

Structure-Aware Generation Network for Recipe Generation from Images

Hao Wang^{1,2}, Guosheng Lin¹, Steven C. H. Hoi³, and Chunyan Miao^{1,2*}

¹ School of Computer Science and Engineering,
Nanyang Technological University, Singapore

² Joint NTU-UBC Research Centre of Excellence
in Active Living for the Elderly, NTU

³ Singapore Management University

{hao005,gslin,ascymiao}@ntu.edu.sg chhoi@smu.edu.sg

Abstract. Sharing food has become very popular with the development of social media. For many real-world applications, people are keen to know the underlying recipes of a food item. In this paper, we are interested in automatically generating cooking instructions for food. We investigate an open research task of generating cooking instructions based on only food images and ingredients, which is similar to the image captioning task. However, compared with image captioning datasets, the target recipes are long-length paragraphs and do not have annotations on structure information. To address the above limitations, we propose a novel framework of Structure-aware Generation Network (SGN) to tackle the food recipe generation task. Our approach brings together several novel ideas in a systematic framework: (1) exploiting an unsupervised learning approach to obtain the sentence-level tree structure labels before training; (2) generating trees of target recipes from images with the supervision of tree structure labels learned from (1); and (3) integrating the inferred tree structures with the recipe generation procedure. Our proposed model can produce high-quality and coherent recipes, and achieve the state-of-the-art performance on the benchmark Recipe1M dataset.

Keywords: Structure Learning, Text Generation, Image-to-Text

1 Introduction

Food-related research with the newly evolved deep learning-based techniques is becoming a popular topic, as food is essential to human life. One of the important and challenging tasks under the food research domain is recipe generation [23], where we are producing the corresponding and coherent cooking instructions for specific food.

In the recipe generation dataset Recipe1M [24], we generate the recipes conditioned on food images and ingredients. The general task setting of recipe generation is almost the same as that of image captioning [6]. Both of them target

* Corresponding author

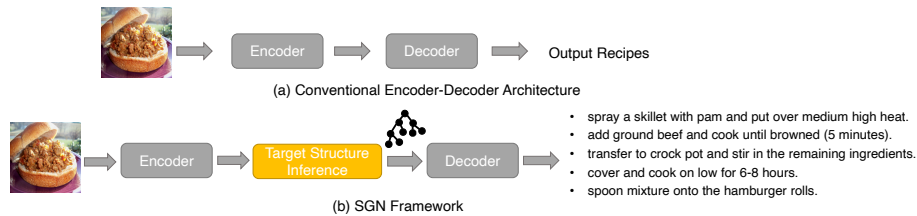


Fig. 1. Comparison between conventional image captioning model and our proposed Structure-aware Generation Network (SGN) for recipe generation. Before generating target recipes, we infer the tree structures of recipes first, then we use graph attention networks to give tree embeddings. Based on the structure information, we can generate better recipes.

generating a description of an image by deep models. However, there still exist two big differences between recipe generation and image captioning: (i) the target caption length and (ii) annotations on structural information.

First, most popular image captioning datasets, such as Flickr [22] and MS-COCO dataset [6], only have one sentence per caption. By contrast, cooking instructions are paragraphs, containing multiple sentences to guide the cooking process, which cannot be fully shown in single food image. Although Recipe1M has ingredient information, the ingredients are actually mixed in cooked food images. Therefore, generating lengthy recipes using traditional image captioning model may hardly capture the whole cooking procedure. Second, the lack of structural information labeling is another challenge in recipe generation. For example, MS-COCO has precise bounding box annotations in images, giving scene graph information for caption generation. This structural information provided by the official dataset makes it easier to recognize the objects, their attributes and relationships within an image. While in food images, different ingredients are mixed when cooked. Hence, it is difficult to obtain the detection labeling for food images.

Benefiting from recent advances in language parsing, some research, such as ON-LSTM [26], utilizes an unsupervised way to produce word-level parsing trees of sentences and achieve good results. Inspired by that, we extend the ON-LSTM architecture to do sentence-level tree structure generation. We propose to train the extended ON-LSTM with quick thoughts manner [18], to capture the order information inside recipes. By doing so, we get the recipe tree structure labels.

After we obtained the recipe structure information, we propose a novel framework named **Structure-aware Generation Network (SGN)** to integrate the tree structure information with the training and inference phases. SGN is implemented to add a target structure inference module on the recipe generation process. Specifically, we propose to use a RNN to generate the recipe tree structures from food images. Based on the generated trees, we adopt the graph attention networks to embed the trees, in an attempt to giving the model more guid-

ance when generating recipes. With the tree structure embeddings, we make the generated recipes remain long-length as the ground truth, and improve the generation performance considerably.

Our contributions can be summarized as:

- We propose a recipe2tree module to capture sentence-level tree structures for recipes, which is adopted to supervise the following img2tree module. We use an unsupervised approach to learn latent tree structures.
- We propose to use the img2tree module to generate recipe tree structures from food images, where we adopt a RNN for conditional tree generation.
- We propose to utilize the tree2recipe module, which encodes the inferred tree structures. It is implemented with graph attention networks, and boosts the recipe generation performance.

