CHEF: Cross-Modal Hierarchical Embeddings for Food Domain Retrieval

Hai X. Pham[1], **Ricardo Guerrero**[1], **Vladimir Pavlovic**[1,2], **Jiatong Li**[2]
[1] Samsung AI Center, Cambridge
[2] Department of Computer Science, Rutgers University
{hai.xuanpham, r.guerrero, v.pavlovic}@samsung.com, jl2312@rutgers.edu

Abstract

Despite the abundance of multi-modal data, such as image-text pairs, there has been little effort in understanding the individual entities and their different roles in the construction of these data instances. In this work, we endeavour to discover the entities and their corresponding importance in cooking recipes *automatically* as a visual-linguistic association problem. More specifically, we introduce a novel cross-modal learning framework to jointly model the latent representations of images and text in the food image-recipe association and retrieval tasks. This model allows one to discover complex functional and hierarchical relationships between images and text, and among textual parts of a recipe including title, ingredients and cooking instructions. Our experiments show that by making use of efficient tree-structured Long Short-Term Memory as the text encoder in our computational cross-modal retrieval framework, we are not only able to identify the main ingredients and cooking actions in the recipe descriptions without explicit supervision, but we can also learn more meaningful feature representations of food recipes, appropriate for challenging cross-modal retrieval and recipe adaption tasks.

Introduction

Computer vision and natural language processing have witnessed outstanding improvements in recent years. Computational food analysis (CFA) broadly refers to methods that attempt automating food understanding, and as such, it has recently received increased attention, in part due to its importance in health and general wellbeing (Min et al. 2019). For instance, CFA can play an important role in assessing and learning the functional similarity and interaction of ingredients, cooking methods and meal preferences, while aiding in computational meal preparation and planning (Teng, Lin, and Adamic 2012; Helmy et al. 2015). However, despite recent efforts CFA still poses specific and difficult challenges due to the highly heterogeneous and complex nature of the cooking transformation process. Further to this, a particular modality may offer only a partial "view"of the item, for example, a cooking recipe often describe elements that can easily be occluded in the visual depiction of a cooked dish, and/or come in a variety of colors, forms and textures (e.g., ingredients such as tomatoes can be green, yellow or red and can also be presented as a sauce, chunks or whole).

Recent approaches that aim at learning the translation between visual and textual representations of food items do so by learning the semantics of objects in a shared latent space (Salvador et al. 2017; Chen et al. 2018; Carvalho et al. 2018; Wang et al. 2019; Marín et al. 2019). Here, representations (also called embeddings) derived from multi-modal evidence sources (e.g., images, text, video, flavours, etc.) that belong to the same item are matched. In effect, this type of approach aims to find a common grounding language that describes items independent of their observed modality, therefore, allowing cross-modal retrieval. Recently, recurrent neural network (RNN) architectures such as Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber 1997) units and Gated Recurrent Units (GRU) (Cho et al. 2014) have (re-)emerged as two popular and effective models that are able to capture some long-term dependencies in sequential data. Previous works on cross-modal image-to-recipe (test) retrieval in the food domain treat textual elements (e.g., words) as a linear sequence in a RNN (Salvador et al. 2017; Chen et al. 2018; Carvalho et al. 2018; Wang et al. 2019; Marín et al. 2019). However, natural language exhibits syntactic properties that would naturally combine words into phrases in a not necessarily sequential fashion (Tai, Socher, and Manning 2015). Chain structured RNNs (such as LSTMs) struggle to capture this type of relationship. Tree-LSTM offers a generalization of LSTMs to tree-structured network topologies (Tai, Socher, and Manning 2015; Zhu, Sobihani, and Guo 2015), further to this, recent advancements in Tree-LSTMs allow online learning of the sequence structure (Choi, Min Yoo, and Lee 2017). In this work, we argue that these recent advancements in Tree-LSTM structure learning are specially well suited to discover the underlying syntactic structure specific to food cooking recipes, exclusively through the signal provided by its pairing with its visual representation (food dish image).

Motivation

One of the more general goals of the work proposed here is to have a system that is able to "understand" food. This is a very broad and challenging task that requires an understanding of not only the visual aspect of a meal, but also understanding what are its basic constituents and how they are processed and transformed into the final dish. Recipes offer an instruction manual as to how to prepare a dish; they specify the building blocks (ingredients), how to process them (instructions) and a succinct summary of the dish in the form of a title. Additionally, an image of a food also provides a type of dish summary in visual form, e.g., it is often possible to see main ingredients and deduce principal cooking techniques from them. As our main task is to delve deeper in the understanding of food, we would naturally focus on as many representations as possible, however, in this work we will focus primarily on recipes and images.

Some key questions that people regularly ask themselves regarding food are: what it is and how it is made. In the proposed framework we aim at learning distinct textual entities that underline a particular dish in an unsupervised way. The proposed framework, driven solely by information arising by paired data (images and recipes), is able to understand concepts such as what is the main ingredient in this dish, thus answering the "what it is". This information can be valuable to recommendation systems, which would benefit from understanding what is the main ingredient in a recipe. For example, a person querying for apple recipes is unlikely to be interested in recipes where apples are only a minor ingredient. In order to facilitate this, it is important to also understand the key actions that are required to create a dish. Food preparation actions describe types of dishes, which can impact the likelihood of an ingredient being either major or minor. For example, assuming "apple" is in the list of ingredients of a dish, while showing the action "bake" as prominent, it is more likely that apples are main ingredient as opposed to a recipe that where the action "grill" is the most prominent. Furthermore, the ability to deeply understand food, through simple pairing of images and recipes, enables the possibility of better "ingredient swap recipe retrieval", where the goal is to find the dishes similar to the one described except for the main ingredient, which is replaced.

