata et al., 2020) in the following sections.

2.1 Flow graph representation

A recipe FG representation is divided into two types of annotations; node and edge annotations. Nodes represent entities with tags in the IOB-format (Ramshaw and Marcus, 1995). As listed in Table 1, 10 types of tags are used in the English-FG. Labeled edges represent the relationships between

the nodes. As listed in Table 2, 13 types of labels are used in the English-FG.

2.2 Flow graph prediction

For the automatic prediction of the FG, previous work (Maeta et al., 2015) proposed to divide the problem into two subtasks: node prediction and edge prediction. In both subtasks, models are trained in a supervised fashion.

Node prediction identifies nodes in an article with the tags. Maeta et al. (2015); Yamakata et al. (2020) formulated this problem as a sequence labeling problem and used NER model (Lample et al., 2016). While predicting tags at sentence-level is common in NER (Lample et al., 2016), previous work (Yamakata et al., 2020) used an entire recipe text as input.[1]

Edge prediction constructs a directed acyclic graph by predicting labeled edges between the nodes. This is formulated as a problem of finding the maximum spanning tree as:

$$\hat{G} = \underset{G \in \mathcal{G}}{\operatorname{argmax}} \sum_{(u,v,l)} s(u, v, l), \qquad (1)$$

where $s(u, v, l)$ represents the score of a labeled edge from u to v with label l. We can solve this problem by using the Chu-Liu-Edmonds algorithm. The scores are calculated using a graph-based dependency parser (McDonald et al., 2005).

3 Flow graph prediction of open-domain procedural texts

Our framework is based on the English-FG and applies to non-cooking domains by treating foods in recipe texts as final product components. Examples of the components include tomato and beef for cooking, cardboard and glue for crafting, and gear and tire for vehicle maintenance. The framework uses tags and labels defined in Table 1 and Table 2, respectively. These tags and labels are slightly modified from the definitions in the English-FG to avoid confusion, and we did not add or delete any tags and labels. Therefore, our framework is compatible with the English-FG, and we can learn representations jointly from the cooking and non-cooking domains.

[1] In our preliminary experiments, we found that predicting the tags at document-level improves accuracy by 10% compared to the prediction at sentence-level.

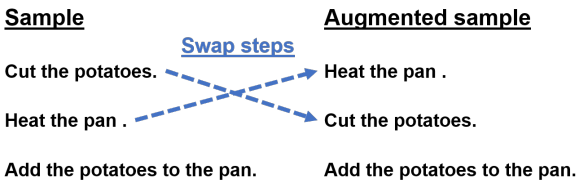

Figure 2: Example of swapping steps. The first and second steps are swappable without violating causality.

With this framework, we consider predicting flow graphs of open-domain procedural texts. The prediction is performed in two stages: node prediction and edge prediction as in Section 2. Models are trained in a supervised way as the previous approach (Maeta et al., 2015). However, preparing a large number of examples in a new domain is unrealistic, considering the huge FG annotation cost. Thus, we assume that only a few training examples are available in the target domain. To tackle this issue, we consider domain adaptation from the existing cooking domain data to the target domain data. In the rest of this section, we formulate the task in Section 3.1, then consider data augmentation techniques that fit our setting in Section 3.2.

3.1 Task definition

We are given N examples in the cooking domain $(V_1^C, E_1^C), \cdots, (V_N^C, E_N^C)$ and M examples in the target domain $(V_1^T, E_1^T), \cdots, (V_M^T, E_M^T)$, where V^C and E^C are a set of vertices and edges in the cooking domain, while V^T and E^T are those in the target domain. Our goal is to maximize the performance of node and edge prediction models in the target domain. In this work, we use the English-FG corpus (300 articles) as the cooking domain examples and the wikiHow-FG corpus (Section 4) as the target domain examples. Note that M is a minimal number (namely, $M = 5$) in our setting, and the task has an aspect of low-resource domain adaptation (Xu et al., 2021). Note also that training with only the cooking or target-domain examples becomes zero-shot or few-shot learning scenarios, respectively.

