Cross-modal Recipe Retrieval with Rich Food Attributes

Jing-jing Chen  
City University of Hong Kong  
Hong Kong  
jingjchen9-c@my.cityu.edu.hk

Chong-Wah Ngo  
City University of Hong Kong  
Hong Kong  
cswngo@cityu.edu.hk

Tat-Seng Chua  
National University of Singapore  
Singapore  
chuats@comp.nus.edu.sg

ABSTRACT

Food is rich of visible (e.g., colour, shape) and procedural (e.g., cutting, cooking) attributes. Proper leveraging of these attributes, particularly the interplay among ingredients, cutting and cooking methods, for health-related applications has not been previously explored. This paper investigates cross-modal retrieval of recipes, specifically to retrieve a text-based recipe given a food picture as query. As similar ingredient composition can end up with wildly different dishes depending on the cooking and cutting procedures, the difficulty of retrieval originates from fine-grained recognition of rich attributes from pictures. With a multi-task deep learning model, this paper provides insights on the feasibility of predicting ingredient, cutting and cooking attributes for food recognition and recipe retrieval. In addition, localization of ingredient regions is also possible even when region-level training examples are not provided. Experiment results validate the merit of rich attributes when comparing to the recently proposed ingredient-only retrieval techniques.

KEYWORDS

Recipe retrieval; cross-modal retrieval; ingredient recognition; cooking and cutting recognition

1 INTRODUCTION

Tracking food-intake has long been an open problem in nutritional science [1] [2] [30]. Manual tracking is known to be cumbersome and error-prone as reported in [12]. A viable solution adopted by commercial apps is by automatic identification of food content from picture and derivation of nutrition facts as well as calories by matching to food composition table [16][32]. The current technologies, nevertheless, are limited to barcode identification of raw ingredients and visual recognition to a limited number of standardized cooked cafeteria foods from chained restaurants [3] [4]. These techniques are difficult to be generalized for recognition of dishes prepared in the wild, which can exhibit diverse visual appearances. Even if the same set of ingredients is used, the outlook of dishes can appear differently depending on the cooking and cutting methods, as shown in Figure 1.

In general, derivation of nutrition content from food picture requires fine-grained localization and recognition of ingredients.

For prepared foods where ingredients are mixed and stirred or scrambled (see Figure 1), uncovering the underlying ingredients is already highly challenging, not to mention the estimation of ingredient quantities. An indirect way of deriving nutrients from picture, as employed in [3] [6], is by retrieval of cooking recipes or restaurant menus such that the ingredient quantities or calories information can be extracted as references for further analysis. Such workflow, from picture-to-text translation, is typically a cross-modal retrieval [11] problem.

In literature, the food identification problem is formulated as either a search [3] or recognition [8] [15] [16] [25] [31] problem. For the former, sample food pictures of a recipe or menu are indexed. During online retrieval, users snap a food picture as query and the problem becomes the matching of food samples for recipe identification. For the latter, food pictures are crawled for training of classifiers for dishes recognition. Except [6] [24] [41] which performs recognition at the ingredient level, the majority of approaches operate at food-level and suffer from scalability problem. In practice, these approaches are only capable of recognizing a limited number of food categories known during training time. Ingredient-level recognition is in principle scalable for treating ingredients as attributes in enabling zero-shot retrieval [6]. Specifically, even when a food category is unseen in advance, it is still possible to retrieve by matching with the predicted ingredients.

Similar in spirit as [6], this paper performs fine-grained ingredient recognition for searching of cooking recipes. Particularly, we address the problem in real-world that there are many different varieties of dishes cooked with the same ingredients. Recognizing using ingredients alone is inherently insufficient to retrieve recipes. Figure 1 shows three different examples of dishes that use the same ingredients. Basically, different cutting methods result in
different shape appearances, for example in Figure 1(a), the shredding or slicing of green pepper and potato alters the outlook of dishes. Similarly, different cooking methods, such as “pan-fry” and “stir-fry” shown in Figure 1(b), can change the colour and texture appearance of the dishes. When the methods for both cutting and cooking are different, the appearance can be wildly diverse as the fish dishes shown in Figure 1(c). Hence, we argue that effective food recognition generally requires knowledge of cutting and cooking attributes beyond ingredients. From the viewpoint of “healthy eating”, knowing cooking attributes also provide helpful clues for nutrition analysis. For example, “boiling” can wash away water-soluble vitamins, and people with diabetes should limit the intake of “deep-fry” food.

Technically, existing approaches tackle the aforementioned problem by directly embedding the cutting and cooking methods into ingredient labels. For example in [6], there are 13 labels used for the “egg” ingredient to characterize different ways of cutting and cooking egg. Such labeling strategy, although practically useful for ingredient recognition, is difficult to scale up due to the exponential number of attribute combinations for ingredient, cutting and cooking methods. In this paper, we use as many as 1,276 ingredients, 8 cutting and 36 cooking methods. By brute force combination of these attributes, there could be close to 0.4 million labels; this is beyond the capacity that a deep neural network can be trained, unless with sufficient training samples and machines.