Figure 1 shows a comparison between vanilla image captioning model and our proposed SGN. SGN outperforms state-of-the-art baselines for recipe generation on Recipe1M dataset [24] significantly. We conduct extensive experiments to evaluate performance of SGN. Finally, we present qualitative results of our proposed methods, and visualizations of the generated recipe results.

2 Related Work

2.1 Image captioning

Image captioning task is defined as generating the corresponding text descriptions from images. Based on MS-COCO dataset [6], most existing image captioning techniques adopt deep learning-based model. One popular approach is Encoder-Decoder architecture [29, 1, 9, 14], where a CNN is used to obtain the image features along with object detection, then a language model is used to convert the image features into text.

Since image features are fed only at the beginning stage of generation process, the language model may face vanishing gradient problem [12]. Therefore, image captioning model is facing challenges in long sentence generation [1]. To enhance text generation process, [5, 10] involve scene graph into the framework. However, scene graph generation rely heavily on object bounding box labeling, which is provided by MS-COCO dataset. When we shift to some other datasets without rich annotation, we can hardly obtain the graph structure information of the target text. Meanwhile, crowdsourcing annotation is high-cost and may not be reliable. Therefore, we propose to produce tree structures for paragraphs unsupervisedly, helping the recipe generation task in Recipe1M dataset [24].

2.2 Multimodal Food Computing

Food computing [20] has raised great interest recently, it targets applying computational approaches for analyzing multimodal food data for recognition [2], retrieval [24, 30, 31] and generation [23] of food. Recipe generation is a challenging task, it is mainly because that cooking instructions (recipes) contain

more than one sentence. Besides, the input images of food datasets [24, 33, 4] suffer from quality issues. Among the existing food datasets, YouCook2 [33] and Storyboarding [4] use image sequence as input, which is not a general image captioning task setting. Recipe1M [24, 23] contains static food image, ingredient and cooking instruction information for each food sample. Therefore, we choose Recipe1M [24] to evaluate our proposed methods.

2.3 Language Parsing

Parsing is served as one effective language analysis tool, it can output the tree structure of a string of symbols. Generally, language parsing is divided into word-level and sentence-level parsing. Word-level parsing is also known as grammar induction, which aims at learning the syntactic tree structure from corpora data. Especially, Shen et al. [26] propose to use ON-LSTM, which equips the LSTM architecture with an inductive bias towards learning latent tree structures. They train the model with normal language modeling way, at the same time they can get the parsing output induced by the model.

Sentence-level parsing is used to identify the elementary discourse units in a text, and it brings some benefits to discourse analysis. Many recent works attempted to use complex model with labeled data to achieve the goal [13, 17]. Here we extend ON-LSTM [26] for unsupervised sentence-level parsing, which is trained using quick thoughts [18].

3 Method

We now present our proposed model SGN for recipe generation from food images. We show the framework of our proposed model in Figure 2.

3.1 Overview

Given the food images and ingredients, our goal is to generate the cooking instructions. Different from image captioning task in MS-COCO [6, 19], where the target caption only have one sentence, the cooking instruction is a paragraph, containing more than one sentence, and the maximum sentence number in Recipe1M dataset [24] is 19. If we infer the recipes directly from the images, i.e. use a decoder conditioned on image features for generation [23], it is difficult for model to fully capture the structured cooking steps. That may result in the generated paragraphs incomplete. Hence, we believe it necessary to infer paragraph structure during recipe generation phase.

To infer the sentence-level tree structure from food images, we need labels to supervise the tree generation process. However, in Recipe1M dataset [24], there has no paragraph tree structure labeling for cooking instructions. And it is very time-consuming and unreliable to use crowdsourcing to give labels. Therefore, in the first step, we use the proposed recipe2tree module to produce the tree

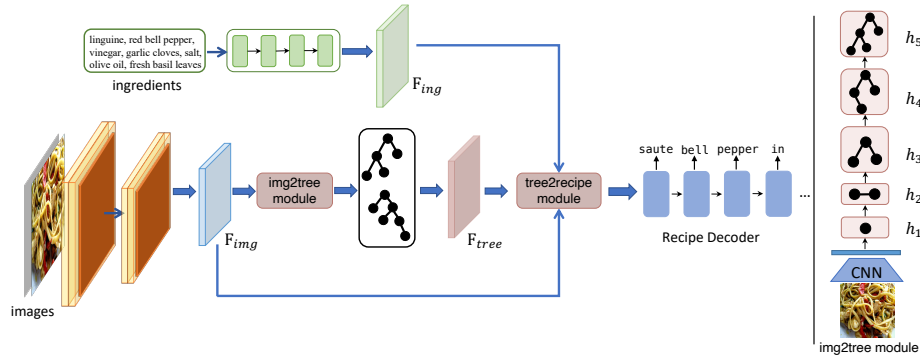


Fig. 2. Left: our proposed framework for effective recipe generation. **Right:** the `img2tree` module. The ingredients and food images are embedded by a pretrained language model and CNN respectively to produce the output features F_{ing} and F_{img} . Before language generation, we first infer the tree structure of target cooking instructions. To do so, we utilize the `img2tree` module, where a RNN produces the nodes and edge links step-by-step based on F_{img} . Then in `tree2recipe` module, we adopt graph attention networks (GAT) to encode the generated tree adjacency matrix, and get the tree embedding F_{tree} . We combine F_{ing} , F_{img} and F_{tree} to construct a final embedding for recipe generation, which is performed using a transformer.