To address some of the aforementioned challenges, we propose a novel cross-modal retrieval computational framework that can effectively learn translations between images of food dishes and their corresponding preparation recipes' textual descriptions. Special emphasis is given to the functional and hierarchical relationship between text and images through the use of Tree-LSTMs. We show that using Tree-LSTMs offers not only a better representation of sentences in the context of cross-modal retrieval, but also allow us to discover important aspects of the data, that are normally lost in sequential RNNs.

In summary our contributions are: (1) a hierarchical cross-modal retrieval framework that allows the discovery of important syntactic concepts, such as ingredient importance, keywords and/or action words, exclusively through visual-text pairings, while also providing (2) extensive experiments that demonstrate state-of-the-art performance in the image-to-recipe retrieval task as well as various recipe modifications enabled by Tree-LSTM. Source code of our proposed method is available at *https://github.com/haixpham/CHEF*.

Cross-Modal Association Model

Cross-modal learning is an active research topic in computer science. In general terms, it describes a system that given a view or modality (e.g., image) of an instance, it retrieves the same instance but as viewed in another modality (e.g., text). These type models are usually trained using a direct correspondence between pairs of instances in different modalities. In the case of food recipe retrieval, these modalities are usually food images and their associated text descriptions (title, ingredients, recipe, etc.).

In order to extract feature representations from both images and recipes (text), we base our architecture on a simplified version of the cross-modal association model presented by Salvador et al. (Salvador et al. 2017). Different to their model, we additionally make use of titles and replace the pre-embedding of instructions with an online instruction embedding module. Such model is trained to match recipes (a concatenation of the encoding of title, ingredients and instructions) and their corresponding images in a joint latent space. Our general cross-modal framework is shown in Fig. 1. During training, the model's objective is formulated as the minimization of the distance between an anchor recipe $\boldsymbol{r}^+$ and matching image $\boldsymbol{v}^+$, while also maximizing (up to a margin ϵ) the distance between the anchor recipe $\boldsymbol{r}^+$ and a non-matching image $\boldsymbol{v}^-$, that is, it minimizes the margin triplet loss of $(\boldsymbol{r}^+, \boldsymbol{v}^+, \boldsymbol{v}^-)$. Using two separate neural networks, one for text encoding $\mathrm{F_p}$ and another for image encoding $\mathrm{F_q}$, each item of the triplet is embedded in a latent space with coordinates $(\boldsymbol{p}^+, \boldsymbol{q}^+, \boldsymbol{q}^-)$.

Formally, with the text encoder $\boldsymbol{p} = \mathrm{F_p}(\boldsymbol{r})$ and image encoder $\boldsymbol{q} = \mathrm{F_q}(\boldsymbol{v})$, the training is a minimization of the following objective function,

$$
\begin{aligned}
&V(\mathrm{F_p}, \mathrm{F_q}) = \\
&\mathbb{E}_{\hat{p}(\boldsymbol{r}^+, \boldsymbol{v}^+), \hat{p}(\boldsymbol{v}^-)} \min\left(\left[d\left[\boldsymbol{p}^+, \boldsymbol{q}^+\right] - d\left[\boldsymbol{p}^+, \boldsymbol{q}^-\right] - \epsilon\right], 0\right) + \\
&\mathbb{E}_{\hat{p}(\boldsymbol{r}^+, \boldsymbol{v}^+), \hat{p}(\boldsymbol{r}^-)} \min\left(\left[d\left[\boldsymbol{p}^+, \boldsymbol{q}^+\right] - d\left[\boldsymbol{p}^-, \boldsymbol{q}^+\right] - \epsilon\right], 0\right),
\end{aligned}
\tag{1}
$$

where $d\left[\boldsymbol{p}, \boldsymbol{q}\right] = \cos\left[\boldsymbol{p}, \boldsymbol{q}\right] = \boldsymbol{p}^\intercal \boldsymbol{q} / \sqrt{(\boldsymbol{p}^\intercal \boldsymbol{p})(\boldsymbol{q}^\intercal \boldsymbol{q})}$ is the cosine similarity in the latent space and $\hat{p}$ denotes the corresponding empirical densities on the training set. The cosine similarity of the positive pair and that of the negative pair together are combined with a margin ϵ, whose goal is to focus the model on "hard" examples (negatives within the margin) while ignoring those that are "good enough" (beyond the margin). We empirically set ϵ to 0.3 by cross-validation.

LSTM Text Encoder

The text encoder $\mathrm{F_p}$ takes the recipe's title, ingredients and instructions as input, and outputs their feature representation in the shared latent space. The goal is to find an embedding that reflects dependencies between a recipe's textual and visual depictions, which could facilitate implicit associations even when some components are present in only one of the two modalities, e.g., not visible but described in text or not described in text but visible. For this purpose, the model first converts word tokens into vectors ($w_i \in \mathbb{R}^{300}$) using a word2vec model (Mikolov et al. 2013), treating each

