Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020), pages 5187–5194
Marseille, 11–16 May 2020

English Recipe Flow Graph Corpus

Yoko Yamakata*, Shinsuke Mori†, John Carroll‡

*The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo, Japan
yamakata@mi.u-tokyo.ac.jp

†Kyoto University, Yoshida-Honmachi, Sakyo-ku, Kyoto, Japan
forest@i.kyoto-u.ac.jp

‡University of Sussex, Falmer, Brighton, UK
J.A.Carroll@sussex.ac.uk

Abstract

We present an annotated corpus of English cooking recipe procedures, and describe and evaluate computational methods for learning these annotations. The corpus consists of 300 recipes written by members of the public, which we have annotated with domain-specific linguistic and semantic structure. Each recipe is annotated with (1) 'recipe named entities' (r-NEs) specific to the recipe domain, and (2) a flow graph representing in detail the sequencing of steps, and interactions between cooking tools, food ingredients and the products of intermediate steps. For these two kinds of annotations, inter-annotator agreement ranges from 82.3 to 90.5 F_1, indicating that our annotation scheme is appropriate and consistent. We experiment with producing these annotations automatically. For r-NE tagging we train a deep neural network NER tool; to compute flow graphs we train a dependency-style parsing procedure which we apply to the entire sequence of r-NEs in a recipe. In evaluations, our systems achieve 71.1 to 87.5 F_1, demonstrating that our annotation scheme is learnable.

Keywords: cooking recipe corpus, English recipes, recipe named entity, recipe flow graph, procedural text annotation

1. Introduction

Procedural text describes in an objective way how to carry out some task. Although this kind of text is widespread, it is found particularly in instruction manuals and cookbooks. The information in procedural text could be used to support a wide range of computer-based applications involving intelligent search, reasoning, and non-textual forms of human-computer interaction; but first the text would need to be transformed into a more computationally tractable representation. At minimum, this representation would represent domain entities, actions, and relationships between them.

A *flow graph* is one kind of computationally tractable representation; it contains steps linking actions with entities, and specifies the obligatory sequencing relationships between steps. A flow graph abstracts away from differences in surface linguistic expression that are irrelevant to correctly performing a procedure. For example, when cooking a dish, the instructions *add diced carrots to the pot* and *dice carrots and add to the pot* are different textually, but would correspond to the same flow graph since they involve the same actions carried out in the same order.

Our research focuses on the cooking domain, and specifically the procedural part of cooking recipes. Our general approach and some of our computational techniques are inspired by recent research on recipes written in Japanese (Mori et al., 2014). We extend that work by adapting the annotation to English, annotating more data, investigating the consistency of annotation, and improving accuracy of processing.

In this paper, we describe how we annotated a corpus of 300 English recipes, which were contributed to a recipes web site by members of the public. We first annotated each recipe with 'recipe named entities' (or r-NEs). These r-NEs are specific to the recipe domain, and account for cooking actions, cooking implements (tools), ingredients, intermediate products, durations and quantities. The r-NE annotation formed the basis for a second level of annotation: a flow graph representing in detail the relationships between the r-NEs (e.g. an action carried out with a certain tool and food items) and the sequencing of the actions[1].

We used the annotated corpus to investigate approaches for automatic r-NE tagging and flow graph computation. For r-NE tagging, we train a deep neural network NER tool, treating the task as a sequence labelling problem. To derive the flow graph representation, we consider the entire sequence of r-NEs in a recipe as a single input string, and extract a flow graph using a dependency-style maximal spanning tree parser. In contrast to standard sentence-by-sentence parsing, the parser creates flow graph edges that span different sentences in exactly the same way as within-sentence edges; this means we do not need any extra processing stages such as co-reference resolution. Our evaluation results are very promising.

The rest of the paper is structured as follows. Section 2 surveys approaches to representing procedural text, and cooking recipe text annotation and processing. Section 3 describes the r-NE tagset, annotation, and an evaluation of automatic r-NE tagging accuracy. Section 4 describes the flow graph representation, annotation, the parsing algorithm for computing a flow graph, and an evaluation of its accuracy. Section 5 discusses the main findings and proposes directions for future research.

[1]The annotated corpus of English recipes is available at `https://sites.google.com/view/yy-lab/resource/english-recipe-flowgraph`

2. Related Work

There has been much research into representing the semantics of natural language sentences (Banarescu et al., 2013, *inter alia*) and developing parsers that output such representations (Flanigan et al., 2014, for example). Semantic representations encode the meanings of linguistic units within a sentence, but do not attempt to capture domain-specific constraints or real-world context. In some domains and applications these latter factors may be very important. (For example, in cooking, a *mixture* can sometimes be *beaten* and sometimes not, depending on which ingredients have been added to it). For this reason, approaches to processing text describing procedures often do not attempt to construct general semantic representations, but instead construct more specialised domain- and genre-specific representations. For example, Momouchi (1980) proposed representing the meaning of procedural text as a flow graph, and described algorithms to convert text documents to flow graphs. Hamada et al. (2000) adopted a similar graph representation for analysing recipes.