3.2 Data augmentation

For improving performance in a low-resource setting, data augmentation is one of possible solutions (Fadaee et al., 2017; Ding et al., 2020). In this work, we consider the following two data augmentation techniques: step swapping and word replacement.

Domain	Task examples
Food and Entertaining	*Cooking acorn squash*, *Making lavender tea*, *Baking a cherry pie*
Hobbies and Crafts	*Making a bar soap*, *Making a duct tape bow*, *Making a paper box*
Home and Garden	*Cleaning a mattress pad*, *Installing a microwave*, *Making a scented candle*
Cars & Other Vehicles	*Fixing a slipped bike chain*, *Cleaning car window*, *Cleaning tail lights*

Table 3: Examples of article titles for each domain.

Domain	# Characters	# Words	# Steps	# Tags	# Labels
Food and Entertaining	10,167	2,761	224	1,132	1,124
Hobbies and Crafts	9,407	2,556	247	1,062	1,076
Home and Garden	7,700	2,010	205	894	886
Cars & Other Vehicles	6,432	1,622	173	625	622

Table 4: Statistics of the wikiHow-FG corpus.

Step swapping augments an example by replacing two arbitrary steps in an article as illustrated in Figure 2. However, randomly choosing and swapping two steps might break their causal relationship. For example, we cannot swap two steps "Cut the potatoes." and "Add the potatoes to the pan." in Figure 2. In this work, we augment examples keeping this constraint by using flow graph annotations.

Word replacement augments an example by replacing a word with an arbitrary one. For each word in a step, we replace a word with one of its synonyms from WordNet (Dai and Adel, 2020) with a probability p.

4 wikiHow-FG corpus

The wikiHow-FG corpus is a new flow graph corpus from articles on wikiHow[2], a website with more than 110K how-to articles. wikiHow articles have been used as a language resource for procedural texts (Zhou et al., 2019; Zellers et al., 2019; Zhang et al., 2020b,a; Zhou et al., 2022). In the following, we describe the data collection, annotation procedure, and statistics of the annotation results.

4.1 Data collection

For the target domains, we selected four categories from wikiHow: (i) *Food and Entertaining*, (ii) *Hobbies and Crafts*, (iii) *Home and Garden*, and (iv) *Cars & Other Vehicles*. We decided on those domains because many articles target and interact with substantial objects. We show examples of article titles in Table 3. *Food and Entertaining* is a domain close to the cooking domain as with the English-FG. *Hobbies and Crafts* is far from the cooking domain in the sense that they assemble non-edible materials (e.g., *making a bar soap*). The remaining two domains are further from these domains because they contain non-assembly tasks (e.g., *cleaning a table* and *fixing a broken chain*).

We collected 30 articles for each domain from the wikiHow corpus (Zhang et al., 2020b). To exclude low-quality articles, we collected articles with 25 or more words and more than 50% user ratings. We then manually excluded articles with too abstract goals or not targeting substantial objects. We used article headlines as steps for annotation and experiments. Note that each article has a title and visual information (e.g., images or videos) describing the procedures. Exploiting them is interesting, but we leave that direction for future work.

4.2 Annotation procedure

Due to the dense, complex nature of the flow graph, constructing a high-quality corpus is a challenging problem. In order to guarantee the annotation quality, we first trained an annotator with 10 recipes sampled from the English-FG corpus (Yamakata et al., 2020). The training continued until the inter-annotator agreements with the ground-truth annotations reached over 80%. We then asked the annotator to annotate wikiHow articles. For both the node and edge annotations, we used the flow graph annotation tool (Shirai et al., 2022).[3] The whole annotation took 40 hours.

4.3 Statistics

We show statistics of the wikiHow corpus in Table 4. The articles comprise 280.9 characters,

[2] https://www.wikihow.com

[3] Before the annotation, we tokenized steps into words with the stanza toolkit (Qi et al., 2020).