structure labels with an unsupervised way. Technically, we use hierarchical ON-LSTM [26] to encode the cooking instructions and train the ON-LSTM with quick thoughts approach [18]. Then we can obtain the latent tree structure of cooking instructions.

During training phase, we input food images and ingredients to our proposed model. We try two different language models to encode ingredients, i.e. non-pretrained and pretrained model, to get the ingredient features F_{ing} . In non-pretrained model training, we use one word embedding layer to give F_{ing} . Besides, we adopt BERT [8] for ingredient embedding, which is one of the state-of-the-art NLP pretrained models.

In image process branch, we adopt a CNN to encode the food images and get the image features F_{img} . Based on F_{img} , we generate the trees and make them align with that produced by the `recipe2tree` module. Specifically, we transform the tree structure to a 1-dimensional adjacency sequence for RNN to generate, where the RNN’s initial state is image feature F_{img} . To incorporate the generated tree structure into the recipe generation process, we get the tree embedding F_{tree} with graph attention networks (GATs) [28], and concatenate it with the image features F_{img} and ingredient features F_{ing} . We then generate the recipes conditioned on the joint features $\langle F_{tree}, F_{img}, F_{ing} \rangle$ with a transformer [27].

Our proposed framework is optimized over two objectives: to generate reasonable recipes given the food images and ingredients; and to produce the sentence-

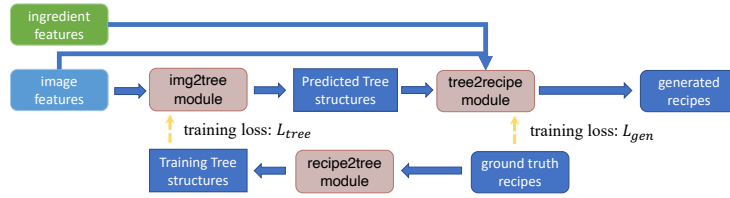


Fig. 3. The concise training flow of our proposed SGN.

level tree structures of target recipes. The overall objective is given as:

$$L = \lambda_1 L_{gen} + \lambda_2 L_{tree} \quad (1)$$

where λ_1 and λ_2 are trade-off parameters. L_{gen} controls the recipe generation training with the input of $\langle F_{tree}, F_{img}, F_{ing} \rangle$, and outputs the probabilities of word tokens. L_{tree} is the tree generation loss, supervising the img2tree to generate trees from images. The training flow is shown in Figure 3.

3.2 ON-LSTM Revisit

Ordered Neurons LSTM (ON-LSTM) [26] is proposed to infer the underlying tree-like structure of language while learning the word representation. It can achieve good performance in unsupervised parsing task. ON-LSTM is constructed based on the intuition that each node in the tree can be represented by a set of neurons in the hidden states of recurrent neural networks. To this end, ordered neurons is an inductive bias, where high-ranking neurons store long-term information, while low-ranking neurons contain short-term information that can be rapidly forgotten. Instead of acting independently on each neuron, the gates of ON-LSTM are dependent on the others by enforcing the order in which neurons should be updated. Hence, ON-LSTM is able to discern a hierarchy of information between neurons. With the learnt ranking, the top-down greedy parsing algorithm [25] is used for unsupervised constituency parsing. However, ON-LSTM is originally trained by language modeling way and learns the word-level order information. To unsupervisedly produce sentence-level tree structure, we extend ON-LSTM in recipe2tree module.

3.3 Recipe2tree Module

In recipe2tree module, we propose to use hierarchical ON-LSTM, i.e. word-level and sentence-level ON-LSTM, to train Recipe1M data. Specifically, in word-level ON-LSTM, we input the word tokens and use the output features as the sentence embeddings. The sentence embeddings will be fed into the sentence-level ON-LSTM for end-to-end training.

