# Revamping Cross-Modal Recipe Retrieval with Hierarchical Transformers and Self-supervised Learning

Amaia Salvador  Erhan Gundogdu  Loris Bazzani  Michael Donoser
Amazon
{asalvada, eggundog, bazzanil, donoserm}@amazon.com

## Abstract

*Cross-modal recipe retrieval has recently gained substantial attention due to the importance of food in people's lives, as well as the availability of vast amounts of digital cooking recipes and food images to train machine learning models. In this work, we revisit existing approaches for cross-modal recipe retrieval and propose a simplified end-to-end model based on well established and high performing encoders for text and images. We introduce a hierarchical recipe Transformer which attentively encodes individual recipe components (titles, ingredients and instructions). Further, we propose a self-supervised loss function computed on top of pairs of individual recipe components, which is able to leverage semantic relationships within recipes, and enables training using both image-recipe and recipe-only samples. We conduct a thorough analysis and ablation studies to validate our design choices. As a result, our proposed method achieves state-of-the-art performance in the cross-modal recipe retrieval task on the Recipe1M dataset. We make code and models publicly available*[1].

## 1. Introduction

Food is one of the most fundamental and important elements for humans, given its connection to health, culture, personal experience, and sense of community. With the development of the Internet and the rise of social networks, we witnessed a substantial surge in digital recipes that are shared online by users. Designing powerful tools to navigate such large amounts of data can support individuals in their cooking activities to enhance their experience with food, and has thus become an attractive research field [30]. Often times, digital recipes come along with companion content such as photos, videos, nutritional information, user reviews, and comments. The availability of such rich large scale food datasets has opened the doors for new applications in the context of food computing [37, 34, 24], one of the most prevalent ones being cross-modal recipe retrieval, where the goal is to design systems that are capable of finding relevant cooking recipes given a user submitted food image. Approaching this challenge requires developing models in the intersection of natural language processing and computer vision, as well as being able to deal with unstructured, noisy, and incomplete data.

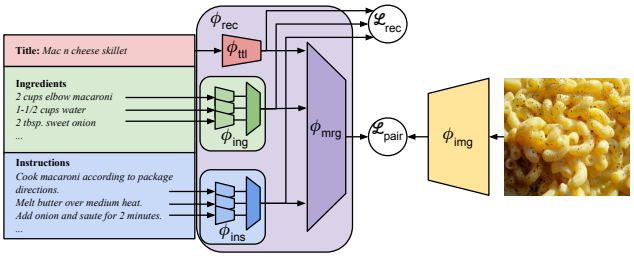



Figure 1: **Model overview.** Our method is composed of three distinct parts: the image encoder $\phi_{img}$, the recipe encoder $\phi_{rec}$, and the training objectives $\mathcal{L}_{pair}$ and $\mathcal{L}_{rec}$.

In this work, we focus on learning joint representations for textual and visual modalities in the context of food images and cooking recipes. Recent works on the task of cross-modal recipe retrieval [37, 3, 7, 42, 50] have introduced approaches for learning embeddings for recipes and images, which are projected into a joint embedding space that is optimised using contrastive or triplet loss functions. Advances were made by proposing complex models and loss functions, such as cross-modal attention [14], adversarial networks [50, 42], the use of auxiliary semantic losses [37, 3, 42, 50, 42], multi-stage training [37, 13], and reconstruction losses [14]. These works are either complementary or orthogonal to each other while bringing certain disadvantages such as glueing independent models [37, 13], which needs extra care, relying on a pre-trained text representations [37, 3, 7, 42, 50, 13] and complex training pipelines involving adversarial losses [50, 42]. In contrast to previous works, we revisit ideas in the context of cross-modal recipe retrieval and propose a *simplified* end-to-end joint embedding learning framework that is plain, effective,

[1] https://github.com/amzn/image-to-recipe-transformers

and straightforward to train. Figure 1 shows an overview of our proposed approach.