Annotation type	Agreement
Node annotation	90.17%
Edge annotation	57.43%

Table 5: Inter-annotator agreements.

74.6 words, and 7.08 steps on average. The average number of tags and labels per article is 37.73 ($\pm$16.48) and 37.47 ($\pm$17.21) on *Food and Entertaining*, 35.40 ($\pm$9.08) and 35.87 ($\pm$9.44) on *Hobbies and Crafts*, 29.80 ($\pm$7.83) and 29.53 ($\pm$8.29) on *Home and Garden*, and 20.83 ($\pm$6.40) and 20.73 ($\pm$7.14) on *Cars & Other Vehicles*. Since articles on *Home and Garden* and *Cars & Other Vehicles* have fewer words than the other two domains, the number of tags and labels also becomes smaller.

4.4 Inter-annotator agreements

To assess the consistency of the annotations, we asked another annotator to re-annotate 10% of articles for each domain. We then measured F1 scores between the two sets of annotations using the original annotations as the ground-truth ones. Table 5 shows the results. The agreement for the node annotation (90.14%) was high, considering entities corresponding to the tags greatly change depending on the domain. For the edge annotation, the agreement (57.43%) drops from the one for node annotation. However, this agreement is also high considering errors from the node annotation influence this step and that a large number of candidate edges for the annotation.

5 Node prediction

5.1 Experimental settings

Model. We adopted a BiLSTM-CRF model (Lample et al., 2016) for node prediction by replacing a BiLSTM encoder with a pre-trained language model (LM) (Devlin et al., 2019; Liu et al., 2019; He et al., 2021). While previous work (Yamakata et al., 2020) used a pre-trained BERT (Devlin et al., 2019) as the encoder, we used a pre-trained DeBERTa (He et al., 2021).[4] This model has 140M parameters in total.

Training. We trained a domain adaptation model by first training on the cooking domain data (English-FG) and then training again on the target domain data (wikiHow-FG). Note that we use only target domain examples of the wikiHow-FG corpus (e.g., when targeting *Hobbies and Crafts*, we do not use examples in the other three domains.). We also train models only on the cooking or target domain data and report the results to compare with the domain adaptation results.

For an optimization method, we used AdamW (Loshchilov and Hutter, 2019) with an initial learning rate of 5.0×10^{-5} and a weight decay of 1.0×10^{-5}. We tuned a learning rate with a cosine-annealing (Loshchilov and Hutter, 2019) (S_d steps) with a linear warm-up (S_w steps) at every iteration. We created a mini-batch from B articles. We set $(B, S_\mathrm{w}, S_\mathrm{d}) = (5, 500, 4500)$ and $(B, S_\mathrm{w}, S_\mathrm{d}) = (3, 100, 900)$ for training on the English-FG corpus and the wikiHow-FG corpus, respectively. We tuned these hyper-parameters on the development set. We used the data augmentation techniques only for the target domain data. For the step swapping, we created 5 augmented examples at maximum from one example. For the word replacement, we set 0.5 to p and created 10 examples from one example.

Evaluation. We split the English-FG corpus into 80% for training, 10% for validation, and the rest of 10% for testing. For the wikiHow-FG corpus, we split 30 articles of each domain into 6 folds. For more reliable results, we performed 6-fold cross-validation by using 1 fold for training, 1 fold for validation, and the remaining 4 folds for testing. We used precision, recall, and F1 for evaluation metrics, following Yamakata et al. (2020). On the evaluation, we report the average scores of the models on the test set.

Model configurations. We refer to the domain adaptation models as **domain-adaptation** models. We also refer to the models trained only on the English-FG or the wikiHow-FG corpus as **cooking-only** and **target-only** models, respectively.

5.2 Results

The results are shown in Table 6. We see that the **target-only** models achieve an F1 score of 66.9% or more in all the target domains. This implies that the node prediction model can predict nodes to an extent with a few annotated articles. We also see that the **cooking-only** models achieve competitive performance with the **target-only** ones and outperform them in the two domains of *Food and*

[4] In our preliminary experiments, we confirmed that DeBERTa improves the accuracy of BERT by 0.47% on the English-FG corpus.