Since the original training way [26], such as language modeling or seq2seq [3] word prediction training, can not be used in sentence representation learning, we incorporate the idea of quick thoughts (QT) [18] to supervise the hierarchical ON-LSTM training. The general objective of QT is a discriminative approximation where the model attempts to identify the embedding of a correct target sentence given a set of sentence candidates. In other words, instead of predicting *what is the next* in language modeling, we predict *which is the next* in QT training to capture the order information inside recipes. Technically, for each recipe data, we select first $N - 1$ of the cooking instruction sentences as context, i.e. $S_{ctx} = \{s_1, \dots, s_{N-1}\}$. Then sentence s_N turns out to be the correct next one. Besides, we randomly select K sentences along with the correct sentence s_N from each recipe, to construct candidate sentence set $S_{cand} = \{s_N, s_i, \dots, s_k\}$. The candidate sentence features $g(S_{cand})$ are generated by the word-level ON-LSTM, and the context embeddings $f(S_{ctx})$ are obtained from the sentence-level ON-LSTM. The computation of probability is given by

$$p(s_{cand}|S_{ctx}, S_{cand}) = \frac{\exp[c(f(S_{ctx}), g(s_{cand}))]}{\sum_{s' \in S_{cand}} \exp[c(f(S_{ctx}), g(s'))]} \quad (2)$$

where c is an inner product, to avoid the model learning poor sentence encoders and a rich classifier. Minimizing the number of parameters in the classifier encourages the encoders to learn disentangled and useful representations [18]. The training objective maximizes the probability of identifying the correct next sentences for each training recipe data D .

$$\sum_{s \in D} \log p(s|S_{ctx}, S_{cand}) \quad (3)$$

We show some qualitative results of sentence-level tree structure for recipe data in Figure 4.

3.4 Img2tree Module

In img2tree module, we generate the tree structures from food images. Tree structure has hierarchical nature, in other words, “parent” node is always one step higher in the hierarchy than “child” nodes. Given the properties, we first represent the trees as sequence under the hierarchical ordering. Then, we use an auto-regressive model to model the sequence, meaning that the edges between subsequent nodes are dependent on the previous “parent” node. Besides, in Recipe1M dataset, the longest cooking instructions have 19 sentences. Therefore, the sentence-level parsing trees have limited node numbers, which avoids the model generating too long or complex sequence.

In Figure 2, we specify our tree generation approach. The generation process is conditioned on the food images. We first map the tree structure to an adjacency matrix according to the hierarchical ordering, which is denoted the links between nodes by 0 or 1. Then the lower triangular part of the adjacency matrix will be

converted to a vector $V \in \mathbb{R}^{n \times 1}$, where each element $V_i \in \{0, 1\}^i, i \in \{1, \dots, n\}$. Since edges in tree structure are undirected, V can determine a unique tree T .

Here the tree generation model is built based on the food images, capturing how previous nodes are interconnected and how following nodes construct edges linking previous nodes. Hence, we adopt Recurrent Neural Networks (RNN) to model the predefined sequence V . We use the image encoded features F_{img} as the initialization of RNN hidden state, and the state-transition function h and the output function y are formulated as:

$$h_0 = F_{img}, h_i = f_{trans}(h_{i-1}, V_{i-1}), \quad (4)$$

$$y_i = f_{out}(h_i), \quad (5)$$

where h_i is conditioned on the previous generated $i - 1$ nodes, y_i outputs the probabilities of next node’s adjacency vector.

The tree generation objective function is:

$$p(V) = \prod_{i=1}^n p(V_i | V_1, \dots, V_{i-1}), \quad (6)$$

$$L_{tree} = \sum_{V \in D} \log p(V), \quad (7)$$

where $p(V)$ is the product of conditional distributions over the elements, D denotes for all the training data.

3.5 Tree2recipe Module

In tree2recipe module, we utilize graph attention networks (GATs) [28] to encode the generated trees. The input of GATs is the generated tree adjacency matrix z . We produce node features with a linear transformation W , which is applied to the adjacency matrix z . We then perform attention mechanism a on the nodes and compute the attention coefficients

$$e_{ij} = a(Wz_i, Wz_j) \quad (8)$$

where e_{ij} measures the importance of node j ’s features to node i .

It is notable that different from most attention mechanism, where every node attends on every other node, GATs only allow each node to attend on its neighbour nodes. The underlying reason is that doing global attention fails to consider the property of tree structure, that each node has limited links to others. While the local attention mechanism used in GATs preserves the structural information well. We can formulate the final attentional score as:

$$\alpha_{ij} = \text{softmax}_j(e_{ij}) = \frac{\exp(e_{ij})}{\sum_{k \in N_i} \exp(e_{ik})} \quad (9)$$

where N_i is the neighborhood of node i , the output score is normalized through the softmax function. Similar with [27], GATs employ multi-head attention and

averaging to stabilize the learning process. We get the tree features by the product of the attentional scores and the node features, and we perform nonlinear activation on the output to get the final features:

$$F_{tree} = \sigma\left(\sum_{j \in N_i} \alpha_{ij} W z_j\right). \quad (10)$$

3.6 Recipe Generation

We adopt a transformer [27] for recipe generation. The input of transformer is the combination of previous obtained features F_{img} , F_{ing} and F_{tree} . The transformer decoder output the token $\hat{x}^{(i)}$ one by one during inference. The training objective is to maximize the following objective:

$$L_{gen} = \sum_{i=0}^M \log p(\hat{x}^{(i)} = x^{(i)}) \quad (11)$$

where L_{gen} is the recipe generation loss, and M is the maximum sentence generation length, $x^{(i)}$ denotes for the ground truth token.