More recent studies have created annotated corpora of procedural text, in order to build machine learning-based systems that extract entity and action information. In the domain of cooking recipes, such information has been shown to be useful for the so-called 'smart kitchen' (Hashimoto et al., 2008), cooking robots (Bollini et al., 2013), etc. In one recent study, Mori et al. (2014) produced flow graph annotations for 266 cooking recipes written in Japanese. That corpus has been used for testing empirical methods for natural language understanding (Maeta et al., 2015) and intelligent search (Yamakata et al., 2013), Similarly, Jermsurawong and Habash (2014) described a corpus of ingredient trees for recipes, and evaluated methods for constructing these automatically. There have been some attempts at unsupervised processing of recipes (Kiddon et al., 2015, *inter alia*) but the outputs are much less rich than those obtained by supervised methods.

This paper concerns a flow graph corpus for recipes in English; the flow graphs are based on the proposals of Mori et al. (2014) in which the graph nodes are 'recipe named entities' (r-NEs). The r-NE types cover domain entities such as ingredients; in addition, Mori et al. argue that r-NE types should also cover actions by the cook / chef (which are often expressed as verbs). Unlike the set of NE types commonly used for general text (Sang and Meulder, 2003)—person name, location, organization, etc.—r-NEs are an example of a domain-specific NE definition. Domain-specific NE definitions are widely used in fields such as bioinformatics (Ben Abacha and Zweigenbaum, 2011). A domain-specific NE definition facilitates the development of intelligent applications that process documents in that particular domain.

The task of identifying NEs is called 'named entity recognition' (NER). NER is usually formulated as a sequence labeling problem, in which each input token is tagged as being inside or outside an NE (and in some approaches, beginning an NE). In this framework many techniques and refinements have been investigated (Borthwick, 1999; Sang and Meulder, 2003; Ratinov and Roth, 2009), and many general-purpose NER tools have been developed. In this study, we use one such tool, PWNER (Sasada et al., 2015a), which computes probabilities for all possible NE tags based on pointwise prediction, and searches for the best sequence of tags under the tag sequence constraints. The tool is distinctive in being trainable on data that has been only partially annotated.

Tag	Meaning	Explanation
F	Food	Eatable; also intermediate products
T	Tool	Knife, container, etc.
D	Duration	Duration of cooking
Q	Quantity	Quantity of food
Ac	Action by chef	Verb representing a chef's action
Ac2	Discontinuous Ac *(English only)*	Second, non-contiguous part of a single action by chef
Af	Action by food	Verb representing action of a food
At	Action by tool *(English only)*	Verb representing a tool's action
Sf	Food state	Food's initial or intermediate state
St	Tool state	Tool's initial or intermediate state

Table 1: Recipe named entity (r-NE) tags.

3. Recipe Named Entities

3.1. Recipe Named Entity Tags

In previous work (Yamakata et al., 2017), we created a corpus of 100 recipes written in English, sampled from the Allrecipes UK/Ireland web site[2] and annotated for 'recipe named entities' (r-NEs). The r-NE tag types were based on a set of eight tags originally devised for annotating recipes written in Japanese (Mori et al., 2014), but with the addition of a further two tags. One is introduced in order to account for linguistic phenomena that occur in English and many other languages but not Japanese: discontinuous multi-word expressions (Constant et al., 2017). The other is to cover expressions for the actions by automatic cooking tools (e.g. food processors), which are not yet common in Japan, so the tag was not necessary for recipes in Japanese. Table 1 lists the ten r-NE tags that we use for English recipe annotation. Note that this tag set is compatible with the original.

Figure 1 shows the annotation that would be given to the recipe step *Preheat oven to 180 C / Gas mark 4*. The tag suffixes -B and -I (abbreviating Begin and Inside respectively) indicate NE spans according to the IOB2 text chunking representation (Sang and Veenstra, 1999). A tag is assigned to each word or word-sequence designating a single and indivisible object/action/phenomenon in the cooking domain. For example, *Gas mark 6* in the example sentence designates the state of the dial of the oven being at 6, so it is annotated as a single r-NE. Therefore, the first word *Gas* is annotated as St-B and the subsequent words *mark* and *6* are both annotated as St-I. The word *to* is annotated O because it is Outside any named entity.

[2]http://allrecipes.co.uk/

Preheat	oven	to	180	C	/	Gas	mark	4	.
Ac-B	T-B	O	St-B	St-I	O	St-B	St-I	St-I	O

Figure 1: Example of r-NE annotation.