Domain	Model	Augmentation		Prec.	Recall	F1
		Step-swap	Word-replace			
Food and Entertaining	Target-only			0.770	0.784	0.777
	Cooking-only			0.884	0.877	0.880
	Domain-adaptation			0.890	0.892	0.891
	Domain-adaptation	✓		**0.894**	**0.895**	**0.895**
	Domain-adaptation		✓	0.885	0.891	0.888
Hobbies and Crafts	Target-only			0.698	0.707	0.702
	Cooking-only			0.703	0.684	0.693
	Domain-adaptation			**0.794**	**0.805**	**0.799**
	Domain-adaptation	✓		0.784	0.795	0.789
	Domain-adaptation		✓	0.781	0.790	0.785
Home and Garden	Target-only			0.663	0.676	0.669
	Cooking-only			0.734	0.742	0.738
	Domain-adaptation			0.780	0.786	0.783
	Domain-adaptation	✓		**0.787**	**0.791**	**0.786**
	Domain-adaptation		✓	0.765	0.773	0.769
Cars & Other Vehicles	Target-only			0.650	0.690	0.669
	Cooking-only			0.646	0.695	0.670
	Domain-adaptation			**0.748**	**0.784**	**0.765**
	Domain-adaptation	✓		0.734	**0.784**	0.761
	Domain-adaptation		✓	0.729	0.772	0.750

Table 6: Results of the node prediction experiments. The check mark symbol (✓) indicates the used training data (in cooking and target domains) and augmentation techniques (step-swap and word-replace).

Domain	Ae			C			T		
	Duplicates	F1		Duplicates	F1		Duplicates	F1	
		Src	Adpt		Src	Adpt		Src	Adpt
Food and Entertaining	92.06%	0.941	0.952	72.11%	0.932	0.933	77.94%	0.896	0.882
Hobbies and Crafts	69.03%	0.943	0.951	10.33%	0.717	0.833	51.79%	0.398	0.588
Home and Garden	65.19%	0.954	0.961	18.40%	0.716	0.795	43.55%	0.567	0.678
Cars & Other Vehicles	46.04%	0.905	0.919	6.88%	0.666	0.805	27.47%	0.459	0.557

Table 7: F1 scores of Ae, C, and T tags with the percentage of entities that appeared in the English-FG corpus and also in the wikiHow-FG corpus. Src and Adpt denote **cooking-only** and **domain-adaptation** models, respectively.

Entertaining and *Home and Garden*. Particularly in *Food and Entertaining*, the **cooking-only** model surpasses the **target-only** one by 10.3% in F1. We consider that this domain is close to the cooking domain of the English-FG; thus, the **cooking-only** model is more advantageous as it can access more examples.

Next, the **domain-adaptation** models achieve the best performance in all the domains compared with the **cooking-only** and **target-only** models (76.5% or more F1). The most significant improvements are obtained in the two domains of *Hobbies and Crafts* and *Cars & Other Vehicles* (9.5% and 10.5% improvements in F1). These results indicate that domain adaptation from the cooking to the target domain is effective for training the node prediction model.

Third, we see that using the augmented data by the step swapping slightly improves performance from the **domain-adaptation** models in *Food and Entertaining* and *Home and Garden*. On the other hand, the word replacement does not contribute to any improvement. One possible reason is that a replaced word does not necessarily match the corresponding tag, and this disrupts the improvement.

5.3 Tag-level prediction performance

Entities for each tag can greatly change depending on the domain. In that case, the degree of improvement from the **cooking-only** to the **domain-adaptation** model is expected to increase as the duplicate entities between the domains decrease.