4 Experiments

4.1 Dataset and Evaluation Metrics

We evaluate our proposed structure-aware generation network (SGN) on Recipe1M dataset [24], which is one of the largest collection of cooking recipe data with food images. Recipe1M has rich food related information, including the food images, ingredients and cooking instructions. In Recipe1M, there are 252, 547, 54, 255 and 54, 506 food data samples for training, validation and test respectively. These recipe data is collected from some public websites, which are uploaded by users.

We evaluate the model using the same metrics as prior work [23]: perplexity. Besides, we extend to use more metrics to evaluate our proposed method. Specifically, we use perplexity, BLEU [21] and ROUGE [16]. Perplexity is used in [23], it measures how well the learned word probability distribution matches the target recipes. BLEU is computed based on the average of unigram, bigram, trigram and 4-gram precision. We use ROUGE-L to test the longest common subsequence. ROUGE-L is a modification of BLEU, focusing on recall rather than precision. Therefore, we can use ROUGE-L to measure the fluency of generated recipes.

4.2 Implementation Details

We adopt a 3-layer ON-LSTM [26] to output the sentence-level tree structure, taking about 50 epoch training to get converged. We set the learning rate as

1, batch size as 60, and the input embedding size is 400, which is the same as original work [26]. We select recipes containing over 4 sentences in Recipe1M dataset for training. And we randomly select several consecutive sentences as the context and the following one as the correct one. We set K as 3. We show some of the predicted sentence-level tree structures for recipes in Figure 4.

To test if our proposed SGN can be applied to different systems, we tried two different ingredient encoders in the experiments, i.e. non-pretrained and pretrained language model. Using non-pretrained model is to compare with the prior work [23], where they use a word embedding layer to give the ingredient embeddings. We use BERT [8] as the pretrained language model, giving 512-dimensional features. The image encoder is used with a ResNet-50 [11] pretrained on ImageNet [7]. And we map image output features to the dimension of 512, to align with the ingredient features. We adopt a RNN for tree adjacency sequence generation, where the RNN initial hidden state is initialized as the previous image features. The RNN layer is set as 2 and the hidden state size is 512. The tree embedding model is graph attention network, its attention head number is set as 6. The output tree feature dimension is set the same as that of image features.

We use the same settings in language decoder as prior work [23], a 16-layer transformer [27]. The number of attention heads in the decoder is set as 8. We use greedy search during text generation, and the maximum generated instruction length is 150.

It is notable that, we use ground truth ingredients and images as input in all experiments for a fair comparison. We set λ_1 and λ_2 in Eq.11 as 1 and 0.5 respectively. The model is trained using Adam [15] optimizer with the batch size of 16. Initial learning rate is set as 0.001, which decays 0.99 each epoch. The BERT model finetune learning rate is 0.0004.

4.3 Baselines

Since Recipe1M has different data components from standard MS-COCO dataset [6], it is hard to implement some prior image captioning model in Recipe1M. To the best of our knowledge, [23] is the only recipe generation work on Recipe1M dataset, where they use the Encoder-Decoder architecture. Based on the ingredient and image features, they generate the recipes with transformer [27].

The SGN model we proposed is an extension of the baseline model, which learns the sentence-level tree structure of target recipes by an unsupervised approach. We infer the tree structures of recipes before language generation, adding an additional module on the baseline model. It means that our proposed SGN can be applied to many other deep model architectures and vision-language datasets. We test the performance of SGN with two ingredient encoders, 1) non-pretrained word embedding model and 2) pretrained BERT model. Word embedding model is used in [23], trained from scratch. BERT model [8] is served as another baseline, to test if SGN can improve language generation performance further under a powerful encoder. We use ResNet-50 in both two baseline models.

Table 1. Main Results. Evaluation of SGN performance against different settings. We test the performance of two baseline models for comparison. We evaluate the model with perplexity, BLEU and ROUGE-L.

Methods	Perplexity	BLEU	ROUGE-L
Non-pretrained Model [23]	8.06	7.23	31.8
+SGN	7.46	9.09	33.4
Pretrained Model [8]	7.52	9.29	34.8
+SGN	6.67	12.75	36.9

Table 2. Recipe Average Length. Comparison on average length between recipes from different sources.

Methods	Recipe Average Length
Pretrained Model [8]	66.9
+SGN	112.5
Ground Truth (Human)	116.5

4.4 Main Results

Generated Recipe Evaluation We show the performance of SGN for recipe generation against the baselines in Table 1. In both baseline settings, our proposed method SGN outperforms the baselines across all metrics. In the method of non-pretrained model, SGN achieves a BLEU score more than 9.00, which is about 25% higher than the current state-of-the-art method. When we shift to the pretrained model method [8], we can see that the pretrained language model gets comparable results as “non-pretrained model + SGN” model, achieving better generation performance than the baseline model [23]. When incrementally adding SGN to pretrained model, the performance of SGN is significantly superior to all the baselines by a substantial margin. On the whole, the efficacy of SGN is shown to be very promising, outperforming the state-of-the-art method across different metrics consistently.