Unlike previous works using LSTMs to encode recipe text [44, 5, 6, 37, 3], we introduce a recipe encoder based on Transformers [40], with the goal of obtaining strong representation for recipe inputs (i.e. titles, ingredients, and instructions) in a bottom-up fashion (see Figure. 1, left). Following recent works in the context of text summarization [48, 26], we leverage the structured nature of cooking recipes with hierarchical Transformers, which encode lists of ingredients and instructions by extracting sentence-level embeddings as intermediate representations, while learning relationships within each textual modality. Our experiments show superior performance of Transformer-based recipe encoders with respect to their LSTM-based counterparts that are commonly used for cross-modal recipe retrieval.

Training joint embedding models requires cross-modal paired data, i.e., each image must be associated to its corresponding text. In the context of cross-modal recipe retrieval, this involves quadruplet samples of pictures, title, ingredients, and instructions. Such a strong requirement is often not fulfilled when dealing with large scale datasets curated from the Web, such as Recipe1M [37]. Due to the unstructured nature of recipes that are available online, Recipe1M largely consists of text-only samples, which are often either ignored or only used for pretraining text embeddings. In this work, we propose a new self-supervised triplet loss computed between embeddings of different recipe components, which is optimised jointly and end-to-end with the main triplet loss computed on paired image-recipe embeddings (see $\mathcal{L}_{rec}$ in Figure 1). The addition of this new loss allows us to use both paired and text-only data during training, which in turn improves retrieval results. Further, thanks to this loss, embeddings from different recipe components are aligned with one another, which allows us to recover (or *hallucinate*) them when they are missing at test time.

Our method encodes recipes and images using simple yet powerful model components, and is optimised using both paired and unpaired data thanks to the new self-supervised recipe loss. Our approach achieves state-of-the-art results on Recipe1M, one of the most prevalent dataset in the community. We perform an ablation study to quantify the contribution of each of the design choices, which range from how we represent recipes, the impact of our proposed self-supervised loss, and a state-of-the-art comparison.

The *contributions* of this work are the following. **(1)** We propose a recipe encoder based on hierarchical Transformers that significantly outperforms its LSTM-based counterparts on the cross-modal recipe retrieval task. **(2)** We introduce a self-supervised loss term that allows our model to learn from text-only samples by exploiting relationships between recipe components. **(3)** We perform extensive experimentation and ablation studies to validate our design choices (i.e. recipe encoders, image encoders, impact of each recipe component). **(4)** As a product of our analysis, we propose a simple, yet effective model for cross-modal recipe retrieval which achieves state-of-the-art performance on Recipe1M, with a medR of 3.0, and Recall@1 of 33.5, improving the performance of the best performing model [13] by 1.0 and 3.5 points, respectively.

## 2. Related Work

### 2.1. Visual Food Understanding

The computer vision community has made significant progress on food recognition since the introduction of new datasets, such as Food-101 [2] and ISIA Food-500 [31]). Most works focus on food image classification [25, 34, 32, 28, 11, 23], where the task is to the determine the category of the food image. Other works study different tasks such as estimating ingredient quantities of a food dish [8, 24], predicting calories [27], or predicting ingredients in a multi-label classification fashion [5, 6]. Since the release of multi-modal datasets such as Recipe1M [37], new tasks in the context of leveraging images and textual recipes have emerged. Several works proposed solutions that use image-recipe paired data for cross-modal recipe retrieval [42, 7, 37, 3, 43], recipe generation [36, 41, 4, 1, 33], image generation from a recipe [49, 35] and question answering [46]. Our paper tackles the task of cross-modal recipe retrieval between food images and recipe text. In the next section, we focus on the specific contributions of previous works addressing this task, highlighting their differences with respect to our proposed solution.

### 2.2. Cross-Modal Recipe Retrieval

Learning cross-modal embeddings for images and text is currently an active research area [19, 15, 18]. Methods designed for this task usually involve encoding images and text using pre-trained convolutional image recognition models and LSTM [17] or Transformer [40] text encoders.