Domain	Model	Augmentation		Prec.	Recall	F1
		Step-Swap	Word-Replace			
Food and Entertaining	Target-only			0.335	0.338	0.337
	Cooking-only			0.725	0.731	0.728
	Domain-adaptation			0.750	**0.756**	**0.753**
	Domain-adaptation	✓		0.747	0.752	0.750
	Domain-adaptation		✓	**0.761**	0.752	0.749
Hobbies and Crafts	Target-only			0.285	0.281	0.283
	Cooking-only			0.613	0.605	0.609
	Domain-adaptation			0.649	0.640	0.644
	Domain-adaptation	✓		0.646	0.638	0.642
	Domain-adaptation		✓	**0.653**	**0.644**	**0.648**
Home and Garden	Target-only			0.229	0.232	0.231
	Cooking-only			0.644	0.649	0.646
	Domain-adaptation			0.659	0.665	0.662
	Domain-adaptation	✓		0.656	0.662	0.659
	Domain-adaptation		✓	**0.674**	**0.680**	**0.677**
Cars & Other Vehicles	Target-only			0.154	0.155	0.154
	Cooking-only			0.587	0.590	0.587
	Domain-adaptation			0.607	0.610	0.609
	Domain-adaptation	✓		0.607	0.610	0.608
	Domain-adaptation		✓	**0.617**	**0.620**	**0.618**

Table 8: Results of the edge prediction experiments. The check mark symbol (✓) indicates the used training data (in cooking and target domains) and augmentation techniques (step-swap and word-replace).

To investigate this assumption, we measured tag-level prediction performance in F1 with the duplicate ratio of entities of the wikiHow-FG in the English-FG. We targeted the three tags of Ae, C, and T because these tags frequently appear in all the domains.

The results are shown in Table 7. For Ae, the degree of improvement from the **cooking-only** to the **domain-adaptation** model is small regardless of the duplicate ratios, which is contrary to our assumption. These results imply that recognizing entities for Ae is easy irrespective of the domain. For C and T, the **domain-adaptation** models significantly outperform the **cooking-only** ones in the three domains other than *Food and Entertaining*. These results imply that the domain adaptation is effective for recognizing C and T tags when the domain is further from the cooking domain.

6 Edge prediction

6.1 Experimental settings

Model. We adopted a biaffine dependency parser (Dozat and Manning, 2018) for edge prediction.[5] This model uses separate modules for edge prediction and label prediction. The resulting loss l is defined as a weighted sum of losses from the two modules:

$$l = \lambda l^{(\text{edge})} + (1 - \lambda) l^{(\text{label})}, \quad (2)$$

where λ controls the strength of the two losses. We empirically set 0.5 to λ. We used a pre-trained DeBERTa (He et al., 2021) to obtain contextualized word representations. This model has 149M parameters in total.

Training. Similarly to Section 5.1, we trained a domain adaptation model for edge prediction first on the English-FG corpus and then on the wikiHow-FG corpus. For an optimization method, we used AdamW (Loshchilov and Hutter, 2019) with a combination of a cosine-annealing and linear warm-up learning rate scheduling method. We used the same hyperparameters in Section 5.1.

Evaluation. We used the same splits of the English-FG and wikiHow-FG corpora as in Section 5 and performed 6-fold cross-validation for

[5] Previous work (Maeta et al., 2015) used a linear model, but we confirmed that our model achieves higher performance on the English-FG corpus.

Domain	F1
Food and Entertaining	0.679 (-9.8%)
Hobbies and Crafts	0.501 (-22.2%)
Home and Garden	0.494 (-25.4%)
Cars & Other Vehicles	0.449 (-26.3%)

Table 9: Results of the pipeline experiments. The inside of the parenthesis represents the performance drop from the **domain-adaptation** model with ground-truth tags.

more reliable results. We report the average scores of the models on the test set. For evaluation metrics, we used precision, recall, and F1 between predicted and ground-truth labeled edges of (u, v, l).

Model configurations. We used the same model notations of **cooking-only**, **target-only**, and **domain-adaptation** models as in Section 5.