Impact of Structure Awareness To explicitly suggest the impact of tree structures on the final recipe generation, we compute the average length for the generated recipes, as shown in Table 2. Average length can show the text structure on node numbers. SGN generates recipes with the most similar length as the ground truth, indicating the help of the tree structure awareness.

4.5 Qualitative Results

Sentence-Level Tree Parsing Results In Figure 4, we visualize some parsing tree results of our proposed recipe2tree module. Due to there is no human

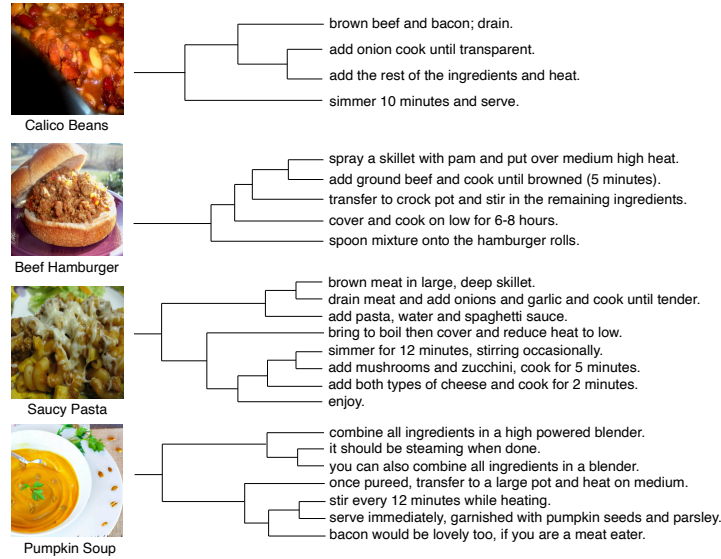


Fig. 4. The visualization of predicted sentence-level trees for recipes. The latent tree structure is obtained from unsupervised learning. The results indicate that we can get reasonable parsing tree structures with varying recipe length.

labelling on the recipe tree structures, we can hardly provide a quantitative analysis on the parsing trees.

We show some examples with varying paragraph length in Figure 4. The first two rows show the tree structures of relatively short recipes. Take the first row (*calico beans*) as example, the generated tree set the food pre-processing part (step 1) as a separate leaf node, and two main cooking steps (step 2&3) are set as deeper level nodes. The last *simmer* step is conditioned on previous three steps, which is put in another different tree level. We can see that the parsing tree results correspond with common sense and human experience.

In the last two rows of Figure 4, we show the parsing results of recipes having more than 5 sentences. The tree of *pumpkin soup* indicates clearly two main cooking phases, i.e. before and after ingredient pureeing. Generally, the proposed *recipe2tree* generated sentence-level parsing trees look plausible, helping on the inference for recipe generation.

Recipe Generation Results We present some recipe generation results in Figure 5. We consider three types of recipe sources, the human, models trained without and with SGN. Each recipe accompanies with a food image. We can observe that recipes generated by model with SGN have similar length with that written by users. It may indicate that, instead of generating language directly




			
User	In a greased oval 5-6 quart slow cooker, combine the vegetables and broth. Place drumsticks over vegetables. Sprinkle remaining ingredients over all. Cover and cook on low for 5-5 1/2 hours, or until meat thermometer registers 180° f.	In medium bowl, gently mix together chicken, bread crumbs, egg, milk, lemon juice, mint, oregano, salt and pepper until well combined. Shape into 4 patties each about 3/4 inch thick. In a large nonstick skillet heat oil over medium-high heat. Cook patties about 8 minutes turning once, until golden brown and no longer pink inside. Meanwhile, cut pitas in half crosswise to make 4 pockets. Warm pitas in 300 degree oven wrapped in aluminum foil for 5 minutes or in the microwave for 1 minute. Spread inside of pitas with mayo then place cooked patty inside with a slice of red onion, tomato, and some lettuce. Enjoy.	Bring a medium size pot of water to a boil, add rice. Bring back to a boil, then reduce heat to simmer. Let rice simmer 15-20 minutes, until tender. Place beans and rice in a medium size saucepan. Heat over a medium heat, stirring frequently. Stir in reserved bean liquid as needed. Remove pan from heat and stir in lemon juice, garlic powder and cilantro. Let sit a moment, and stir in fresh oregano. Serve immediately.
Model without SGN	Place all ingredients in crock pot. Cook on low for 8 hours.	In a large bowl, combine the ground chicken, bread crumbs, egg, milk, lemon juice, oregano, mint, salt and pepper. Mix well. Shape into 4 patties. In a large skillet, heat oil over medium heat. Cook patties for 5 minutes on each side or until cooked through. Serve with lemon wedges.	In a medium saucepan, bring water to a boil. Stir in rice, reduce heat to low, cover, and simmer until rice is tender, about 20 minutes. Stir in black beans, lemon juice, garlic powder, and cilantro.
Model with SGN	In a large pot, combine potatoes, carrots, celery, onion, and garlic. Add chicken broth, salt, pepper, and thyme. Bring to a boil, reduce heat and simmer for 20 minutes. Add potatoes and simmer for another 10 minutes. Add cream and simmer for another 10 minutes. Serve with crusty bread. Enjoy!	Combine chicken, bread crumbs, egg, milk, lemon juice, oregano, mint, salt and pepper in a large bowl. Shape into 4 patties. Heat oil in a large nonstick skillet over medium-high heat. Cook patties until golden brown, about 5 minutes per side. Transfer to a plate. Add tomato slices to skillet and cook until lightly browned, about 1 minute. Add onion and cook until tender, about 2 minutes. Stir in tomato sauce and cook until heated through, about 1 minute. Serve with lettuce, tomato slices, and burgers. Note: you can substitute any combination of the tomato slices, onion, and garlic.	Bring rice and water to a boil in a saucepan. Reduce heat to low, cover, and simmer until rice is tender, 20 to 25 minutes. Stir black beans, lemon juice, garlic powder, and cilantro into rice; cook until heated through, about 5 more minutes. Serve warm. Enjoy!