In contrast to short descriptions from captioning datasets [10, 47, 38], cooking recipes are long and structured textual documents which are non-trivial to encode. Due to the structured nature of recipes, previous works proposed to encode each recipe component independently, using late fusion strategies to merge them into a fixed-length recipe embedding. Most works [37, 3, 42, 50, 13] do so by first pretraining text representations (e.g. word2vec [29] for words, skip-thoughts [21] for sentences), training the joint embedding using these representations as fixed inputs. In contrast to these works, our approach resembles the works of [7, 14] in that we use the raw recipe text directly as input, training the representations end-to-end.

In the literature of cross-modal recipe retrieval, there is still no consensus with regards to how to best utilise the

recipe information, and which encoders to use to obtain representations for each component. First, most early works [37, 3, 42, 50, 42] treat recipe ingredients as single words, which requires an additional pre-processing step to extract ingredient names from raw text (e.g. extracting *salt* from *1 tbsp. of salt*). Only a few works [7, 14] have removed the need for this pre-processing step by using raw ingredient text as input. Second, it is worth noting that most works ignore the recipe title when encoding the recipe, and only use it to optimise auxiliary losses [37, 3, 42, 50, 42]. Third, when it comes to architectural choices, LSTM encoders are the choice of most previous works in the literature, using single LSTMs to encode sentences (e.g. titles, categorical ingredient lists), and hierarchical LSTMs to encode sequences of sentences (e.g. raw ingredient lists or cooking instructions). In contrast with the aforementioned works, we propose to standardise the process of encoding recipes by (1) using the recipe in its complete form (i.e. titles, ingredients, and instructions are all inputs to our model), and (2) removing the need of pre-processing and pre-training stages by using text in its raw form as input. Further, we propose to use Transformer-based text encoders, which we empirically demonstrate to outperform LSTMs.

While triplet losses are often used to train such cross-modal models, most works proposed auxiliary losses that are optimised together with the main training objective. Examples of common auxiliary losses include cross-entropy or contrastive losses using pseudo-categories extracted from titles as ground truth [37, 3, 42, 50, 42, 14] and adversarial losses on top of reconstructed inputs [50, 42, 14], which come with an increase of complexity during training. Other works have also proposed architectural changes such as incorporating self- and cross-modal-attention mechanisms [14]. In our work, we err on the side of simplicity by using encoder architectures that are ubiquitous in the literature (namely, vanilla image encoders such as ResNets and Transformers), optimised with triplet losses.

Finally, it is worth noting that while Recipe1M is a multimodal dataset, only 33% of the recipes contain images. Previous works [37, 3, 7, 50, 42] only make use of the paired samples to optimise the joint image-recipe space, while ignoring the text-only samples entirely [14, 43], or only using them to pre-train text embeddings [37, 3, 7, 42, 50, 13]. In contrast, we introduce a novel self-supervised loss that is computed on top of the recipe representations of our model, which allows us to train with additional recipes that are not paired to any image in an end-to-end fashion.

# 3. Learning Image-Recipe Embeddings

We train a joint neural embedding on data samples from a dataset of size $N$: $\{(x_I^n, x_R^n)\}_{n=1}^N$. Each $n^{th}$ sample is composed of an RGB image $x_I$ depicting a food dish, and its corresponding recipe $x_R = (r_{ttl}, r_{ing}, r_{ins})$, composed of a title $r_{ttl}$, a list of ingredients $r_{ing}$, and a list of instructions $r_{ins}$. In case that the recipe sample is not paired to any image, $x_I^j$ is not available for the $j^{th}$ sample and only $x_R^j$ is used during training. Figure 1 shows an overview of our method. Images and recipes are encoded with $\phi_{img}$ and $\phi_{rec}$, respectively, and embedded into to the same space through $\mathcal{L}_{pair}$. We incorporate a self-supervised recipe loss $\mathcal{L}_{rec}$ acting on pairs of individual recipe components. We describe each of these components below.