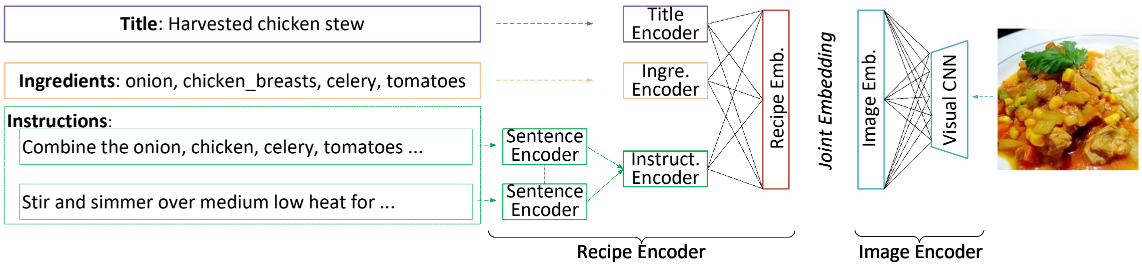


Figure 1: The general cross-modal retrieval framework, including the image encoder and recipe encoder. Within the recipe encoder, each sentence is encoded by a sub-network, and their outputs are further mapped into instruction-level latent features to be concatenated with title and ingredient embeddings.

vector w_i as part of a sequenced input to a bi-directional LSTM. The word2vec model is pre-trained on all textual information available from the recipes, then fine-tuned while training the cross-modal retrieval system.

During training, the model learns a contextual vector $\boldsymbol{h_m} \in \mathbb{R}^{600}$ for each of the m parts of the textual information, that is, title, ingredients and instructions, before concatenating them into a single vector $\boldsymbol{h}$. Finally, $\boldsymbol{h}$ is projected to the shared space by three fully connected (FC) layers, each of dimensionality 1024, to yield the latent text features $\boldsymbol{p} \in \mathbb{R}^{1024}$. We therefore consider four LSTM modules, one each for title and ingredients, and two for instructions. The first level of the instruction encoder module encodes each instruction's word sequence, then the second encodes the sequence of instructions into a single vector representation.

Tree-LSTM Text Encoder

As mentioned before, the text encoder $\mathrm{F_p}$ is tasked with encoding all textual information. RNNs are popular techniques to encode text, and more specifically, LSTMs (along with GRUs) have shown, to some degree, being able to capture the semantic meaning of text. LSTMs assume a chain graph can approximate arbitrary word dependencies within a recipe, however, this structure might not fully capture complex relationships between words. Therefore, it might be beneficial to model word dependencies as a tree structure. Tree-LSTM (Choi, Min Yoo, and Lee 2017; Tai, Socher, and Manning 2015; Zhu, Sobihani, and Guo 2015) offers an elegant generalisation of LSTMs, where information flow from children to parent is controlled using a similar mechanism to a LSTM. Tree-LSTM introduces cell state in computing parent representation, which assists each cell to capture distant vertical dependencies, thus breaking the inherent linearity of a chain LSTM. The following formulas are used to compute the model's parent representation from its children in the special case of a binary tree-LSTM:

$$\begin{bmatrix} \mathbf{i} \\ \mathbf{f}_l \\ \mathbf{f}_r \\ \mathbf{o} \\ \mathbf{g} \end{bmatrix} = \begin{bmatrix} \sigma \\ \sigma \\ \sigma \\ \sigma \\ \tanh \end{bmatrix} \left(\mathbf{W}_{\text{comp}} \begin{bmatrix} \mathbf{h}_l \\ \mathbf{h}_r \end{bmatrix} + \mathbf{b}_{\text{comp}} \right) \tag{2}$$

$$\mathbf{c}_p = \mathbf{f}_l \odot \mathbf{c}_l + \mathbf{f}_r \odot \mathbf{c}_r + \mathbf{i} \odot \mathbf{g} \tag{3}$$

$$\mathbf{h}_p = \mathbf{o} \odot \tanh\left(\mathbf{c_p}\right) \tag{4}$$

where σ is the sigmoid activation function, the pairs $\langle \mathbf{h}_l, \mathbf{c}_l \rangle$ and $\langle \mathbf{h}_r, \mathbf{c}_r \rangle$ are the two input tree nodes popped off the stack, $\mathbf{W}_{\text{comp}} \in \mathbb{R}^{5D_h \times 2D_h}$, $\mathbf{b}_{\text{comp}} \in \mathbb{R}^{2D_h}$ and $\odot$ is the element-wise product. The result of this function, the pair $\langle \mathbf{h_p}, \mathbf{c_p} \rangle$, is placed back on the stack. Note that the formulation used here follows that of (Choi, Min Yoo, and Lee 2017), which in turn is similar to (Bowman et al. 2016).

(2) is similar to that of traditional LSTM equations except that instead of a single forget gate, there is one for each possible child of a node in the tree. More specifically to the formulation here, there are two forget gates, $\mathbf{f}_l$ and $\mathbf{f}_r$, corresponding to left and right children in a binary tree. In this work, we make use of the Gumbel-softmax Tree-LSTM model (Choi, Min Yoo, and Lee 2017), which can learn the tree structure representation without supervision.

Image Encoder

The image encoder $\mathrm{F_q}$ takes an image as input and generates its feature representation in the shared latent space. ResNet50 (He et al. 2016) pre-trained on ImageNet is used as the backbone for feature extraction, where the last FC layer is replaced with three consecutive FC layers (similar to the recipe encoder) to project the extracted features into the shared latent space to get $\boldsymbol{q} \in \mathbb{R}^{1024}$. Particularly, the middle FC layer is shared with the recipe encoder in order to preliminarily align the two modalities' distributions.