This paper proposes the retrieval of recipes by rich food attributes, i.e., ingredients, cutting and cooking methods. The three attributes are recognized by a three-way deep architecture, which are learnt in a multi-task manner. Specifically, each task aims to predict a particular type of attribute, while shared visual features are learnt to minimize the overall prediction error of the three attributes. A peculiar challenge is that the prediction of cutting and cooking attributes requires the knowledge of ingredient locations. In other words, ingredient regions have to be localized as the recognition of attributes should happen at the image region level. This basically makes the design of deep neural architecture fundamentally different from [6]. In our proposal, the localization of ingredients is learnt in an unsupervised manner without the requirement of region-level training information. In addition, to keep track of the localized correspondence among the three types of attributes, a new pooling technique is tailored to combine the results of prediction from different tasks.

The main contribution of this paper is on the introduction of rich food attributes for cross-modal recipe retrieval, which addresses the limitation of existing literature on how the ingredients are labeled and utilized for search. To the best of our knowledge, there is no research effort yet on the prediction and leveraging of all the three attributes for food recognition and recipe retrieval.

2 RELATED WORK

While foods are rich in visual (e.g., ingredient composition) and procedural (e.g., cutting and cooking methods) attributes, the current literature leverages these attributes in a very different way. Visual attributes are utilized for retrieval and classification of food pictures [6] [41] [42], while procedural attributes are employed for construction of cooking workflow [33] [35] and ingredient network [9] for text-based recipe recommendation. There is no work that investigates the interplay between visual and procedural attributes for cross-modal recipe retrieval. There are few works that examine the hardware sensors [13] and capture video for recognition of cutting and cooking activities in kitchen [26], but are out of scope of this paper. Contextually related attributes, such as venue and GPS, cuisine (e.g., Chinese, American) and course (e.g., breakfast, afternoon tea), have also been exploited in [4] [24] [34]. These attributes are not always available but are generally effective for boosting recognition performance. Although not being considered in this paper, these attributes can be readily engineered to narrow the search scope of recipe retrieval.

Most of the existing works employ deep features extracted directly from neural networks for image-level food categorization [36]. As reported in [5] [17], deep features perform significantly better than hand-crafted features such as SIFT [20] and HOG [10] on benchmark food datasets such as UEC Food-100 [17] and Food-101[5]. Similar conclusion is also recently reported by [6] in VIREO Food-172 dataset. Furthermore, the performance is also related to the type of neural network being employed. As studied in [6] [19] [21], deeper networks such as VGG [28], GoogLeNet [29] and ResNet [14] tend to generate better food features than AlexNet.

Ingredient recognition receives much few attentions than food categorization [6] [24] [37] [41]. The problem is more challenging as ingredients are small in size and can exhibit larger variances in appearance. Early studies include PFD (pairwise local feature distribution) [37], which defines 8 types of ingredients for pixel labeling. PFD explores spatial relationship between ingredient labels to generate high dimensional features for food recognition. Although powerful, PDF is not scalable as feature dimension is exponential to the number of ingredients, and can grow dramatically to tens of thousands of dimensions with only 8 ingredients. In [5] [18], instead of explicitly defining ingredient labels for recognition, discriminative features corresponding to ingredients are mined for recognition. For example, DPM (deformable part-based model) and STF (semantic texon forest) are proposed in [18] for detection of ingredient regions. Random forest is employed in [5] to cluster super-pixels of food pictures for inferring prominent features for recognition. These approaches, while capable of showing excellent performance for standardized cooked food such as desserts and fast food, require tuning of hand-crafted parameters for performance optimization. More importantly, for dishes with wild composition of ingredients, learning of discriminative features is practically difficult to achieve with shallow methods such as feature clustering [5].

Deep-based ingredient recognition has also been recently investigated. In [6], a VGG multi-task learning framework is proposed for simultaneous recognition of food categories and ingredient labels. Similarly in [41] which further incorporates recognition of cooking attributes into multi-task learning. In [24], multimodal deep Boltzmann machine is applied for ingredient recognition and food image retrieval. Another work is [42] which exploits the rich relationships among ingredient, food category and restaurant through bipartite-graph representation. Segmentation of food into ingredients is also explored in [23], by convolutional network and CRF (conditional random field). Nevertheless, this approach requires location labels for training, which are difficult or even impossible to be obtained for prepared foods with ingredients being cut and mixed or stirred.
The more recent works [7][27] learn a joint multi-modal embedding space for transformation of visual and text features extracted from pictures and recipes respectively. The joint space in [7] is learnt through attention model to highlight ingredient regions, and is demonstrated to be effective in quantifying the similarity between food images and ingredients extracted from recipes. In [27], both ingredients and cooking instructions are extracted for learning joint space together with global image features. Different from [7], however, no attention model is considered during learning.