3.2. Annotating Recipe Named Entities

Our corpus of English recipes consists of the 100 recipes mentioned above, plus a further 200, which we have selected and annotated recently.

The first, 100-recipe subset (which we call '100-r') was sampled from Allrecipes, selecting the most popular recipes within each dish category (e.g. 'Main course', 'Dessert') in proportion to the total number of recipes in that category. These recipes were annotated by an English native speaker, and all annotations were verified by the first author, as described by Yamakata et al. (2017).

The second, 200-recipe subset ('200-r') was sampled from Allrecipes completely randomly. These recipes were annotated by two Japanese homemakers who had previously annotated Japanese recipes in the study of Mori et al. (2014). Both are good English readers; in preparation, they refamilarized themselves with the annotation guidelines. They annotated the first half of 200-r completely manually; the average annotation time was 19 minutes per recipe (including initial preparation). To speed up the workflow, we automatically tagged the remainder of 200-r using the PWNER NE recognizer trained on 100-r; the annotators then checked and corrected the recognition results. The 300-recipe corpus comprises 38,062 tokens in total.

To estimate inter-annotator agreement, we asked the second pair of annotators to re-annotate 100-r independently. Unfortunately we cannot use Cohen's kappa or related measures, since they assume that each annotation decision is independent of any other; however, with IOB2 some combinations of tags are illegal, e.g. an Inside tag may only occur after a Begin or Inside with the same tag prefix. Instead, we took one of the annotators as the gold standard and evaluated the accuracy of the other with respect to this. We measured the agreement as 89.9% precision, 92.2% recall, and 90.5 F_1. We consider this to be a high level of agreement.

3.3. Automatic Recognition of r-NEs

Having created the 300-recipe r-NE annotated English corpus, we investigated the extent to which the annotation supports NER, and whether the r-NE tagging accuracy is comparable to previous results for Japanese recipes. Although PWNER works well for r-NE tagging, we decided instead to experiment with the more modern deep learning named entity recognizer BERT-NER[3], a state-of-the-art tool based on the BERT neural network architecture (Devlin et al., 2019). For this evaluation we used 10-fold cross-validation, splitting the data into 80% training, 10% hyperparameter tuning, and 10% test. The overall F_1 for r-NE recognition is 87.6. Accuracies by recipe subset are shown in Table 2. Table 3 shows accuracy by tag; F_1 ranges from a low of 17.1 for the rarest tag At to 92.7 for the very common tag Ac.

[3]https://github.com/kyzhouhzau/BERT-NER

Dataset	Precision	Recall	F_1	#NEs
100-r	81.6%	87.4%	84.4	5629
200-r	85.4%	87.9%	86.7	9728
Overall	86.5%	88.8%	87.6	15435

Table 2: r-NE recognition accuracy by corpus subset and overall.

Tag	Precision	Recall	F_1	#NEs
F	90.5%	93.9%	92.1	5007
T	87.7%	90.1%	88.9	1904
D	87.4%	90.2%	88.8	589
Q	70.9%	76.8%	73.7	529
Ac	92.3%	93.1%	92.7	4977
Ac2	43.8%	46.8%	45.3	178
Af	51.4%	50.4%	50.9	255
At	60.0%	10.0%	17.1	15
Sf	66.1%	71.4%	68.6	1105
St	81.6%	82.5%	82.0	876
Total	86.5%	88.8%	87.6	15435

Table 3: r-NE recognition accuracy by tag for the full 300-recipe corpus.

In previous research, Sasada et al. (2015b) trained PWNER on a corpus of 193 Japanese recipes of comparable complexity annotated with the same set of r-NE tags, and reported 84.4 F_1; our F_1 for 200-r is comparable. We also note that F_1 is 2.3 points higher for 200-r than 100-r; this indicates that annotating more data is likely to lead to further worthwhile improvements in NE recognition accuracy. It is likely that annotating more data would be particularly beneficial for the tags Ac2, Af and At, which are relatively rare.

4. Recipe Flow Graphs

4.1. Flow Graph Representation

Mori et al. (2014), in a study of Japanese recipe text, showed how the procedural aspects of a recipe can be represented as a flow graph. A flow graph is a directed acyclic graph (DAG) with a single root. Graph nodes are r-NEs and labelled edges represent the relationships between these nodes. Table 4 lists the flow graph edge labels.