Experiments

In this section we will use L, T, G, and S as shorthand for LSTM, Tree-LSTM, GRU and Set (Zaheer et al. 2017), respectively. In this work, we use GRU to encode short sequences and LSTM for longer ones. Thus, the title encoder is chosen as G through out our experiments. The ingredient encoder is chosen amongst {S, G, T}, where S means all ingredients contribute equally to the final embedding. The sentence and instruction encoders are either L or T. Different configurations of the recipe encoder are shown in Tab. 1 and subsequent tables, in which a 3-tuple such as **[G+T+L]** means the model uses GRU as ingredient encoder, Tree-LSTM for sentence encoder and LSTM for the instruction-level encoder. All models are trained end-to-end. Particularly, the attention-based model introduced by Chen et al. (Chen et al. 2018) is similar to one variant of our framework, **[G+L+L]**, where the attention mechanism is integrated into every text encoder. However, this model is trained within our experimental setting, which lead to improved performance in general.

We evaluate our proposed models in four different tasks, including (i) cross-modal retrieval, (ii) main ingredients detection (including ingredient pruning), (iii) ingredient substitution and (iv) action words extraction. Additionally, we compare the performance of our models to that of the original R1M pic2rec model (Salvador et al. 2017) and the state-of-the-art ACME (Wang et al. 2019), retrained to adapt to our vocabulary (cf. Sec.), while faithfully following their original settings. We also retrained Adamine (Carvalho et al. 2018) using publicly available source code, however, we were unable to make it work adequately with our modified data, thus we did not include it in this paper.

Dataset

During the preparation of the work presented here, all experiments were conducted using data from Recipe1M (R1M) (Salvador et al. 2017; Marín et al. 2019). This dataset consists of ∼1M text recipes that contain titles, instructions and ingredients. Additionally, a subset of ∼0.5M recipes contain at least one image per recipe. Data is split into 70% train, 15% validation and 15% test sets. During training at most 5 images from each recipe are used, while a single image is kept for validation and testing.

Canonical Ingredient Construction. To analyze ingredient relative importance across different recipes, a standardized version of R1M ingredients was created. R1M contains ∼16k unique ingredients, with the top 4k accounting for ∼95% occurrences. Focusing on these, we reduced them to ∼1.4k through the following operations. First, ingredients are merged if they have the same name after stemming and singular/plural conversion. Second, ingredients are merged if they are close together in our word2vec (Mikolov et al. 2013) embedding space, if they share two or three words or are mapped to the same item by Nutritionix[1]. Lastly, mergers are inspected by a human who can accept or reject them.

Cross-Modal Recipe Retrieval

The cross-modal retrieval performance of our proposed models are compared against the baselines in Tab. 1. We report results on two subsets of randomly selected 1,000 and 10,000 recipe-image pairs from the test set, similar to (Wang et al. 2019). These experiments are repeated 10 times, and we report the averaged results. All variants of our proposed framework, except **[S+T+T]**, outperform the current state-of-the-art, ACME - retrained on the modified dataset, across all evaluation metrics. Interestingly, the attention-based model (Chen et al. 2018), when trained within our paradigm, achieves significant gain over ACME, and it scores the best median rank (*medR*) on the recipe-to-image retrieval task. It is worth noting that, we were unable to reproduce the results of ACME as reported by the authors on the original, unmodified dataset (more analysis of ACME performance included in the supplement).

Our proposed models have similar performance scores, and amongst those, **[T+L+T]** is the best performer in most cases. More importantly, these empirical evidences where **[T+L+L]** and **[G+L+L]** perform better than **[S+L+L]** suggest that ingredients hold significant information to recognize a dish from its image, thus a more effective ingredient encoder like Tree-LSTM or GRU with attention will be able to embed more meaningful information in the food space, which also improves the capability of the image encoder when they are trained jointly using our proposed triplet loss (1), which is not regularized by semantics information, unlike pic2rec and ACME. Furthermore, Tree-LSTM explicitly imposes implicit importance scores to different ingredients due to its inherent hierarchical structure, hence this

[1]https://www.nutritionix.com/

Methods	medR↓	R@1↑	R@5↑	R@10↑
Size of test set: 1k				
pic2rec	4.10	26.8	55.8	67.5
ACME	2.00	45.4	75.0	83.7
S+L+L	1.80	48.0	77.0	84.1
S+T+T	2.20	37.7	68.0	78.7
G+L+L	**1.60**	49.3	78.1	85.2
G+T+L	1.80	49.0	78.0	85.8
G+T+T	1.80	48.7	78.3	85.7
T+L+L	1.80	49.4	**79.6**	86.1
T+L+T	**1.60**	**49.7**	79.3	**86.3**
T+T+L	1.75	49.0	78.8	85.9
T+T+T	1.70	49.4	78.8	85.9
Size of test set: 10k				
pic2rec	33.25	7.7	21.8	30.8
ACME	9.40	18.0	40.3	52.0
S+L+L	8.10	19.6	42.8	54.5
S+T+T	15.90	13.2	31.8	42.8
G+L+L	7.50	20.7	44.7	56.2
G+T+L	7.60	20.7	44.3	55.9
G+T+T	7.50	**20.9**	44.5	56.0
T+L+L	**7.30**	**20.9**	**44.8**	**56.3**
T+L+T	**7.30**	20.7	44.7	56.2
T+T+L	7.50	20.8	44.3	**56.3**
T+T+T	**7.30**	**20.9**	44.6	56.1

Table 1: Image-to-Recipe retrieval performance comparison between model variants of our proposed framework and the baselines. The Recipe-to-Image retrieval performance is similar, and is included in the supplementary materials. The models are evaluated on medR (lower is better) and Recall@K (R@K - higher is better). In this table and subsequent tables, our proposed models are ordered by type of ingredient - sentence - instruction encoders. Best results are marked in bold.