Among the aforementioned works [6] [7] [24] [23] [41], none of them indeed explore the interplay of ingredient, cutting and cooking attributes for recipe retrieval. As localization of ingredients is required for cutting and cooking attributes, the multi-task model proposed in this paper is different from [6] [41] which does not exploit region information. In [41] for example, cooking attributes are assumed to be globally associated with dishes and not locally with ingredients. Although this assumption simplifies the design of deep architecture, the model cannot be employed for retrieving recipes where ingredients are individually cooked before composing into dishes. Our proposed work is also different from [23] as labeling of ingredient regions for model training is not required. Furthermore, our model not only locates ingredient regions as in [23], but can also label the regions with rich attributes, which are infeasible to be learnt in [23]. Finally and more importantly, this paper aims to investigate the decoupling of ingredient labels from cutting and cooking attributes, which hinders scalability of food recognition. The problem is yet to be explored by any prior works for a more practical way of attribute recognition and recipe retrieval.

3 RICH ATTRIBUTE LEARNING

Framework. Figure 2 presents an overview of the proposed framework. Given a food picture I, a pyramid of multi-resolution images is generated and input to a deep convolutional network (DCNN). The corresponding feature maps are extracted from DCNN and subsequently transformed into embedded features for the prediction of ingredients, cutting and cooking methods. The attributes are pooled across image regions and scales before being matched against text recipes for retrieval.

Feature embedding layer. Our DCNN architecture uses the VGG network [28]. The Pool5 feature maps, which correspond to the last convolution layer of DCNN and retain the spatial information of the original image, are extracted from VGG for feature embedding. The Pool5 feature is divided into m x m grids, where each grid is represented by a vector of 512 dimensions. The value of m varies depending on the image size. For an image of size 448 x 448, m = 14 and each grid corresponds to a receptive field of 32 x 32 resolution. We denote fi as the Pool5 feature and its grids or regions as fi, where i ∈ [0, m x m]. Each region fi is transformed to an embedding feature as follows:

\[ v_i = \tanh(W_f i + b) \]  (1)

where \( v_i \in \mathbb{R}^d \) is the transformed vector in d dimensional space, \( W_f \in \mathbb{R}^{d \times 512} \) is the learnt transformation matrix and \( b \in \mathbb{R}^d \) is the bias term.

Region-wise multi-task classification. The predictions of attributes are learnt simultaneously in a multi-task manner, taking advantage of their joint relationships in modeling the diverse dish appearances. Specifically, while each prediction is regarded as a separate task, they share the same feature embedding layer which is learnt to optimize the predictions of three different attribute types. Notice that the learning is conducted region-wise and the prediction is made directly on each grid of an image.

As each grid depicts a small region of the original image, a reasonable assumption being adopted is that there is one dominant ingredient per region. Furthermore, an ingredient is assumed to undergo at most one cutting procedure and one dominant cooking procedure. There are few cases where this assumption is violated, such as the example, shown in Figure 2, where the egg and tomato are stir-fried followed by boiled. In this case, we consider “stir-fry” as the dominant cooking method for altering the appearance of egg and tomato. Therefore, the predictions of ingredient, cutting and cooking labels for each region are regarded as single-label classification.

The activation function being applied is \( \text{softmax} \) for getting the probability distributions of ingredient, cutting and cooking labels, denoted as \( \hat{p}_{\text{ingr},i} \in \mathbb{R}^c, \hat{p}_{\text{cut},i} \in \mathbb{R}^c \) and \( \hat{p}_{\text{cook},i} \in \mathbb{R}^k \) respectively for \( i^{th} \) region as follows:

\[ \hat{p}_{\text{ingr},i} = \text{softmax}(W_{\text{ingr}}v_i + b_{\text{ingr}}). \]  (2)

\[ \hat{p}_{\text{cut},i} = \text{softmax}(W_{\text{cut}}v_i + b_{\text{cut}}). \]  (3)
where \( t, c \) and \( k \) denote the number of ingredients, cutting and cooking labels respectively. The learnt transformation matrices are \( \mathbf{W}_{\text{ingredient}} \in \mathbb{R}^{c \times d}, \mathbf{W}_{\text{cut}} \in \mathbb{R}^{c \times d} \) and \( \mathbf{W}_{\text{cook}} \in \mathbb{R}^{k \times d} \), and similarly for the bias terms \( b_{\text{ingredient}} \in \mathbb{R}^c, b_{\text{cut}} \in \mathbb{R}^c \) and \( b_{\text{cook}} \in \mathbb{R}^k \).

Region-level dependency pooling. Since each region is associated with the probability distributions of three different attributes, a straightforward way to obtain the image-level labels is by an independent max-pooling over the three distributions across regions. Nevertheless, such simple scheme overlooks the joint relationship among the three attributes. For example, a region that contributes to the response of an ingredient does not guarantee that its cooking and cutting attributes will be counted during max pooling. As a result of independent pooling, the three image-level attributes could be inconsistent which could confuse and complicate the learning of network parameters.