Figure 2 shows the procedural text in a recipe "Almost no fat banana bread" on the Allrecipes web site[4]. Figure 3 contains the flow graph for this recipe. The flow graph makes explicit what actions are applied to which ingredients and using what tools. The graph is read downwards from the top; nodes without any incoming edges represent ingredients, and a node with no outgoing edges denotes an end product. A path of directed edges from an action A to an action B implies that A must be carried out before B. For example, the flow graph explicitly represents the fact that *batter* in sentence 4 is the end product of stirring the contents of the large bowl; this relationship is only implicit

[4]From http://allrecipes.co.uk/recipe/722/almost-no-fat-banana-bread.aspx, February 2020.

Label	Meaning	Explanation
Agent	Subject	Relationship with actions (Ac or Af)
Targ	Direct object	
Dest	Indirect object (container)	
t-comp	tool complement	Tool used in an action
F-comp	Food complement	Food used as a tool
F-eq	Food equality	Identical food
F-part-of	Food part-of	Refer to a part of a food
F-set	Food set	Refer to a set of foods
T-eq	Tool equality	Identical tool
T-part-of	Tool parf-of	Refer to a part of a tool
A-eq	Action equality	Identical action (Ac, Af)
V-tm	Head verb for timing, etc.	
other-mod	Other relationships	

Table 4: Flow graph edge labels.

(#Step=1)-(#Sentence=0): **Preheat oven to 180 C / Gas mark 4**.
(1)-(1): Lightly grease a 20x10cm (8x4 in) loaf tin.
(2)-(2): In a large bowl, stir together flour, sugar, baking powder, bicarbonate of soda and cinnamon.
(2)-(3): Add egg whites, bananas and apple purée; stir just until combined.
(2)-(4): Pour batter into prepared tin.
(3)-(5): **Bake in preheated oven for 50 to 55 minutes, until a skewer inserted into centre of loaf comes out clean.**
(3)-(6): Turn out onto wire rack and allow to cool before slicing.

Figure 2: The recipe "Almost no fat banana bread", comprising 7 sentences (numbered 0 to 6), grouped into 3 steps. Sentences used as examples are in bold.

in the recipe text since *batter* is not previously mentioned. Please note that a flow graph is an internal, symbolic representation designed to support any kind of computation over recipes, and we do not intend that it be displayed verbatim to a cook in a recipe visualization application, for instance.

Starting from the r-NE annotations, the same Japanese homemakers annotated all 300 sentences. The average annotation time was 37 minutes per recipe (including initial preparation); at first it was longer but as the annotators gained experience they got faster. The average annotation time for the final ten recipes was 26 minutes per recipe. To measure inter-annotator agreement, we arranged for both annotators to annotate the 100-r subset. We compute agreement similarly to the previous section, for similar reasons. Taking one of the annotators as the gold standard, the other's annotations had 84.4% precision, 80.4% recall, and 82.3 F_1. Although lower than the r-NE agreement, this is still a good result bearing in mind the very large space of possible edge annotations.

4.2. Automatic Computation of Flow Graphs

Maeta et al. (2015) describe a procedure for computing a flow graph from recipe text. The procedure consists of three steps, which are applied in turn to the entire r-NE tagged recipe, and output a set of directed edges between r-NEs:

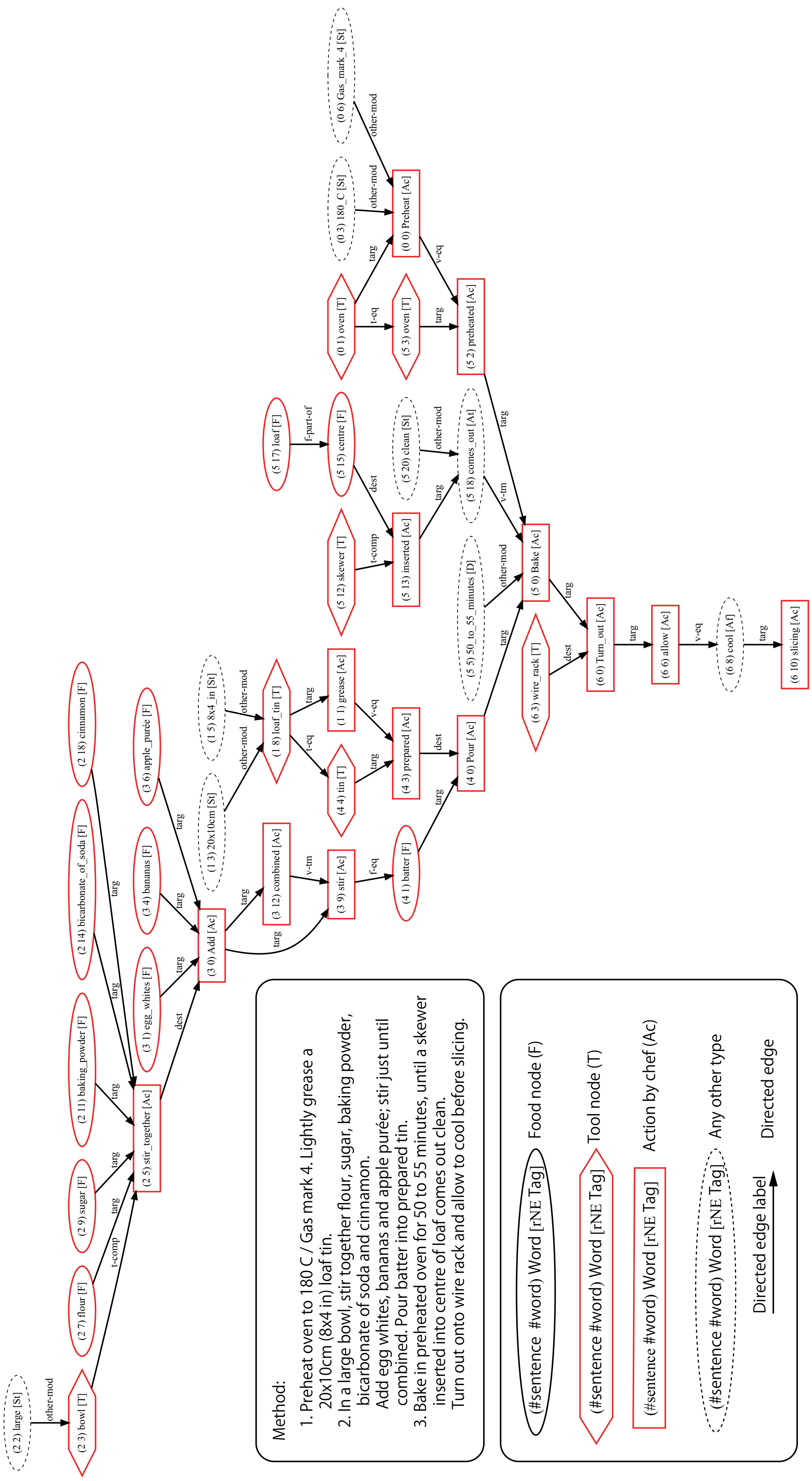
1. Calculate weights of labelled edges between all pairs of r-NEs, using an SVM-based machine learning model.
2. Given the resulting weighted digraph, select a root node and compute a spanning arborescence of minimum weight. A reliable heuristic is to select as root node the Ac that occurs last in the input.
3. Since the previous step can only produce a tree, extend this to a DAG by adding further edges whose weights are below a threshold and which satisfy certain conditions to ensure consistency.

In our work, we apply a similar procedure but ignore labels in step 1 and omit step 3; we output a spanning arborescence with r-NEs as nodes connected by (directed) unlabelled edges[5].
We estimate the weight of a directed edge from node u to v with label l as follows:

$$s(u, v, l) = \frac{\exp\{\boldsymbol{\Theta} \cdot \boldsymbol{f}(u, v, l)\}}{\sum_{(x,r)\in(V\setminus\{u\})\times L} \exp\{\boldsymbol{\Theta} \cdot \boldsymbol{f}(u, x, r)\}} \quad (1)$$

where V is a set of nodes, L is the set of edge labels (see Table 4), $\boldsymbol{\Theta}$ is a weight vector, and $\boldsymbol{f}(u, v, l)$ is a function returning the feature vector for the edge. Intuitively, the right hand side of the equation computes the total weight of the edge's features as a proportion of the grand total of the weights of all edges starting at node u.

[5]An alternative approach would be to parse recipes syntactically in a sentence-by-sentence manner, and then induce flow graph edges and their labels based on paths between nodes in the parses; however, as Maeta et al. (2015) argue, in that case further processing would be required to deal with inter-sentence phenomena such as coreference, anaphora and ellipsis—whereas these do not require any special treatment in our framework.



Method:

1. Preheat oven to 180 C / Gas mark 4. Lightly grease a 20x10cm (8x4 in) loaf tin.
2. In a large bowl, stir together flour, sugar, baking powder, bicarbonate of soda and cinnamon.
   Add egg whites, bananas and apple purée; stir just until combined. Pour batter into prepared tin.
3. Bake in preheated oven for 50 to 55 minutes, until a skewer inserted into centre of loaf comes out clean.
   Turn out onto wire rack and allow to cool before slicing.

Figure 3: Flow graph for the recipe "Almost no fat banana bread".

r-NER	Precision	Recall	F_1
Gold	73.7%	68.6%	71.1
BERT-NER	51.1%	37.7%	43.3

Table 5: Edge estimation accuracy with gold r-NEs and with automatically recognized r-NEs.