## 3.1. Image Encoder $\phi_{img}$

The purpose of the image encoder is to learn a mapping function $e_I^n = \phi_{img}(x_I^n)$ which projects the input image $x_I^n$ into the joint image-recipe embedding space. We use ResNet-50 [16] initialised with pre-trained ImageNet [22] weights as the image encoder. We take the output of the last layer before the classifier and project it to the joint embedding space with a single linear layer to obtain an output of dimension $D = 1024$. We also experiment with ResNeXt [45] based models, as well as the recently introduced Vision Transformer (ViT) encoder [12][2].

## 3.2. Recipe Encoder $\phi_{rec}$

The objective of the recipe encoder is to learn a mapping function $e_R^n = \phi_{rec}(x_R^n)$ which projects the input recipe $x_R^n$ into the joint embedding space to be directly compared with the image $e_I^n$. Previous works in the literature have encoded recipes using LSTM-based encoders for recipe components, which are either pre-trained on self-supervised tasks [37, 3, 42, 50], or optimised end-to-end [14, 43] with an objective function computed for paired image-recipe data. Similarly, our model uses a specialised encoder for each of the recipe components (namely, title, ingredients, and instructions). We use three separate encoders to process sentences from the title, ingredients and instructions. In contrast to previous works, we propose to use Transformer-based encoders for recipes as opposed to LSTMs, given their ubiquitous usage and superior performance in natural language processing tasks.

**Sentence Representation.** Given a sequence of word tokens $s = (w^0, ..., w^K)$, we seek to obtain a fixed length representation that encodes it in a meaningful way for the task of cross-modal recipe retrieval. The title consists of a single sentence, i.e. $r_{ttl} = s_{ttl}$, while instructions and ingredients are list of sentences, i.e. $r_{ing} = (s_{ing}^0, ..., s_{ing}^M)$ and $r_{ins} = (s_{ins}^0, ..., s_{ins}^O)$. We take advantage of the training flexibility of Transformers [40] for encoding sentences in the recipe. We encode each sentence with Transformer network of 2 layers of dimension $D = 512$, each with 4 attention heads, using a learned positional embeddings in the first layer. The representation for the sen-

[2]We use the ViT-B/16 pretrained model from https://rwightman.github.io/pytorch-image-models/.

tence is the average of the outputs of the Transformer encoder at the last layer. Figure 2a shows a schematic of our Transformer-based sentence encoder, $\mathrm{TR}(\cdot, \theta)$, where $\theta$ are the model parameters, which are different for each recipe component. Thus, we extract title embeddings as: $e_{ttl} = \phi_{ttl}(r_{ttl}) = \mathrm{TR}(r_{ttl}, \theta_{ttl})$.

**Hierarchical Representation.** Both ingredients and instructions are provided as lists of multiple sentences. To account for these differences and exploit the input structure, we propose a hierarchical Transformer encoder, named $\mathrm{HTR}(\cdot, \theta)$, which we will use to encode inputs composed of sequences of sentences (see Figure 2b). Given a list of sentences of length $M$, a first Transformer model $TR_{L=1}$ is used to obtain $M$ fixed-sized embeddings, one for every sentence in the list. Then, we add a second Transformer $TR_{L=2}$ with the same architecture (2 layers, 4 heads, $D = 512$) but different parameters, which receives the list of sentence-level embeddings as input, and outputs a single embedding for the list of sentences. We use this architecture to encode both ingredients and instructions separately, using different sets of learnable parameters: $e_{ing} = \phi_{ing}(r_{ing}) = \mathrm{HTR}(r_{ing}, \theta_{ing})$, and $e_{ins} = \phi_{ins}(r_{ins}) = \mathrm{HTR}(r_{ins}, \theta_{ins})$.

The recipe embedding $e_R$ is computed with a final projection layer on top of concatenated features from the different recipe components: $e_R = \phi_{mrg}([e_{ing}; e_{ins}; e_{ttl}])$, where $\phi_{mrg}$ is a single learnable linear layer of $D = 1024$, and $[\cdot;\cdot;]$ denotes embedding concatenation[3].

In order to compare with previous works [37, 3, 42, 50, 42], we also experimented with LSTM [17] versions of our proposed recipe encoder with the same output dimensionality $D = 512$, keeping the last hidden state as the representation for the sequence.