We propose dependency pooling by first performing max pooling of ingredient labels across regions, followed by pooling of cutting and cooking attributes only from the regions that contribute most to the image-level ingredient labels. Let \( \hat{P}_{\text{ingredient}, l} \) be the probability distribution of ingredients for image \( l \). The response of an ingredient indexed by \( j \) element is obtained as follows:

\[
\hat{P}_{\text{ingredient}, l}(j) = \max\{\hat{P}_{\text{ingredient}, l}(j)\}_{i=1}^{m^2}
\]

where \( m^2 \) is the total number of image grids. For each ingredient, the \( i^{th} \) region which contributes most to the response will be tracked. Subsequently, the \( \hat{P}_{\text{cut}, l} \) and \( \hat{P}_{\text{cook}, l} \) of every region will be pooled or aggregated to form two matrices, denoted as \( \hat{P}_{\text{cut}, l} \in \mathbb{R}^{c \times 1} \) and \( \hat{P}_{\text{cook}, l} \in \mathbb{R}^{k \times 1} \) respectively. In other words, each \( \hat{P}_{\text{ingredient}, l}(j) \) is indexed to vectors \( \hat{P}_{\text{cut}, l}(j) \) and \( \hat{P}_{\text{cook}, l}(j) \), corresponding to the prediction of cutting and cooking attributes for ingredient \( j \).

\textbf{Loss function.} The loss function is defined for each attribute type and then linearly averaged summed as follows:

\[
L = \frac{1}{N} \sum_{n=1}^{N} (L_1 + L_2 + L_3)
\]

where \( L_1 \) is a loss function referring to the prediction of ingredient \( (L_1) \), cutting \( (L_2) \) or cooking \( (L_3) \), and \( N \) is the total number of training images. Note that, as \( L_1 \) is calculated at the image-level, ingredient recognition is a multi-label classification problem. The loss functions \( L_2 \) and \( L_3 \), on the other hand, are calculated on the basis of every ingredient. As each ingredient is associated to one cooking and one cutting method, \( L_2 \) and \( L_3 \) are all single-label classification problems. The loss function used for \( L_1, L_2 \) and \( L_3 \) is cross-entropy.

We denote \( \hat{P}_{\text{ingredient}, l} \in \{0, 1\}^c \) as the ground-truth ingredients for a food picture \( I_n \), represented by a binary vector whose elements are either 1 or 0 indicating the presence or absence of a particular ingredient. The loss function \( L_1 \) is defined as

\[
L_1 = \sum_{j=1}^{c} \hat{P}_{\text{ingredient}, l}(j) \log(\hat{P}_{\text{ingredient}, l}(j)) + (1 - \hat{P}_{\text{ingredient}, l}(j)) \log(1 - \hat{P}_{\text{ingredient}, l}(j))
\]

Furthermore, we denote \( a = [a, \hat{P}_{\text{ingredient}, l}(a)]_{u=1}^{c} \) as the ingredients visible in \( I_n \). For each ingredient \( a \in a \), let \( \hat{P}_{\text{cut}, l}(a) \in \{0, 1\}^c \) as its ground-truth cutting label and \( \hat{P}_{\text{cook}, l}(a) \in \{0, 1\}^k \) as its ground-truth cooking label. The loss functions \( L_2 \) and \( L_3 \) are defined as follows:

\[
L_2 = \sum_{a \in a} \sum_{v=1}^{c} \hat{P}_{\text{cut}, l}(a, v) \log(\hat{P}_{\text{cut}, l}(a, v)) + (1 - \hat{P}_{\text{cut}, l}(a, v)) \log(1 - \hat{P}_{\text{cut}, l}(a, v))
\]

\[
L_3 = \sum_{a \in a} \sum_{\mu=1}^{k} \hat{P}_{\text{cook}, l}(a, \mu) \log(\hat{P}_{\text{cook}, l}(a, \mu)) + (1 - \hat{P}_{\text{cook}, l}(a, \mu)) \log(1 - \hat{P}_{\text{cook}, l}(a, \mu))
\]

During training, the errors accumulated from three tasks are backpropagated through the network till the embedding layer. The involved parameters, including the shared \( (e.g., \mathbf{W}_I) \) and dedicated \( (e.g., \mathbf{W}_{\text{ingredient}}, \mathbf{W}_{\text{cut}}, \mathbf{W}_{\text{cook}}) \) parameters, will be updated correspondingly to simultaneously optimize the recognition performances of three tasks.

\textbf{Multi-scale recognition.} The size of ingredient varies depending on factors such as the cutting methods and camera-to-dish distance. Using a fixed resolution of grids cannot handle the change in scale. The problem is tackled by the generation of pyramid images in multiple resolutions. In this way, the receptive field of a grid can spatially extend to a larger scope depending on the resolution of input image. For example, for an image of size 448 × 448, each grid in the pool5 feature map corresponds to a receptive field of 32 × 32 image region. By reducing the size of image to a resolution of 224 × 224, the receptive field extends to the spatial size of 64 × 64 in the original image before resizing.