Model	Split	medR↓	R@1↑	R@2↑	R@3↑
Tree-LSTM	Test	1.0	47.0	78.9	95.5
	Val.	1.0	47.3	79.7	95.5
Attention	Test	4.0	9.1	19.5	32.1
	Val.	4.0	9.5	19.7	32.2

Table 2: Main ingredient prediction performance. The models evaluated on medR (lower is better) and Recall@K (R@K - higher is better).

observation encourages us to analyze the roles of ingredients in the joint food space more thoroughly in the next two sections. Sec. will also show that Tree-LSTM is better at attending to the important ingredients than the soft attention mechanism.

Main Ingredients Detection

One benefit of using Tree-LSTM to encode text is that, the information flow in the network follows a learned hierarchical structure, and the closer the word (its corresponding leaf node in the tree) is to the root, the more importance it holds in the text. Thus, intuitively when embedding the list of ingredients, Tree-LSTM should be able to infer a hierarchy that emphasizes the more important ingredients, including the main one. We extract tree structures of all ingredient embeddings generated by Tree-LSTM in the test set, and calculate their average depths.

We observed that all top ingredients have the average depths in the range of 2.x, which is very close to the root. In other words, the Tree-LSTM ingredient encoder has learned the hierarchical representations that impose higher importance on some ingredients. An example of ingredient tree is illustrated in Fig. 2, where the main ingredient, "green_breans" is at the lowest leaf of the tree, meaning it is the most important ingredient of this dish, as the model has learned. More examples are included in the supplementary materials. However, these observations still do not answer the question, are these top ingredients always the main ones in their respective recipes?

We conjecture that the main ingredient of each recipe is likely the one with the largest mass among all ingredients. In order to validate this intuition, a subset of ∼30k test and validation samples (15k each) from Recipe1M has been curated to include ingredient amounts and units. In this data, ingredient amounts and units have been normalized to a standard total weight of 1kg, e.g. fluid units were converted to weight. An experiment was designed where the gold standard for a recipe's main ingredient was assumed to be well aligned with the amount by weight. Therefore, such ingredient should be predicted as the shallowest during Tree-LSTM inference. From the Tree-LSTM embedding of each of these recipes we can quantify the rank of the *true* main ingredient with regards to the root, that is, what is the distance between the true main ingredient and the shallowest ingredient in the tree. Tab. 2 summarizes these results, and as it can be seen in ∼47% of the cases the correct main ingredient is identified. Given the median recipe contains 9 ingredients (with maximum ingredient depth of 8), a random naïve predictor would rank ingredients as 4, ∼95% of our predicted ranks are better than chance. This table also includes the prediction performance of the attention-based approach in(Chen et al. 2018), which one would expect to "attend" to the main ingredient in a recipe. However, as the results indicate, the attention mechanism fails to uncover the main ingredients.

Ingredient Pruning As Tree-LSTM ingredient encoder has the unique ability to address the importance of ingredients, a set of experiments were conducted in which we removed the K least important ingredients in the list, corresponding to the K highest leaves in the tree, and their occurrences in the instructions were also removed. The retrieval performance of these novel (ingredient pruned) recipes using the **[T+T+L]** recipe encoder is demonstrated in Tab. 3. This table shows that removing the least important ingredient of each recipe (K=1) actually brought significant performance gain: medR of image-to-recipe and recipe-to-image retrievals are 1.1 and 1.0, respectively, while R@1 improves by 2 points. Removing two ingredients also improves retrieval performance, without learning a new model. The performance after removing three ingredients is roughly equal to that of the original recipes, and performance starts decreasing after removing four ingredients, which makes sense as on average four ingredients constitute 25% of the ingredient list. Overall, these results reaffirm the ability of Tree-LSTM ingredient encoder to attend to the more important ingredients in the recipes, and suggest the exploration of a self-pruning recipe encoder that may improve performance of downstream tasks.

	Image-to-Recipe			Recipe-to-Image		
K	**medR↓**	**R@1↑**	**R@5↑**	**medR↓**	**R@1↑**	**R@5↑**
0	1.75	49.0	78.8	1.60	49.7	78.9
1	**1.10**	**51.1**	**79.9**	**1.00**	**51.9**	**79.9**
2	1.50	50.0	79.4	1.20	50.8	79.3
3	1.85	48.7	78.2	1.80	49.4	77.8
4	1.90	46.3	76.6	1.95	47.8	76.6

Table 3: Retrieval performance of **[T+T+L]** after pruning K ingredients corresponding to K highest leaves in the ingredient tree. The results are averaged over 10 runs of 1k random recipes each.

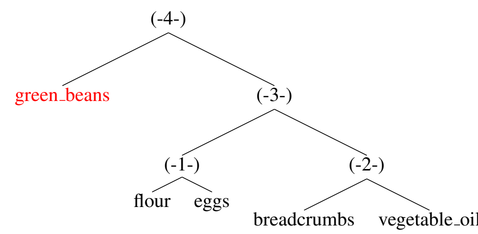


Figure 2: Ingredient tree of "Deep fried green beans". The order of flattened leaves from left to right is the same order they appear in the original ingredient list. The labels of intermediate nodes indicate the order of word combinations by Tree-LSTM, the higher value means its children are combined later. In this example, "flour" and "eggs" are combined first, and "green_beans" is embedded last into the final ingredient embedding.