6.2 Results

The results are shown in Table 8. We used ground-truth tags to identify nodes. Contrary to the node prediction results, the **target-only** models achieve poor performance in all the domains (33.8% or less in all the metrics). On the other hand, the scores of the **cooking-only** models are more than twice those of the **target-only** models. These results show that the edge prediction model requires more training examples than the node prediction one. These also show that with the English-FG corpus, predicting edges with 58.7% or more F1 is possible in non-cooking domains.

Next, the **domain-adaptation** models outperform the **target-only** and **cooking-only** ones in all the domains. This is consistent with the results in the node prediction task. These results mean that the domain adaptation from the cooking to the target domain is also effective for the edge prediction model. For the results with the data augmentation techniques, the step swapping does not contribute to any improvement, contrary to Section 5.2. The word replacement improves the performance of the **domain-adaptation** models in the three domains other than *Food and Entertaining*.

6.3 Pipeline experiments

So far, the model has used ground-truth tags to identify nodes. However, in a realistic scenario, the model must predict labeled edges with the predicted nodes. In this scenario, errors in the node prediction step would affect performance in the edge prediction step. To investigate edge prediction performance in this setting, we conducted experiments of edge prediction with the predicted nodes. We predicted nodes using the models in Section 5.2. In order to evaluate the model with tag information, we measured F1 of tuples of (u, v, l, n_u, n_v) between ground-truth and predicted ones, where n_u and n_v are the tags of the starting and ending nodes, respectively.

Table 9 shows the results with performance drops from those in Table 8. We see that 9.8% drops in *Food and Entertaining*, and more significant drops occur in the other three domains (about 24.6%). In these three domains, F1 scores of the node prediction are about 10% smaller than that of *Food and Entertaining*, and this gap would cause such large performance drops. We consider that improving node prediction performance would alleviate these drops.

7 Related work

Mori et al. (2014) designed a flow graph (FG) representation in the cooking domain and introduced a corpus of recipe texts. Subsequent works introduced a corpus in English (Yamakata et al., 2020) and corpora with visual annotations (Nishimura et al., 2020; Shirai et al., 2022). Maeta et al. (2015) proposed a method for an automatic FG prediction. Our work stems from this line of research and is the first attempt to apply the framework to non-cooking domains. Ours is also the first work to use a neural network-based method for the edge prediction.

Other than the recipe FG, there are several works that focus on obtaining an actionable representation from a procedural text. In cooking, Kiddon et al. (2015) proposed an unsupervised EM algorithm, while Pan et al. (2020); Papadopoulos et al. (2022) proposed supervised approaches. In biochemistry, Kulkarni et al. (2018); Tamari et al. (2021) introduced datasets for mapping wet lab protocols to an action graph. In material science, Kuniyoshi et al. (2020) represented the synthesis process with flow graphs. The works (Pan et al., 2020; Papadopoulos et al., 2022; Tamari et al., 2021; Kuniyoshi et al., 2020) are especially close to ours in the sense that they aim to obtain a document-level action graph in a supervised way.

8 Conclusion

We proposed a framework based on the English-FG and investigated flow graph prediction performance in non-cooking domains. We presented the

wikiHow-FG corpus from wikiHow articles. We considered domain adaptation from the cooking to the target domain. Experimental results show that domain adaptation models outperform those trained only on the cooking or target domain data. In future work, we consider applying this framework to other domains, such as material science and biochemistry. One can also try improving performance using more sophisticated data augmentation techniques. We hope that our work will provide new insights into procedural text understanding.

Acknowledgments

We would like to thank anonymous reviewers for their insightful comments. This work was supported by JSPS KAKENHI Grant Number 20H04210 and 21H04910.

References

Mario Bollini, Stefanie Tellex, Tyler Thompson, Nicholas Roy, and Daniela Rus. 2013. Interpreting and executing recipes with a cooking robot. In *Experimental Robotics*, pages 481–495. Springer.

Xiang Dai and Heike Adel. 2020. An analysis of simple data augmentation for named entity recognition. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3861–3867. International Committee on Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. 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 4171–4186. Association for Computational Linguistics.