Fig. 5. Visualization of recipes from different sources. We show the food images and the corresponding recipes, obtained from users and different types of models. Words in red indicate the matching parts between recipes uploaded by users and that generated by models. Words in yellow background show the redundant generated sentences.

from the image features, allowing the deep model to be aware of the structure first brings benefits for the following recipe generation task.

We indicate the matching parts between recipes provided by users and that generated by models, in red words. It is observed that SGN model can produce more coherent and detailed recipes than non-SGN model. For example, in the middle column of Figure 5, SGN generated recipes include some ingredients that do not exist in the non-SGN generation, but are contained in users’ recipes, such as *onion*, *lettuce* and *tomato*.

However, although SGN can generate longer recipes than non-SGN model, it may produce some redundant sentences. These useless sentences are marked with yellow background, as shown in the first column of Figure 5. Since *Cream* is not supposed to be used in the chicken soup, in the future work, we may need to use the input ingredient information better to guide the recipe generation.

Tree Generation Results There are some graph evaluation metrics proposed in [32], however, these metrics are used for unconditional graph evaluation. How to evaluate the graph similarities for conditional generation remains an open problem. Here we show some examples of generated recipe tree structures in Figure 6 for qualitative analysis. Tree generation results from image features are by-product of our proposed SGN framework. They are used to improve the final recipe generation performance.

It is notable that only the leaf nodes in the tree represent the sentences of recipe. We can observe that the overall img2tree generated structures look similar with the ground truth trees, which are produced by recipe2tree module.










Images	Ground Truth	RNN Generation	Cooking Instructions
			boil chicken with just enough water to cover garlic powder. add homestyle noodles two garlic water. in separate pot cook onions celery carrots garlic in broth until almost done. add vegetables broth and chicken and noodles together. desired thickness I like mine the consistency of egg drop soup. add with grilled cheese sandwich and enjoy.
			saute onion in olive oil until just beginning to brown. add tomato and cook until tender. add chicken broth and beans (do not drain). cook for 20 minutes or until to desired temperature. place in serving bowls and dust with grated cheese.
			combine coffee, almond milk, ice cubes and cottage cheese in a blender. blend for about 25 seconds or until ice cubes are no longer chunky. pour into cup of choice and drink!

Fig. 6. The comparison between the ground truth trees (produced by recipe2tree module) and img2tree generated tree structures.

And the generated trees have some diversity. However, it is hard to align the number of generated nodes with the ground truth. For example, in the last row of Figure 6, the generated tree has one more node than the ground truth.

5 Conclusion

In this paper, we have proposed a structure-aware generation network (**SGN**) for recipe generation, where we are the first to implement the idea of inferring the target language structure to guide the text generation procedure. We propose effective ways to address some challenging problems, including unsupervisedly extracting the paragraph structures, generating tree structures from images and using the produced trees for recipe generation. Specifically, we extend ON-LSTM to label recipe tree structures using an unsupervised manner. We propose to use RNN to generate the tree structures from food images, and adopt the inferred trees to enhance the recipe generation. We have conducted extensive experiments and achieved state-of-the-art results in Recipe1M dataset for recipe generation.

Acknowledgement

This research is supported, in part, by the National Research Foundation (NRF), Singapore under its AI Singapore Programme (AISG Award No: AISG-GC-2019-003) and under its NRF Investigatorship Programme (NRFI Award No. NRF-NRFI05-2019-0002). Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not reflect the views of National Research Foundation, Singapore. This research is also supported, in part, by the Singapore Ministry of Health under its National Innovation Challenge on Active and Confident Ageing (NIC Project No. MOH/NIC/COG04/2017 and MOH/NIC/HAIG03/2017), and the MOE Tier-1 research grants: RG28/18 (S) and RG22/19 (S).