### 3.3. Supervised Loss for Paired Data, $\mathcal{L}_{pair}$

Inspired by the success of triplet hinge-loss objective for recipe retrieval [3, 7, 42, 50], we define the main component of our loss function as follows:

$$\mathcal{L}_{cos}(a, p, n) = \max(0, c(a, n) - c(a, p) + m) \quad (1)$$

where $a$, $p$, and $n$ refer to the anchor, positive, and negative samples, $c(\cdot)$ is the cosine similarity metric, and $m$ is the margin (empirically set to 0.3 for all triplet losses used in this work). In practice, we use the popular bi-directional triplet loss function [42] on feature sets $a$ and $b$:

$$\begin{aligned}\mathcal{L}'_{bi}(i, j) = \mathcal{L}_{cos}(a^{n=i}, b^{n=i}, b^{n=j}) \\ + \mathcal{L}_{cos}(b^{n=i}, a^{n=i}, a^{n=j})\end{aligned} \quad (2)$$

where $a^{n=i}$ and $b^{n=i}$ are positive to each other, and $b^{n=j}$ and $a^{n=j}$ are negative to $a^{n=i}$ and $b^{n=i}$, respectively. We use the notation $n = i$ to denote same-sample embeddings (e.g. recipe and image embeddings from the same sample in the dataset) and $n = j$ for embeddings from a different sample $j$. During training, for a batch of size $B$, the loss for every sample $i$ is the average of all losses considering all other samples in the batch as negatives.

$$\mathcal{L}_{bi}(a^{n=i}, b^{n=i}, b^{n\neq i}, a^{n\neq i}) = \frac{1}{B}\sum_{j=0}^{B} \mathcal{L}'_{bi}(i, j)\delta(i, j), \quad (3)$$

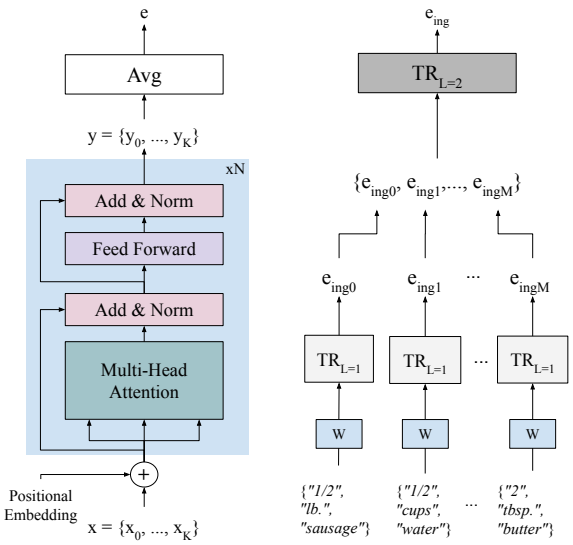
where $\delta(i, j) = 1$ if $i \neq j$ and 0 otherwise. In the case that we have paired image-recipe data, we define the following loss by setting $a$ and $b$ to correspond to the image and recipe embeddings, respectively:

$$\mathcal{L}_{pair} = \mathcal{L}_{bi}(e_I^{n=i}, e_R^{n=i}, e_R^{n\neq i}, e_I^{n\neq i}) \quad (4)$$

where $e_I = \phi_{img}(I)$, $e_R = \phi_{rec}(R)$ with $e_I, e_R \in \mathbb{R}^D$ are fixed sized representations extracted using the image and recipe encoders described in the previous sections.

### 3.4. Self-supervised Recipe Loss, $\mathcal{L}_{rec}$

In the presence of unpaired data or partially available information, it is not possible to optimise Eq. 4 directly.




(a) TR (b) HTR

Figure 2: **(a) Transformer Encoder, TR:** Given a recipe sentence, our model encodes it into a fixed length representation using the Transformer encoder. **(b) Hierarchical Transformer Encoder, HTR:** For sequences of sentences (i.e. ingredients, or instructions), we use a hierarchical model, where a first Transformer encodes each sentence separately into a fixed sized vector, and a second Transformer encodes them into a single representation.