Bosheng Ding, Linlin Liu, Lidong Bing, Canasai Kruengkrai, Thien Hai Nguyen, Shafiq Joty, Luo Si, and Chunyan Miao. 2020. DAGA: Data augmentation with a generation approach for low-resource tagging tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6045–6057. Association for Computational Linguistics.

Timothy Dozat and Christopher D. Manning. 2018. Simpler but more accurate semantic dependency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 484–490. Association for Computational Linguistics.

Marzieh Fadaee, Arianna Bisazza, and Christof Monz. 2017. Data augmentation for low-resource neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 567–573. Association for Computational Linguistics.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: decoding-enhanced bert with disentangled attention. In *9th International Conference on Learning Representations*.

Chloé Kiddon, Ganesa Thandavam Ponnuraj, Luke Zettlemoyer, and Yejin Choi. 2015. Mise en place: Unsupervised interpretation of instructional recipes. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 982–992. Association for Computational Linguistics.

Chaitanya Kulkarni, Wei Xu, Alan Ritter, and Raghu Machiraju. 2018. An annotated corpus for machine reading of instructions in wet lab protocols. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 97–106. Association for Computational Linguistics.

Fusataka Kuniyoshi, Kohei Makino, Jun Ozawa, and Makoto Miwa. 2020. Annotating and extracting synthesis process of all-solid-state batteries from scientific literature. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 1941–1950. European Language Resources Association.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

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. Association for Computational Linguistics.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language*

Processing, pages 523–530. Association for Computational Linguistics.

Yoshio Momouchi. 1980. Control structures for actions in procedural texts and PT-chart. In *COLING 1980 Volume 1: The 8th International Conference on Computational Linguistics*.

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.

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. European Language Resources Association.

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. In *Proceedings of the 28th ACM International Conference on Multimedia*, 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. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16559–16569.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 101–108. Association for Computational Linguistics.

Lance Ramshaw and Mitch Marcus. 1995. Text chunking using transformation-based learning. In *Third Workshop on Very Large Corpora*.

Keisuke Shirai, Atsushi Hashimoto, Taichi Nishimura, Hirotaka Kameko, Shuhei Kurita, Yoshitaka Ushiku, and Shinsuke Mori. 2022. Visual recipe flow: A dataset for learning visual state changes of objects with recipe flows. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 3570–3577. International Committee on Computational Linguistics.

Ronen Tamari, Fan Bai, Alan Ritter, and Gabriel Stanovsky. 2021. Process-level representation of scientific protocols with interactive annotation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2190–2202. Association for Computational Linguistics.

Haoran Xu, Seth Ebner, Mahsa Yarmohammadi, Aaron Steven White, Benjamin Van Durme, and Kenton Murray. 2021. Gradual fine-tuning for low-resource domain adaptation. In *Proceedings of the Second Workshop on Domain Adaptation for NLP*, pages 214–221. Association for Computational Linguistics.

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. European Language Resources Association.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800. Association for Computational Linguistics.

Li Zhang, Qing Lyu, and Chris Callison-Burch. 2020a. Intent detection with WikiHow. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 328–333. Association for Computational Linguistics.

Li Zhang, Qing Lyu, and Chris Callison-Burch. 2020b. Reasoning about goals, steps, and temporal ordering with WikiHow. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 4630–4639. Association for Computational Linguistics.

Shuyan Zhou, Li Zhang, Yue Yang, Qing Lyu, Pengcheng Yin, Chris Callison-Burch, and Graham Neubig. 2022. Show me more details: Discovering hierarchies of procedures from semi-structured web data. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2998–3012. Association for Computational Linguistics.

Yilun Zhou, Julie Shah, and Steven Schockaert. 2019. Learning household task knowledge from WikiHow descriptions. In *Proceedings of the 5th Workshop on Semantic Deep Learning (SemDeep-5)*, pages 50–56. Association for Computational Linguistics.