Ingredient Substitution

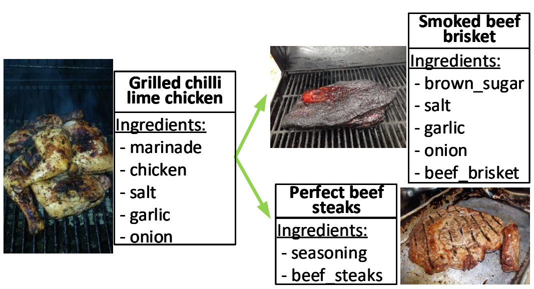
The above observation that Tree-LSTM can put more importance upon main ingredients leads to one interesting exploration: if the main ingredient of a recipe were replaced with another, would the embedding of the new recipe reflects this change, e.g., by retrieving real recipes containing this new ingredient? Inspired by earlier research on this task (Shidochi et al. 2009; Tsukuda et al. 2010; Yokoi et al.

	To Beef		To Apple		To Pork		To Fish		
Methods	**R-2-I**	**R-2-R**	**R-2-I**	**R-2-R**	**R-2-I**	**R-2-R**	**R-2-I**	**R-2-R**	**Med. Rank**
pic2rec (Salvador et al. 2017)	17.3	**22.5**	5.9	9.0	10.6	**18.9**	2.7	2.9	10.0
ACME (Wang et al. 2019)	18.6	18.9	5.4	6.3	8.2	8.9	3.3	3.1	10.0
S+L+L	29.6	16.1	10.3	8.4	**15.9**	6.3	**8.6**	3.5	5.5
S+T+T	29.9	7.3	9.3	7.2	13.9	4.1	5.9	2.6	9.5
G+L+L (Chen et al. 2018)	28.5	16.2	9.2	7.8	15.2	5.8	7.1	3.1	8.0
G+T+L	29.2	16.2	9.9	9.2	15.3	7.1	7.5	3.6	6.0
G+T+T	28.2	17.6	11.4	9.4	13.9	7.1	6.9	3.1	6.5
T+L+L	29.8	20.0	12.0	10.4	14.5	7.0	6.0	3.5	5.0
T+L+T	29.6	20.9	11.3	11.3	15.0	7.4	7.4	4.1	4.0
T+T+L	**31.0**	21.1	11.7	9.7	13.8	7.6	7.0	**4.2**	3.5
T+T+T	30.0	20.3	**12.2**	**11.7**	15.4	7.8	6.5	4.0	**2.5**

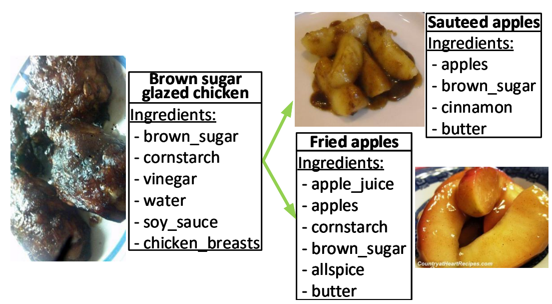
Table 4: Substitution from "chicken" to other ingredients. The values shown are Success Rate (SR) (higher is better). R-2-I and R-2-R indicate novel recipe-to-image and novel recipe-to-recipe retrievals, respectively. The last column shows the median ranks of all models in terms of SR across all experiments (lower is better). Best results are marked in bold.

	To Chicken		To Beef		To Apple		To Fish		
Methods	**R-2-I**	**R-2-R**	**R-2-I**	**R-2-R**	**R-2-I**	**R-2-R**	**R-2-I**	**R-2-R**	**Med. Rank**
pic2rec (Salvador et al. 2017)	41.2	**47.4**	25.6	27.6	6.8	13.7	3.2	4.2	11.0
ACME (Wang et al. 2019)	30.0	29.8	30.7	**28.4**	7.1	9.7	2.5	3.7	10.0
S+L+L	47.2	23.0	39.6	17.7	19.6	21.3	**9.0**	6.2	5.0
S+T+T	47.7	17.0	41.0	12.8	13.4	18.0	7.4	4.4	9.0
G+L+L (Chen et al. 2018)	47.9	23.6	40.8	17.2	14.6	17.5	7.0	5.4	5.0
G+T+L	47.8	24.6	39.5	18.6	14.4	20.6	8.5	6.4	5.0
G+T+T	46.4	24.8	39.5	21.1	17.5	20.3	7.8	5.5	5.0
T+L+L	**49.7**	29.9	42.9	21.1	18.1	20.3	6.9	5.3	**3.0**
T+L+T	44.8	29.2	40.9	22.7	17.8	22.8	8.6	6.5	4.0
T+T+L	47.6	28.4	42.0	22.1	**20.3**	19.4	8.4	**7.5**	6.0
T+T+T	45.7	27.7	**43.9**	23.0	16.9	**23.1**	6.0	6.2	4.0

Table 5: Substitution from "pork" to other ingredients. Best results are marked in bold.



(a) Modifying grilled chicken to grilled beef



(b) Modifying grilled chicken to fried apple

Figure 3: Ingredient substitution and recipe retrieval. "Chicken" in each recipe is replaced with "beef" or "apple" and carry out retrievals on both image and text embeddings. In each sub-figure: top-right and bottom-right show top-1 image retrieval and top-1 recipe text retrieval, respectively.