[3]We also experimented with embedding average instead of concatenation, which gave slightly worse retrieval performance.

This is a rather common situation for noisy datasets collected from the Internet. In the case of Recipe1M, 66% of the dataset consists of samples that do not contain images, i.e. only include a textual recipe. In practice, this means that $e_I$ is missing for those samples, which is why most works in the literature simply ignore text-only samples to train the joint embeddings. However, many of these works [37, 3, 7, 50, 42] use recipe-only data to pre-train text representations, which are then used to encode text. While these works implicitly make use of all training samples, they do so in a suboptimal way, since such pre-trained representations are unchanged when training the joint cross-modal embeddings for retrieval.

In this paper, we propose a simple yet powerful strategy to relax the requirement of relying on paired data when training representations end-to-end for the task of cross-modal retrieval. Intuitively, while the individual components of a particular recipe (i.e. its title, ingredients, and instructions) provide complementary information, they still share strong semantic cues that can be used to obtain more robust and semantically consistent recipe embeddings. To that end, we constrain recipe embeddings so that intermediate representations of individual recipe components are close together when they belong to the same recipe, and far apart otherwise. For example, given the title representation of a recipe $e_{ttl}^{n=i}$ we define an objective function to make it closer to its corresponding ingredient representation $e_{ing}^{n=i}$ and farther from the representation of ingredients from other recipes $e_{ing}^{n\neq i}$. Formally, during training we incorporate a triplet loss term between title, ingredient and instruction embeddings that is defined as follows:

$$\mathcal{L}'_{rec}(a,b) = \mathcal{L}_{bi}(e_a^{n=i}, \hat{e}_{b\to a}^{n=i}, \hat{e}_{b\to a}^{n\neq i}, e_a^{n\neq i}) \tag{5}$$

where $a$ and $b$ can both take values among the three different recipe components (title, ingredient and instructions). For every pair of values for $a$ and $b$, the embedding feature $e_b$ is projected to another feature space as $\hat{e}_{b\to a}$ using a single linear layer $g_{b\to a}(e_b)$. Figure 3 shows the 6 different projection functions for all possible combinations of $a$ and $b$. Note that, similarly to previous works in the context of self-supervised learning [39] and learning from noisy data [9], we optimise the loss between $e_a$ and $\hat{e}_{b\to a}$, instead of between $e_a$ and $e_b$. The motivation for this design is to leverage the shared semantics between recipe components, while still keeping the unique information that each component brings (i.e. information that is present in the ingredients might be missing in the title). By adding a projection before computing the loss, we enforce embeddings to be similar but not the same, avoiding the trivial solution of making all embeddings equal.

We compute the loss above for all possible combinations of $a$ and $b$, and average the result:

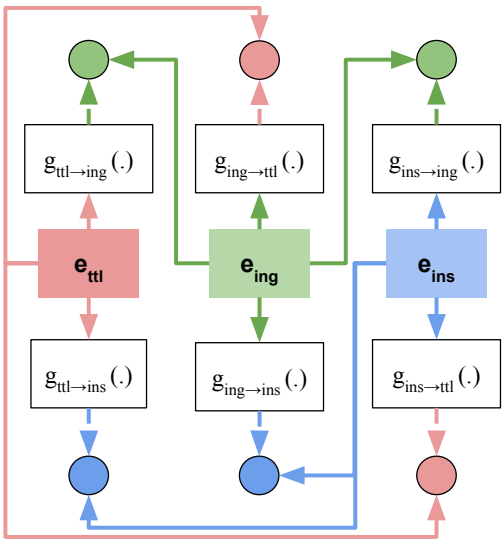



Figure 3: **Self-supervised recipe losses.** Coloured dots denote loss terms computed for each recipe component. Each component embedding, e.g. $e_{ttl}$, is optimised to be close to the projected embeddings of the other two recipe components, namely: $g_{ing\to ttl}(e_{ing})$, and $g_{ins\to ttl}(e_{ins})$.

$$\mathcal{L}_{rec} = \frac{1}{6}\sum_a \sum_b \mathcal{L}'_{rec}(a,b)\delta(a,b), \tag{6}$$

where $a, b \in \{ttl, ing, ins\}$. Figure 3 depicts the 6 different loss terms computed between all possible combinations of recipe components.