The function $\boldsymbol{f}(u, v, l)$ returns a feature vector for an edge, containing information about the nodes u and v that the edge links. The vector comprises the following features:

- words in u and v, and their concatenation;
- concatenation of the r-NE tags of u and v;
- whether u is in the same, a previous or a subsequent sentence as v;
- number of words between u and v (positive if u appears before v, otherwise negative); and
- whether u and v are in the same sentence and there is a preposition between them[6].

In initial experiments, we confirmed that each of these five features improves edge detection accuracy.
The weight vector $\boldsymbol{\Theta}$ is estimated from the training data by a log-linear model (Berger et al., 1996). Given training data consisting of T manually annotated edge-label pairs (u_t, v_t, l_t), we maximize the following value:

$$\sum_{t=1}^{T} s(u_t, v_t, l_t) - \frac{1}{2}\|\boldsymbol{\Theta}\|^2 \qquad (2)$$

To evaluate the accuracy of flow graph construction, we used 10-fold cross-validation, splitting the data into 270 recipes for training and 30 recipes for test. When we started from the gold standard r-NE tagging, the overall F_1 for edge detection was 71.1. When we started instead from the automatic r-NE tagging, the edge detection F_1 reduced to 43.3; the main reason for the large decrease is that an edge will always be wrong if the r-NE at even just one of its two end points is mis-recognized. The results are shown in Table 5.
Table 6 shows the recall of edge detection for each label. Recall is low for several labels: F-eq, F-part-of, F-set, T-eq, T-part-of and A-eq. Edges with these labels often form confluences in the recipe flow graph and therefore cannot be detected in step 2 of the procedure (the step that extracts a spanning arborescence). We intend to address this issue in future work.

4.3. Flow Graph Computation Examples

Figure 4 shows flow graph nodes and edges corresponding to the two sentences labeled (0)-(0) and (3)-(5) in Figure 2. Figure 4(a) shows the manually-annotated gold standard flow graph, (b) shows the flow graph produced automatically starting from gold standard r-NEs, and (c) shows the

[6]To detect prepositional context, we use the RASP grammatical analysis system (Briscoe et al., 2006)—having adapted its lexicon to account for the non-standard writing style of cooking recipes.

Label	#Gold	#Est	Recall
Agent	674	371	55.1%
Targ	6210	5109	82.3%
Dest	1983	1576	79.5%
T-comp	650	540	83.0%
F-comp	286	252	88.2%
F-eq	1139	227	19.9%
F-part-of	737	91	12.3%
F-set	23	2	8.7%
T-eq	316	41	13.0%
T-part-of	190	71	37.5%
A-eq	237	85	35.7%
V-tm	640	474	74.1%
other-mod	2776	2033	73.2%
Total	15,861	10,883	68.6%

Table 6: Edge estimation recall.

flow graph resulting from end-to-end processing (i.e. with r-NE recognition and edge detection[7] both done automatically).
In (b) and (c), our method successfully detected the action equality A-eq between #(0-0) *Preheat* and #(5-2) *Preheated*, even though they are several steps apart. In contrast, tool equality T-eq was not detected between #(0-1) *oven* and #(5-3) *oven*; this exemplifies the low recall of T-eq edges as reported in Table 6. In (b), the out-edge of #(5-17) *loaf* is linked to the wrong node, although all the other edges are correct. In (c), our system failed to detect the r-NEs for *skewer*, *centre* and *inserted*. The destination of the out-edge from *loaf* should be *centre*, but this node does not exist and the edge wrongly connects to the word *Bake*. As a result, the part of the flow graph corresponding to the subordinate clause *until a skewer inserted into centre of loaf comes out clean* fails to correctly represent the semantic structure. However, the edges in the rest of the flow graph are correct.

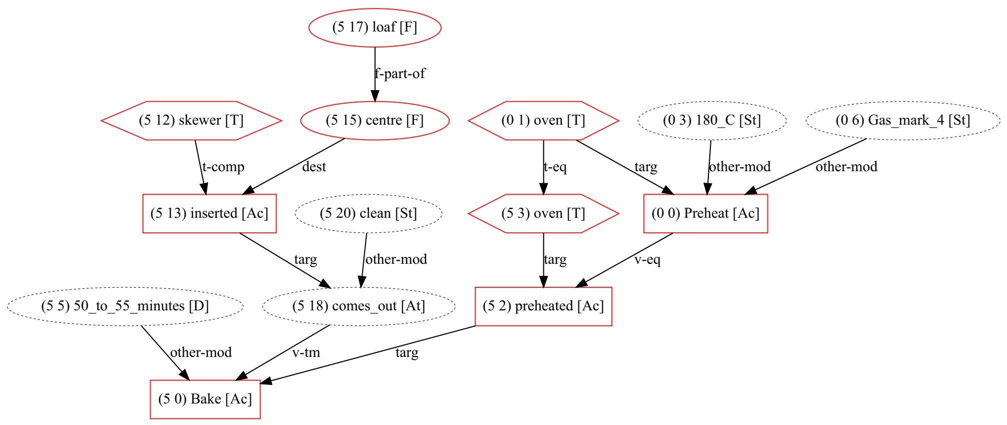
5. Conclusion

Taking as a starting point research on recipes written in Japanese (Mori et al., 2014), we have presented an annotated corpus of English cooking recipe procedures, and described and evaluated computational methods for learning these annotations. In this paper we have demonstrated that:

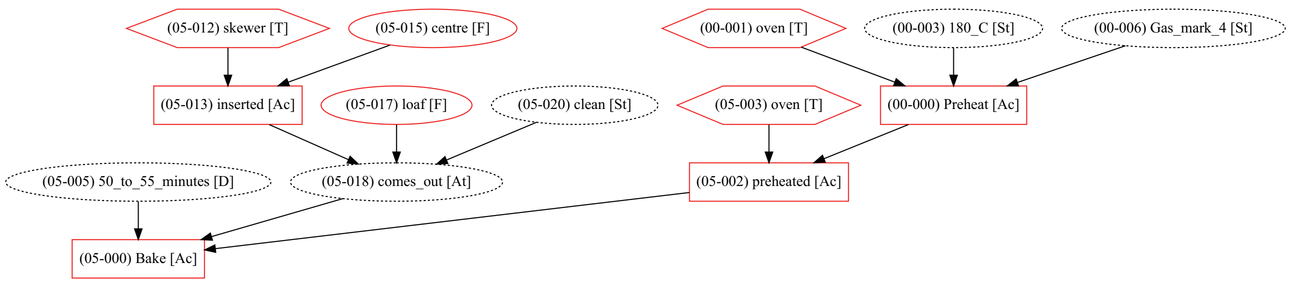
- the approach to annotating and analysing recipes transfers well to a typologically different language;
- the annotation scheme supports high inter-annotator agreement; and
- recipe named entities can be identified with good accuracy using a deep neural network tagger.

The paper described how we annotated each recipe in the corpus with recipe named entities (r-NEs); inter-annotator agreement was high. A deep neural network NER tool

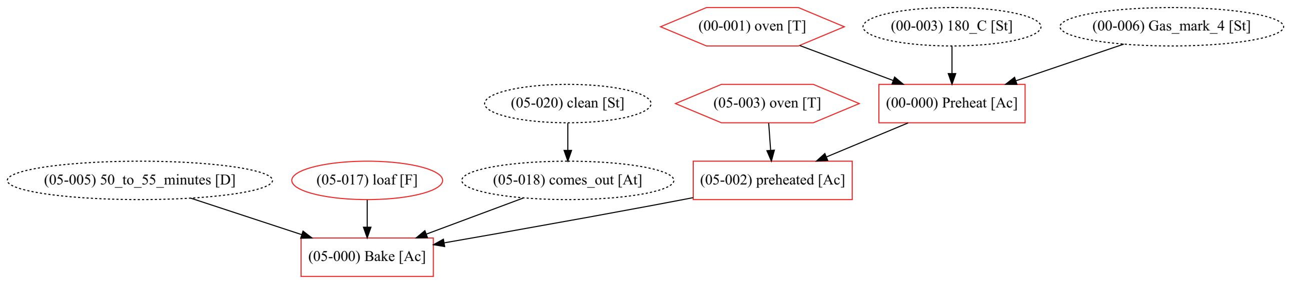
[7]In this setup, we are pipelining two modules; eventually we would like to combine these into a single model by training on both module representations jointly.



(a) Ground truth: manually annotated flow graph.



(b) Flow graph computed from manually annotated r-NEs.



(c) End-to-end flow graph estimation.

Figure 4: Flow graph nodes and edges corresponding to the sentences *Preheat oven to 180 C / Gas mark 4* and *Bake in preheated oven for 50 to 55 minutes, until a skewer inserted into centre of loaf comes out clean.*

trained on this data obtained an overall F_1 of 87.5. This accuracy is comparable to r-NE tagging of Japanese recipes. Starting from the r-NE annotation, inter-annotator agreement for flow-graph annotation was 82.3 F_1. This level of agreement is good, considering the very large space of possible edge annotations. Computing flow graphs used a dependency-style parsing procedure, which achieved an F_1 for edge detection of 71.1. Previous work on analysing Japanese recipes has shown that this level of accuracy is sufficient to support tasks including recipe information retrieval and symbol grounding for cross-modal cooking applications.

In future work, we intend to refine the procedure for extracting a flow graph from the weighted digraph of r-NE nodes. We will experiment with training classifiers for adding labels to edges, and for inserting links that form confluences in the flow graph. Although these are extra processing steps, they would add structure only monotonically to the minimum weight spanning arborescence, so should not have a negative impact on efficiency or on correct structure that is already in the flow graph.