The consideration of multi-scale recognition will only introduce minor change to the network architecture. Except region-level pooling that involves consolidation of attribute predictions from multiple scales, the updating of parameters remains the same throughout the learning procedure. By denote \( \hat{P}_{\text{ingredient}, l}^{l} \) as the probability distribution of ingredients at scale \( l \), max pooling is conducted across different regions and scales as follows:

\[
\hat{P}_{\text{ingredient}, l}(j) = \max\{\max\{\hat{P}_{\text{ingredient}, l}^{l}(j)\}_{l=1}^{L}\}_{l=1}^{L}
\]

During pooling, the regions which contribute most to the response of a particular ingredient are tracked for aggregation of \( \hat{P}_{\text{cut}, l} \) and \( \hat{P}_{\text{cook}, l} \). Basically, the multi-scale design ensures that an ingredient and its cooking and cutting attributes can be adaptively pooled from a region in a particular scale that exhibits the highest possible prediction confidence.

\section{Cross-Modal Recipe Retrieval}

Using the proposed deep architecture, a query picture \( Q \) is represented by \( \hat{P}_{\text{ingredient}, Q} \in \mathbb{R}^c, \hat{P}_{\text{cut}, Q} \in \mathbb{R}^{c \times 1} \) and \( \hat{P}_{\text{cook}, Q} \in \mathbb{R}^{k \times 1} \), corresponding to the probability distributions of ingredients, cooking and cutting methods respectively. On the other hand, a text-based recipe \( R \) is represented by a set of attribute triplets, \( \{c, cut_x, cook_x \mid x \in O\} \), where \( O \) is the set of ingredients extracted from \( R \). \( cut_x \) denotes the cutting attribute for ingredient \( x \).
and similarly for \( \text{cook}_x \). The similarity between \( Q \) and \( R \) is defined as

\[
\text{Sim}(Q, R) = \frac{\sum_{x \in \mathcal{O}} P_{\text{ingre},Q}(x) + \lambda (P_{\text{cut},Q}(x, \text{cut}_x) + P_{\text{cook},Q}(x, \text{cook}_x))}{|\mathcal{O}|}.
\]

The parameter \( \lambda \) denotes the importance of cutting and cooking attributes, where its value is learnt to be 0.2 on the validation set. The similarity score is normalized in order not to erroneously favouring recipes that contain an excessive number of ingredients.

5 EXPERIMENTS

5.1 Dataset

There are a number of public food datasets, such as UEC Food-100 [22], Food-101 [5], VIREO Food-172 [6], but none of them includes the cooking and cutting attributes of ingredients. We therefore constructed a new food dataset, by crawling 47,882 recipes along with images associated with these recipes from the "Go Cooking" website. The dataset is composed of mostly Chinese food, ranging from regular dishes, snacks and desserts to Chinese-style western food. We recruited 10 homemakers who have cooking experience for attribute labeling. A total of 1,276 ingredients compiled from the recipes are labeled. The lists of cutting and cooking attributes are shown in Table 1, which are respectively compiled from "Go cooking" and "meishijie" websites.

During manual annotation, a homemaker was provided with recipes along with food images. The homemaker was first instructed to read a recipe so as to understand the cooking procedure. The list of ingredients extracted from the recipe was then prompted to homemaker for selection of visible ingredients in a food picture. Ingredients missing from the recipe can also be manually input. As some ingredients are named together with cooking and cutting methods in the recipe, we manually reverted these ingredients to their original names before prompting to homemakers for selection. For example, the ingredient "seasoned beef slice" was reverted to "beef". The homemakers were also requested to also label the cutting and cooking methods of each ingredient from the provided lists as shown in Table 1. Input of "null" label for cutting and cooking method is allowed when ingredient did not subject to any cutting and cooking methods. To guarantee the quality of annotation, we sampled a subset of labels for manual checking and provided verbal feedbacks to homemakers whenever labelling is inaccurate or imprecise. The whole annotation procedure ended in about two weeks. On average, each food picture contains three visible ingredients. Furthermore, each ingredient has on average 121 positive training samples.

5.2 Experimental setting

Instead of learning a deep model from scratch, we extend the VGG architecture trained on VIREO Food-172 [6]. The model was reported to exhibit fairly good recognition performance on 353 ingredients. We extend the model to 1,276 ingredients and plug in two additional tasks for the recognition of cooking and cutting