The final loss function is the composition of the paired loss and the recipe loss defined as: $\mathcal{L} = \alpha\mathcal{L}_{pair} + \beta\mathcal{L}_{rec}$, where both $\alpha$ and $\beta$ are set to 1.0 for paired samples, and $\alpha = 0.0$ and $\beta = 1.0$ for text-only samples.

## 4. Experiments

This section presents the experiments to validate the effectiveness of our proposed approach, including ablation studies, and comparison to previous works.

### 4.1. Implementation Details

**Dataset.** Following prior works, we use the Recipe1M [37] dataset to train and evaluate our models. We use the official dataset splits which contain 238.999, 51.119 and 51.303 image-recipe pairs for training, validation and testing, respectively. When we incorporate the self-supervised recipe loss from Section 3.4, we make use of the remaining part of the dataset that only contains recipes (no images), which adds 482.231 samples that we use for training.

**Metrics.** Following previous works, we measure retrieval performance with the median rank (medR) and Recall@{1, 5, 10} (referred to as R1, R5, and R10). on rank-

| | medR | R1 | R5 | R10 |
|---|---|---|---|---|
| LSTM + avg | 9.0 | 17.9 | 41.2 | 52.9 |
| H-LSTM | 7.0 | 19.8 | 44.8 | 57.2 |
| Transformer + avg | 7.0 | 20.2 | 45.2 | 57.3 |
| H-Transformer | **5.0** | **24.4** | **51.4** | **63.4** |

Table 1: **Comparison between recipe encoders**. Image-to-recipe retrieval results reported on the validation set of Recipe1M. Results reported on rankings of size $10k$.

| | medR | R1 | R5 | R10 |
|---|---|---|---|---|
| Ingredients only | 8.2 | 19.1 | 42.8 | 54.3 |
| Instructions only | 15.0 | 12.6 | 32.2 | 43.3 |
| Title only | 35.5 | 6.0 | 18.7 | 28.1 |
| Ingrs + Instrs | 6.0 | 22.4 | 48.3 | 60.4 |
| Title + Ingrs | 6.0 | 22.1 | 47.7 | 59.8 |
| Title + Instrs | 10.5 | 15.9 | 38.4 | 50.2 |
| Full Recipe | **5.0** | **24.4** | **51.4** | **63.4** |

Table 2: **Ablation studies of recipe components.** Image-to-recipe retrieval results reported on the validation set of Recipe1M. Results reported on rankings of size $10k$.

ings of size $N = \{1.000, 10.000\}$. We report average metrics on 10 groups of $N$ randomly chosen samples.

**Training details.** We train models with a batch size of 128 using the Adam [20] optimiser with a base learning rate of $10^{-4}$ for all layers. We use step-wise learning rate decay of 0.1 every 30 epochs and monitor validation $R@1$ every epoch, keeping the best model with respect to that metric for testing. Images are resized to 256 pixels in their shortest side, and cropped to $224 \times 224$ pixels. During training, we take a random crop and horizontally flip the image with 0.5 probability. At test time, we use center cropping. For experiments using text-only samples, we alternate mini-batches of paired and text-only data with a 1:1 ratio. In the case of recipe-only samples, we increase the batch size to 256 to take advantage of the lower GPU memory requirements when dropping the image encoder.