2015; Cordier et al. 2013), we carry out the substitutions by directly replacing the ingredient token in the input recipe with another. However, unlike prior endeavors which rely on standard text ranking, in this work we utilize deep models to extract the embeddings that can be used to retrieve *both* images and recipes. We propose a new metric to compare substitution performance of different models, namely *success rate* (SR), defined as the percentage of recipes containing ingredient A that were successfully identified as containing ingredient B by retrieval on a dataset, after replacing A with B in these recipes. Moreover, we report results where the original tree structures of the recipes are retained when inferring the new embeddings. There are marginal changes in performance when we let the text encoders infer new hierarchies from the modified recipes, suggesting that the new structures are similar to the originals in most cases.

We select recipes containing "chicken" and "pork" where they are identified as main ingredient by the Tree-LSTM ingredient encoder, and replacing them with one of the following ingredients where applicable: "chicken", "beef", "pork", "fish" and "apple". These ingredients are recognized as top ingredients by Tree-LSTM encoder. The results are shown in Tab. 4 and 5.

In the experiments converting chicken-based recipes to beef- and pork-based dishes, we see two contradicting results between image retrievals and text retrievals. In the R-2-I task, our proposed models outperform pic2rec and ACME by $\sim$50%, however, in the R-2-R task, pic2rec and ACME perform better, especially in the case of "pork" conversion. We observe that in the case of "pork" recipes, the titles usually do not contain the word "pork", but title contributes 33% of information to the final recipe embedding. Thus, the novel embeddings encoded by our models do not move to the correct manifold of "pork" recipes. On the other hand, pic2rec and ACME models do not include titles in the embeddings, hence they can match the novel recipes with real recipes better. It also suggests that the image embeddings are influenced more by ingredients in the corresponding recipes and their titles, hence our models perform better in the R-2-I task. It is also commonly observed that the meal images often expose the main ingredients. This explains why our proposed models perform better in more unrealistic substitutions to "fish" and particularly "apple", as these substitutions may generate nonsensical recipes, however, these ingredients are more effectively emphasized by using the tree hierarchies, thus the final embeddings are moved towards the respective manifolds of the substituting ingredients. The overall median ranks indicate that our proposed models, which use TreeLSTM ingredient encoder, are generally the better performers. Similar conclusions can be deduced from Tab. 5. These results suggest that **[T+T+L]** and **[T+T+T]** perform consistently well across different ingredient substitution experiments. These results also show that using Tree-LSTM as sentence and instruction encoders does not really have an effect on boosting successful substitution rates.

Fig. 3 demonstrates replacing "chicken" with "beef" and "apple". Replacing "chicken" with "beef" in the original recipe, "grilled chicken", in Fig. 3a, will match with real "grilled beef" recipes. In Fig. 3b, replacing "chicken" will retrieve "fried apple". This suggests that the cooking methods of the original recipes are preserved in the modified embeddings. In the next section, we will investigate whether the Tree-LSTM sentence encoder can capture the cooking methods, i.e., emphasize the importance of action words.

Action Word Extraction

Previous sections have demonstrated that it is possible to discover main ingredients with unsupervised training in our cross-modal retrieval framework. However, a cooking recipe not only consists of ingredients, but also describes a series of actions applied to ingredients in order to prepare the meal. Thus, it is equally important to emphasize these key actions to understand the recipe. An RNN model encodes a sentence sequentially, hence it is unable to specifically focus on a word arbitrarily positioned in the sentence. This problem can be partially remedied by applying the attention mechanism (Bahdanau, Cho, and Bengio 2015; Vaswani et al. 2017), however, Sec. demonstrates lack of correlation with importance of words. Tree-LSTM, on the other hand, provides a natural representation of sentence structure.

	Validation Set		Test Set	
Models	Verb Count	%	Verb Count	%
G+T+L	**229,042**	**78.44**	**227,840**	**78.20**
T+T+L	224,709	76.96	224,550	77.07
S+T+T	185,877	63.66	185,599	63.70
G+T+T	164,436	56.32	163,432	56.09
T+T+T	139,972	47.94	139,499	47.88

Table 6: Number of action words as the lowest leaves and their percentage over the number of sentence trees.

We investigate different recipe encoders trained with our cross-modal retrieval objective, in which sentences are encoded using Tree-LSTM, while the ingredient and instruction encoder sub-networks vary. Tree structures of all sentences in the validation and test sets are thoroughly analyzed. Based on the intuition that the leaf closest to the root of the tree might be the most important word, and that a sentence in a cooking recipe typically has one action word - a verb, we collect all leaf nodes closest to the root across all sentence trees, and count how many of them are verbs appearing in the WordNet database.

The results in Tab. 6 demonstrate that two models using Tree-LSTM to encode sentences and LSTM to encode the whole instructions are able to emphasize on the main action words in more than 76% of the number of sentences. **[G+T+L]** is marginally better than **[T+T+L]**. It can be explained that when Tree-LSTM is used for ingredient encoder, the recipe encoding network learns to focus more on ingredients, thus the importance of instructions is somewhat subsided. It is also noticeable that performance of models using Tree-LSTM to encode the whole instruction significantly declines. This is perhaps because sentences in a recipe are usually written in chronological order, hence learning instruction-level Tree-LSTM actually is detrimental to the ability to encode action words. Examples of inferred sentence trees are included in the supplementary materials.

Conclusion

In this paper, we present a novel cross-modal recipe retrieval framework that learns to jointly align the latent representations of images and texts. We particularly provide in-depth analysis of different architectures of the recipe encoder through a series of experiments. By using Tree-LSTM to model the hierarchical relationships between ingredients, and between words in instructional sentences, it is possible to capture more meaningful semantics from the recipe descriptions, such as main ingredient, cooking actions, thus also gaining better recipe adaptation capability and improving cross-modal retrieval performance as demonstrated by variants of our proposed framework, especially the **[T+T+L]** model which performs consistently well across different experimental tasks. In the future, we would like to jointly model the relationships between entities of the visual and textual modalities.