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|}
\hline
\textbf{Cut} & \textbf{Batonnet} & \textbf{Bruitene dice} \\
\hline
\textbf{Mince} & \textbf{Roll cut} & \textbf{Shreds} \\
\hline
\textbf{Slice} & \textbf{Smash} & \textbf{} \\
\hline
\textbf{Baking} & \textbf{Bake in pan} & \textbf{Braising} \\
\hline
\textbf{Braising} & \textbf{Bake stewing} & \textbf{Boiling} \\
\hline
\textbf{Braising with starch} & \textbf{Clay pot cooking} & \textbf{Casserole} \\
\hline
\textbf{Cover and simmer} & \textbf{Deep frying} & \textbf{Dressing} \\
\hline
\textbf{Extreme-heat stir-fry} & \textbf{Food drying} & \textbf{Flash-frying} \\
\hline
\textbf{Grated} & \textbf{Griddle cooking} & \textbf{Grilling} \\
\hline
\textbf{Gradual simmering} & \textbf{Hot candied} & \textbf{Jellying} \\
\hline
\textbf{Juicing} & \textbf{Most stir-fry} & \textbf{Microwaving} \\
\hline
\textbf{Pan-frying} & \textbf{Pickling} & \textbf{Quick-frying} \\
\hline
\textbf{Quick boiling} & \textbf{Roasting} & \textbf{Scalding} \\
\hline
\textbf{Seasoned with soy sauce} & \textbf{Steaming} & \textbf{Sugar dipped} \\
\hline
\textbf{Slow red cooking} & \textbf{Stir-frying} & \textbf{Smoking} \\
\hline
\end{tabular}
\end{table}

labels. The dataset is split into three sets: 80% for training, 10% for validation and 10% for testing.

During training, the dimension of embedding feature (Equation 1) is set to \( d = 300 \) and validated to be effective on the validation set. Two-level of pyramid images, respectively at resolutions of \( 448 \times 448 \) and \( 224 \times 224 \), are used for multi-scale recognition. The model is trained using stochastic gradient descent with the momentum set as 0.9. The initial learning rate is set to be 0.1 and the batch size is 50. The learning rate decays after every 3,000 iterations. Finally, as each food picture only has a small number of ingredients out of the available 1,276 ingredients, the ground-truth vector \( P_{\text{ingre},i,t} \) is very sparse. As a result, negative sampling is adopted by randomly selecting 10% of negative samples for training.

5.3 Recognition performance

As ingredients involve multiple labels, a threshold is required to gate the selection of labels. The threshold is set to 0.5, following the common practice for deep learning based multi-label recognition. In other words, only labels whose prediction scores surpass the threshold are considered as being recognized. Table 2 shows the performance of our proposed multi-task model. As recognition of cutting and cooking methods belong to single-label classification, only top-1 accuracy is shown. As ingredients are cut into small pieces in most of the dishes, the use of higher resolution images achieves better recall in ingredient recognition and accuracy in prediction of cooking and cutting methods. On the other hand, lower resolution (224 \( \times \) 224), which has larger receptive field and hence can consider more surrounding context of ingredients, obtains higher precision. By combining both scales, the best performances are attained for all the three attributes. Figure 3 shows some examples of attribute predictions. In general, the results are satisfactory except in several cases which are summarized as follows.

First, the model is limited in recognizing ingredients with similar shape and color. For example, in Figure 3(e), "fish ball" is incorrectly predicted as "lotus seed". Both ingredients appear somewhat similar when cooked in soup. More typical examples are indeed different kind of meats (e.g., "duck" and "chicken") which could have similar texture pattern when being cooked by certain methods like "stewing" or "braising". In Figure 3(f) for example, "pork rib" is wrongly recognized as "beef" and "chicken". Despite these failure

1\(^{http://www.xiachufang.com}\)

2\(^{http://so.meisha.cc/index.php?q=cooking}\)
Table 2: Food attribute prediction at different scales

<table>
<thead>
<tr>
<th>Scale</th>
<th>Ingredient</th>
<th>Cutting</th>
<th>Cooking</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Recall</td>
<td>Precision</td>
<td>F1</td>
</tr>
<tr>
<td>Single (448 × 448)</td>
<td>0.369</td>
<td>0.233</td>
<td>0.286</td>
</tr>
<tr>
<td>Single (224 × 224)</td>
<td>0.260</td>
<td>0.252</td>
<td>0.256</td>
</tr>
<tr>
<td>Multi (448 × 448 + 224 × 224)</td>
<td>0.391</td>
<td>0.240</td>
<td>0.297</td>
</tr>
</tbody>
</table>

Table 3: Ingredient recognition: multi versus single task learning

<table>
<thead>
<tr>
<th>Task</th>
<th>Ingredient recognition</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Recall</td>
</tr>
<tr>
<td>Ingredient (FC) [6]</td>
<td>0.121</td>
</tr>
<tr>
<td>Ingredient (Region)</td>
<td>0.358</td>
</tr>
<tr>
<td>Ingredient + cutting</td>
<td>0.381</td>
</tr>
<tr>
<td>Ingredient + cooking</td>
<td>0.373</td>
</tr>
<tr>
<td>Ingredient + cutting + cooking</td>
<td>0.391</td>
</tr>
</tbody>
</table>

Figure 3: Examples of attribute prediction. False positives are marked in red. The sign “-” indicates no cutting or cooking method is applied.