6. Acknowledgements

This work was supported by JSPS KAKENHI Grant Number 18K11425 and ROIS NII Open Collaborative Research 2020 (No. 19S0404).

7. Bibliographical References

Banarescu, L., Bonial, C., Cai, S., Georgescu, M., Griffitt, K., Hermjakob, U., Knight, K., Koehn, P., Palmer, M., and Schneider, N. (2013). Abstract meaning representation for sembanking. In *the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186.

Ben Abacha, A. and Zweigenbaum, P. (2011). Medical entity recognition: A comparison of semantic and statistical methods. In *Proceedings of BioNLP 2011 Workshop*, pages 56–64, Portland, Oregon, USA, June. Association for Computational Linguistics.

Berger, A. L., Pietra, S. A. D., and Pietra, V. J. D. (1996). A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1).

Bollini, M., Tellex, S., Thompson, T., Roy, N., and Rus, D. (2013). Interpreting and executing recipes with a cooking robot. In *Experimental Robotics*, pages 481–495.

Borthwick, A. (1999). *A Maximum Entropy Approach to Named Entity Recognition*. Ph.D. thesis, New York University.

Briscoe, T., Carroll, J., and Watson, R. (2006). The second release of the RASP system. In *Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions*, pages 77–80. Association for Computational Linguistics.

Constant, M., Eryigit, G., Monti, J., van der Plas, L., Rosner, C. R. M., and Todirascu, A. (2017). Multiword expression processing: A survey. *Computational Linguistics*, 43(4):837–892.

Devlin, J., Chang, M. W., Lee, K., and Toutanova, K. (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*, pages 4171–4186.

Flanigan, J., Thomson, S., Carbonell, J., Dyer, C., and Smith, N. A. (2014). A discriminative graph-based parser for the abstract meaning representation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 1426–1436.

Hamada, R., Ide, I., Sakai, S., and Tanaka, H. (2000). Structural analysis of cooking preparation steps in Japanese. In *Proceedings of the fifth International Workshop on Information Retrieval with Asian Languages*, pages 157–164.

Hashimoto, A., Mori, N., Funatomi, T., Yamakata, Y., Kakusho, K., and Minoh, M. (2008). Smart kitchen: A user centric cooking support system. In *Proceedings of the 12th Information Processing and Management of Uncertainty in Knowledge-Based Systems*, pages 848–854.

Jermsurawong, J. and Habash, N. (2014). Predicting the structure of cooking recipes. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2370–2377.

Kiddon, C., Ponnuraj, G. T., Zettlemoyer, L., and Choi, Y. (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.

Maeta, H., Sasada, T., and Mori, S. (2015). A framework for procedural text understanding. In *Proceedings of the 14th International Conference on Parsing Technologies*, pages 50–60.

Momouchi, Y. (1980). Control structures for actions in procedural texts and PT-Chart. In *Proceedings of the Eighth International Conference on Computational Linguistics*, pages 108–114.

Mori, S., Maeta, H., Yamakata, Y., and Sasada, T. (2014). Flow graph corpus from recipe texts. In *Proceedings of the Nineth International Conference on Language Resources and Evaluation*, pages 2370–2377.

Ratinov, L. and Roth, D. (2009). Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 147–155, Boulder, Colorado, June. Association for Computational Linguistics.

Sang, E. F. T. K. and Meulder, F. D. (2003). Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Computational Natural Language Learning*, pages 142–147.

Sang, E. F. T. K. and Veenstra, J. (1999). Representing text chunks. In *Proceedings of EACL'99*, pages 173–179.

Sasada, T., Mori, S., Kawahara, T., and Yamakata, Y. (2015a). Named entity recognizer trainable from partially annotated data. In *Proceedings of the Eleventh International Conference Pacific Association for Computational Linguistics*.

Sasada, T., Mori, S., Yamakata, Y., Maeta, H., and Kawahara, T. (2015b). Definition of recipe terms and corpus annotation for their automatic recognition (in Japanese). *Journal of Natural Language Processing*, 22(2):107–131.

Yamakata, Y., Imahori, S., Sugiyama, Y., Mori, S., and Tanaka, K. (2013). Feature extraction and summarization of recipes using flow graph. In *Proceedings of the 5th International Conference on Social Informatics*, LNCS 8238, pages 241–254.

Yamakata, Y., Carroll, J., and Mori, S. (2017). A comparison of cooking recipe named entities between Japanese and English. In *Proceedings of the 9th Workshop on Multimedia for Cooking and Eating Activities, in Conjunction with IJCAI 2017*, pages 7–12.