References

Bahdanau, D.; Cho, K.; and Bengio, Y. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *International Conferenece on Learning Representation*.

Bowman, S. R.; Gauthier, J.; Rastogi, A.; Gupta, R.; Manning, C. D.; and Potts, C. 2016. A Fast Unified Model for Parsing and Sentence Understanding. *arXiv preprint arXiv:1603.06021v3* .

Carvalho, M.; Cadène, R.; Picard, D.; Soulier, L.; Thome, N.; and Cord, M. 2018. Cross-Modal Retrieval in the Cooking Context: Learning Semantic Text-Image Embeddings. *Conference on Research and Development in Information Retrieval, SIGIR 2018* (18): 35–44. doi:10.1145/3209978.3210036.

Chen, J.-J.; Ngo, C.-W.; Feng, F.-L.; and Chua, T.-S. 2018. Deep Understanding of Cooking Procedure for Cross-modal Recipe Retrieval. In *2018 ACM Multimedia Conference on Multimedia Conference*, 1020–1028. ACM.

Cho, K.; Van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; and Bengio, Y. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, 1724–1734. Association for Computational Linguistics (ACL).

Choi, J.; Min Yoo, K.; and Lee, S.-g. 2017. Learning to Compose Task-Specific Tree Structures. In *In Association for the Advancement of Artificial Intelligence*. URL www.aaai.org.

Cordier, A.; Dufour-Lassier, V.; Lieber, J.; Nauer, E.; Badra, F.; Cojan, J.; Gaillard, E.; Infante-Blanco, L.; Molli, P.; Napoli, A.; and Skaf-Molli, H. 2013. TAABLE: A case-based system for personalized cooking. *Successful Case-based Reasoning Applications-2* 121–162.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Helmy, T.; Al-Nazer, A.; Al-Bukhitan, S.; and Iqbal, A. 2015. Health, food and user's profile ontologies for personalized information retrieval. *Procedia Computer Science* 52: 1071–1076.

Hochreiter, S.; and Schmidhuber, J. 1997. Long Short-Term Memory. *Neural Computation* 9(8): 1735–1780. ISSN 08997667. doi:10.1162/neco.1997.9.8.1735.

Marín, J.; Biswas, A.; Ofli, F.; Hynes, N.; Salvador, A.; Aytar, Y.; Weber, I.; and Torralba, A. 2019. Recipe1M + : A Dataset for Learning Cross-Modal Embeddings for Cooking Recipes and Food Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence* .

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, 3111–3119.

Min, W.; Jiang, S.; Liu, L.; Rui, Y.; and Jain, R. 2019. A Survey on Food Computing. *ACM Comput. Surv.* 52(5). ISSN 0360-0300. doi:10.1145/3329168. URL https://doi.org/10.1145/3329168.

Salvador, A.; Hynes, N.; Aytar, Y.; Marin, J.; Ofli, F.; Weber, I.; and Torralba, A. 2017. Learning cross-modal embeddings for cooking recipes and food images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 3020–3028.

Shidochi, Y.; Takahashi, T.; Ide, I.; and Murase, H. 2009. Finding replaceable materials in cooking recipe texts considering characteristic cooking actions. In *Proc. ACM Multimedia 2009 Workshop on Multimedia for Cooking and Eating Activities (CEA'09)*, 9–14.

Tai, K. S.; Socher, R.; and Manning, C. D. 2015. Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 1556–1566. Beijing, China: Association for Computational Linguistics. doi:10.3115/v1/P15-1150. URL https://www.aclweb.org/anthology/P15-1150.

Teng, C.-Y.; Lin, Y.-R.; and Adamic, L. A. 2012. Recipe recommendation using ingredient networks. In *Proceedings of the 4th Annual ACM Web Science Conference*, 298–307. ACM.

Tsukuda, K.; Yamamoto, T.; Nakamura, S.; and Tanaka, K. 2010. Plus one or minus one: A method to browse from an object to another object by adding or deleting an element. In *Proc. 21st Int. Conf. on Database and Expert Systems Applications*, 258–266.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention Is All You Need. In *Advances in neural information processing systems*.

Wang, H.; Sahoo, D.; Liu, C.; Lim, E.-p.; and Hoi, S. C. 2019. 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*, 11572–11581.

Yokoi, S.; Doman, K.; Hirayama, T.; Ide, I.; Deguchi, D.; and Murase, H. 2015. Typicality analysis of the combination of ingredients in a cooking recipe for assisting the arrangement of ingredients. In *Seventh Int. Workshop on Multimedia for Cooking and Eating Activities (CEA'15)*.

Zaheer, M.; Kottur, S.; Ravanbakhsh, S.; Poczos, B.; Salakhutdinov, R. R.; and Smola, A. J. 2017. Deep Sets. In Guyon, I.; Luxburg, U. V.; Bengio, S.; Wallach, H.; Fergus, R.; Vishwanathan, S.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, 3391–3401. Curran Associates, Inc. URL http://papers.nips.cc/paper/6931-deep-sets.pdf.

Zhu, X.; Sobihani, P.; and Guo, H. 2015. Long Short-Term Memory Over Recursive Structures. In Bach, F.; and Blei, D., eds., *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, 1604–1612. Lille, France: PMLR. URL http://proceedings.mlr.press/v37/zhub15.html.