Examples, our model can always correctly recognize the ingredients with unique shape, such as the "chicken wing" in Figure 3(c). Second, our model always fails to predict ingredients covered under sauces such as that shown in Figure 3(d). Such examples are not easy to be recognized even by human. Third, as we consider only two levels of pyramid images, our model cannot deal with pictures with close-up view of dishes, such as that in Figure 3(f). In this example, both “carrot” and “white radish” are identified incorrectly due to limited scope of receptive field. Fourth, certain cutting attributes are relatively hard to be distinguished, for example "roll cut" and "large dice", where the former (latter) cuts ingredients into pieces with two angled sides (blocks). The cutting effect is not easily observed especially when ingredients are mixed or occluded with each other. Comparatively, cutting methods such as “shred”, "mince" and "slice" which result in distinguishable shapes of ingredients can always be predicted correctly. Similarly for cooking attributes, where the effect of "gradual simmering", “cover and simmer” and “bake stewing” are not visually distinguishable. Finally, the correlation between ingredient and cooking attributes also affects the prediction accuracy. For example, the outlooks of certain ingredients such as "carrot", "black fungus" and "peas" change a little despite undergone different cooking methods. As a result, while these ingredients are relatively easy to recognize, their cooking attributes are often predicted wrongly. On the other hand, there are ingredients which associate only with few cooking methods, for example "flour" and "baking". Multi-task model always yields high prediction accuracy for these attributes. We also compare our model with the deep architecture in [6], which reported state-of-the-arts results for ingredient recognition on UEC Food-100 and VIREO Food-172 datasets. The architecture has two pathways, one for food categorization and the other for ingredient recognition.

Figure 4: The ingredient “flour” appears wildly diverse under different cooking methods but still can be recognized by our model.

We take away the pathway for food categorization and fine tune the network with 1,276 ingredients in our dataset. Additionally, as [6] learns embedding features from FC (fully-connected) layer of DCNN, we also compare to a variant of the model which learns features from Pool5 convolutional layer and performs region-level prediction and pooling for ingredient recognition.

Table 3 lists the performance of three different approaches. Basically our proposed model, which is composed of three pathways, outperforms the other two single task models. Besides, learning features from convolution layer (region) performs significantly better than FC layer. As ingredient is generally small in size, region-level recognition is superior to image-level. We attribute the success of our model to the fact that, by having cutting and cooking information, our model has better capability in dealing with diverse appearances of ingredient. Cutting attribute contributes to larger improvement than cooking for enjoying higher prediction accuracy as shown in Table 3. Figure 4 shows examples where the ingredient “flour” appears to be varied in different dishes but can still be correctly recognized by our model and not by [6], either with FC or region. To verify that the improvement is not by chance, we conduct significance test to compare our architecture with both single task models. Using the source code provided by TRECVID, the test is performed by partial randomization with 100,000 number of iterations for F1 measure. At the significance level of 0.05, our architecture is significantly better than other models with p-values close to 0, which rejects the null hypothesis that the improvement is by chance.

<http://www-nlpir.nist.gov/projects/t01v/trecvid.tools/randomization.testing>
We compile a list of 1,000 images from the test set as queries for cross-modal recipe retrieval. Each query has only one ground-truth recipe. These queries are searched against a dataset composed of 4,985 recipes. Among them, 1,278 of recipes share exactly the same set of ingredients with at least another one recipe, despite belonging to different dishes. We purposely select the queries such that there are 412 of them whose ground-truth recipes are a subset of 1,278 recipes. Furthermore, in order to verify the advantage of cutting and cooking attributes, only images whose F1>0.3 in ingredient recognition for all the three approaches shown in Table 3 are selected as queries. The following metrics are employed for performance evaluation.

- Mean reciprocal rank (MRR): MRR measures the reciprocal of rank position where the ground truth recipe is returned, averaged over all the queries. This measure assesses the ability of the system to return the correct recipe at the top of the ranking. The value of MRR is within the range of [0, 1]. A higher score indicates a better performance.
- Recall at Top-K (R@K): R@K computes the fraction of times that a correct recipe is found within the top-K retrieved candidates. R@K provides an intuitive sense of how quickly the best recipe can be located by investigating a subset of the retrieved items. As MRR, a higher score also indicates a better performance.

Table 4 shows the incremental improvement in retrieval when cutting and cooking methods are incorporated. Cutting attributes basically introduce higher degree of improvement than cooking methods across all the measures. This is mainly because the prediction of cutting attributes is more accurate. The best result is attained when all the three attributes are jointly considered. We conduct significance test to verify the improvement is not by chance. Using partial randomization, the test suggests that there is a significant difference between using all three attributes and using ingredient attribute only at the level of 0.05.

Figure 5 shows three examples of recipe retrieval. In Figure 5(a), the two recipes ranked at the top contain the same set of ingredients. By correctly recognizing the cooking methods of ingredients, our model successfully ranks the ground-truth recipe at top. Figure 5(b) shows an example where one of the key ingredients ("pork") and its cooking attribute are predicted wrongly. However, by correctly recognizing cutting attributes of key ingredients, our model is still able to rank the ground-truth recipe higher than the recipe with the same ingredients but different cutting and cooking methods. Figure 5(c) shows an example where our model cannot distinguish recipes with similar ingredients where cooking attributes are predicted wrongly.