References

1. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473 (2014)
2. Bossard, L., Guillaumin, M., Van Gool, L.: Food-101—mining discriminative components with random forests. In: European conference on computer vision. pp. 446–461. Springer (2014)
3. Britz, D., Goldie, A., Luong, M.T., Le, Q.: Massive exploration of neural machine translation architectures. arXiv preprint arXiv:1703.03906 (2017)
4. Chandu, K.R., Nyberg, E., Black, A.: Storyboarding of recipes: Grounded contextual generation (2019)
5. Chen, L., Zhang, H., Xiao, J., He, X., Pu, S., Chang, S.F.: Counterfactual critic multi-agent training for scene graph generation. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 4613–4623 (2019)
6. Chen, X., Fang, H., Lin, T.Y., Vedantam, R., Gupta, S., Dollár, P., Zitnick, C.L.: Microsoft coco captions: Data collection and evaluation server. arXiv preprint arXiv:1504.00325 (2015)
7. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
8. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
9. Golland, D., Liang, P., Klein, D.: A game-theoretic approach to generating spatial descriptions. In: Proceedings of the 2010 conference on empirical methods in natural language processing. pp. 410–419. Association for Computational Linguistics (2010)
10. Gu, J., Joty, S., Cai, J., Zhao, H., Yang, X., Wang, G.: Unpaired image captioning via scene graph alignments. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 10323–10332 (2019)
11. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
12. Hossain, M.Z., Sohel, F., Shiratuddin, M.F., Laga, H.: A comprehensive survey of deep learning for image captioning. *ACM Computing Surveys (CSUR)* **51**(6), 1–36 (2019)
13. Jia, Y., Ye, Y., Feng, Y., Lai, Y., Yan, R., Zhao, D.: Modeling discourse cohesion for discourse parsing via memory network. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers). pp. 438–443 (2018)
14. Kazemzadeh, S., Ordonez, V., Matten, M., Berg, T.: Referitgame: Referring to objects in photographs of natural scenes. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). pp. 787–798 (2014)
15. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
16. Lin, C.Y.: ROUGE: A package for automatic evaluation of summaries. In: Text Summarization Branches Out. pp. 74–81. Association for Computational Linguistics, Barcelona, Spain (Jul 2004)
17. Lin, X., Joty, S., Jwalapuram, P., Bari, S.: A unified linear-time framework for sentence-level discourse parsing. arXiv preprint arXiv:1905.05682 (2019)

18. Logeswaran, L., Lee, H.: An efficient framework for learning sentence representations. arXiv preprint arXiv:1803.02893 (2018)
19. Luo, R., Price, B., Cohen, S., Shakhnarovich, G.: Discriminability objective for training descriptive captions. arXiv preprint arXiv:1803.04376 (2018)
20. Min, W., Jiang, S., Liu, L., Rui, Y., Jain, R.: A survey on food computing. *ACM Computing Surveys (CSUR)* **52**(5), 1–36 (2019)
21. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: Bleu: a method for automatic evaluation of machine translation. In: *Proceedings of the 40th annual meeting on association for computational linguistics*. pp. 311–318. Association for Computational Linguistics (2002)
22. Plummer, B.A., Wang, L., Cervantes, C.M., Caicedo, J.C., Hockenmaier, J., Lazebnik, S.: Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In: *Proceedings of the IEEE international conference on computer vision*. pp. 2641–2649 (2015)
23. Salvador, A., Drozdal, M., Giro-i Nieto, X., Romero, A.: Inverse cooking: Recipe generation from food images. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 10453–10462 (2019)
24. Salvador, A., Hynes, N., Aytar, Y., Marin, J., Ofli, F., Weber, I., Torralba, A.: Learning cross-modal embeddings for cooking recipes and food images. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. pp. 3020–3028 (2017)
25. Shen, Y., Lin, Z., Huang, C.W., Courville, A.: Neural language modeling by jointly learning syntax and lexicon. arXiv preprint arXiv:1711.02013 (2017)
26. Shen, Y., Tan, S., Sordani, A., Courville, A.: Ordered neurons: Integrating tree structures into recurrent neural networks. arXiv preprint arXiv:1810.09536 (2018)
27. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: *Advances in neural information processing systems*. pp. 5998–6008 (2017)
28. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)
29. Vinyals, O., Toshev, A., Bengio, S., Erhan, D.: Show and tell: A neural image caption generator. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. pp. 3156–3164 (2015)
30. Wang, H., Sahoo, D., Liu, C., Lim, E.p., Hoi, S.C.: Learning cross-modal embeddings with adversarial networks for cooking recipes and food images. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 11572–11581 (2019)
31. Wang, H., Sahoo, D., Liu, C., Shu, K., Achananuparp, P., Lim, E.p., Hoi, S.C.: Cross-modal food retrieval: Learning a joint embedding of food images and recipes with semantic consistency and attention mechanism. arXiv preprint arXiv:2003.03955 (2020)
32. You, J., Ying, R., Ren, X., Hamilton, W.L., Leskovec, J.: Graphrnn: Generating realistic graphs with deep auto-regressive models. arXiv preprint arXiv:1802.08773 (2018)
33. Zhou, L., Xu, C., Corso, J.J.: Towards automatic learning of procedures from web instructional videos. In: *Thirty-Second AAAI Conference on Artificial Intelligence* (2018)