### 5.4 Recipe retrieval

Next, we compare our architecture with three other approaches. The first is multi-task model in [6], where we take the pathway for ingredient recognition and fine tune with 1,274 ingredient labels in our dataset. We term the method as "single-task". As [6] embeds cooking and cutting attributes directly into the ingredient labels, the second approach takes the same strategy. We combine the three different attributes by brute-force, resulting in 20,736 labels with training examples. By further removing labels with less than 10 examples, we only manage to retain 1,345 labels, which are used to fine tune the ingredient pathway in [6]. We term the second method as "single-task (BF)". In the experiment, we implement two versions of these approaches, by extracting features from fully-connected (FC) and convolutional (region) layers of DCNN. The third approach is based on the attention model recently proposed in [7], which learns a joint feature representation between visual and text by stacked attention model [38]. We train the model using the same training and validation sets as our proposed architecture. Table 5 lists the result of comparison. As expected, the brute-force combination of different attributes leads to better performance than the "single-task" and attention model which use ingredient-only attributes. With additional attributes, "single-task (BF)" manages to distinguish ingredients undergone drastic appearance changes because of cutting and cooking methods. Nevertheless, due to the lack of training examples, the performance of "single-task (BF)" is not as good as our proposed architecture. For example, the ingredient "wild rice stem" has limited examples and is being applied to different cutting methods. In such circumstance, "single-task (BF)" will perform poorly as compared to our model. On the other hand, when sufficient training examples are available, for example, different of "egg" that are cut and cook under various ways, "single-task (BF)" indeed exhibits better performance.

### 5.5 Response map

A by-product of our multi-task model is the capability of locating ingredients. We visualize the result in a response map, which is formed by converting the prediction score of an ingredient on an...
image grid into pixel intensity value. Figure 6 shows the response maps of three ingredients for a query image. Generally speaking, the better the result of localization is, the higher the prediction accuracy. Figure 7 visualizes more results for different kinds of ingredient composition. For dishes with well-separated ingredients such as Figure 7(a), our model often achieves high prediction as well as localization accuracy despite different ingredients being cooked or cut with different methods. Localization becomes challenging when ingredients are mixed as in Figure 7(b), but our model still manages to show reasonable result. As our model considers only region-level information, the ingredient labels at different locations of dishes could be inconsistent. For example in Figure 7(c), "garlic sprout" is sometimes predicted as "green onion" which has similar visual appearance. Similarly for the ingredients "lamp" and "pork". Some of the consistency can indeed be removed by noise filtering through techniques such as region merging and common sense rules that certain kinds of meats are seldom cooked together to reduce the risk of bacteria cross-contamination. Finally, although multi-dish segmentation is not considered, our model is able to locate ingredients for different dishes in a picture as shown in Figure 7(d).

6 CONCLUSIONS

We have presented a multi-task deep learning architecture for addressing the challenge of recognizing ingredients under different cutting and cooking methods. Particularly, we shed light that, instead of coupling all three attributes to generate exponential number of ingredient labels for model training, learning the attributes in multi-task manner can generate predictions feasible for recipe retrieval. The model suffers less from the need of a large amount of learning samples and is easier to be trained with a smaller number of network parameters. Experimental results basically confirm the merit of using cutting and cooking attributes in recognizing diverse appearance of ingredients. More importantly, leveraging three attributes altogether enables more effective way of ranking recipes that share same or similar set of ingredients. Despite these encouraging results, ingredients with similar visual outlook and ingredients that are covered under sauces or being occluded remain difficult to be identified, which affect the retrieval effectiveness.

There are a number of issues not being investigated in this paper and require further studies. First, the interplay of the three attributes with non-visible ingredients (e.g., salt, sugar) and visible sauces (e.g., ketchup, soy sauce) are not considered, where recognizing both of them can practically help in distinguishing similar recipes (e.g., "sweet and sour spare ribs" and "braised spare ribs in brown sauce"). Second, automatic extraction of attributes from recipes by itself is also a challenging natural language processing problem, in view that cooking procedures are written in free-form in most recipes. How would automated extraction of attribute affect the result presented in this paper, is yet to be examined. Third, the retrieval performance can potentially be further improved if cutting and cooking attributes are weighted differently during similarity measure (Equation-11). Recipe-to-image search is also feasible under the current retrieval modal although is not empirically evaluated in this paper. Finally, this paper considers only region-level identification of ingredients, which results in inconsistent predictions throughout different regions of a dish. Proper utilization of context-level information, such as common sense in food preparation, could be helpful in getting rid of some false predictions.

7 ACKNOWLEDGEMENT

The work described in this paper was supported in part by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (CityU 11203517) and NExT++ project supported by National Research Foundation, Prime Minister’s Office, Singapore under its IRC@SG Funding Initiative.
REFERENCES