### 4.2. Recipe Encoders

We compare the proposed Transformer-based encoders from Section 3.2 with their corresponding LSTM variants. We also quantify the gain of employing hierarchical versions by comparing them with simple average pooling on top of the outputs of a single sentence encoder (either LSTM or Transformer). Table 1 reports the results for the task of image-to-recipe retrieval in the validation set of Recipe1M. Transformers outperform LSTMs in both the averaging and hierarchical settings (referred to as *+avg* and *H-*) by 4.6 and 2.3 R1 points, respectively. Further, the use of hierarchical encoders provides a boost in performance with respect to the averaging baseline both for Transformers and LSTMs (increase of 4.2 and 1.9 R1 points, respectively). Given its favourable performance, we adopt the H-Transformer in the rest of the experiments.

### 4.3. Ablation Study on Recipe Components

In this section, we aim at quantifying the importance of each of the recipe components. Table 2 reports image-to-recipe retrieval results for models trained and tested using different combinations of the recipe components. Results in the first three rows indicate that the ingredients are the most important component, as the model achieves a R1 of 19.1 when ingredients are used in isolation. In contrast, R1 drops to 12.6 and 6.0 when using only the instructions and the title, respectively. Further, results improve when combining pairs of recipe components (rows 4-6), showing that using ingredients and instructions achieves the best performance of all possible pairs (R1 of 22.4). Finally, the best performing model is the one using the full recipe (last row: R1 of 24.4), suggesting that all recipe components contribute to the retrieval performance.

### 4.4. Self-supervised Recipe Loss

With the goal of understanding the contribution of the self-supervised loss described in Section 3.4, we compare the performance of three model variants in the last three rows of Table 3: $\mathcal{L}_{pair}$ only uses the loss function for paired image-recipe data, $\mathcal{L}_{pair} + \mathcal{L}_{rec}$ adds the self-supervised loss considering only paired data, and $(\mathcal{L}_{pair} + \mathcal{L}_{rec})^{\diamond}$ is trained on both paired and recipe-only samples. The self-supervised learning approach $\mathcal{L}_{pair} + \mathcal{L}_{rec}$ improves performance with respect to $\mathcal{L}_{pair}$, while using the same amount of paired data (improvement of 0.5 R1 points on the image-to-recipe setting for rankings of size 10k). These results indicate that enforcing a similarity between pairs of recipe components helps to make representations more robust, leading to better performance even without extra training data. The last row of Table 3 shows the performance of $(\mathcal{L}_{pair} + \mathcal{L}_{rec})^{\diamond}$, which is trained with the addition of the self-supervised loss, optimised for both paired and recipe-only data. Significant improvements for image-to-recipe retrieval are obtained for both median rank and recall metrics with respect to $\mathcal{L}_{pair}$: medR decreases to 4.1 from 4.0 and R1 lifts up from 26.8 to 27.9. These results indicate that both the self-supervised loss term and the additional training data contribute to the performance improvement. We also quantify the contribution of the $g(\cdot)$ functions from Figure 3 by comparing to a baseline model in which they are replaced with identity functions. This model achieves slightly worse retrieval results with respect to $(\mathcal{L}_{pair} + \mathcal{L}_{rec})^{\diamond}$ (0.5 point decrease in terms of R1).

| | 1k | | | | | | | | 10k | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | image-to-recipe | | | | recipe-to-image | | | | image-to-recipe | | | | recipe-to-image | | | |
| | medR | R1 | R5 | R10 | medR | R1 | R5 | R10 | medR | R1 | R5 | R10 | medR | R1 | R5 | R10 |
| Salvador et al. [37] $^\diamond$ | 5.2 | 24.0 | 51.0 | 65.0 | 5.1 | 25.0 | 52.0 | 65.0 | 41.9 | - | - | - | 39.2 | - | - | - |
| Chen et al. [7] | 4.6 | 25.6 | 53.7 | 66.9 | 4.6 | 25.7 | 53.9 | 67.1 | 39.8 | 7.2 | 19.2 | 27.6 | 38.1 | 7.0 | 19.4 | 27.8 |
| Carvalho et al. [3] $^\diamond$ | 2.0 | 39.8 | 69.0 | 77.4 | 1.0 | 40.2 | 68.1 | 78.7 | 13.2 | 14.9 | 35.3 | 45.2 | 12.2 | 14.8 | 34.6 | 46.1 |
| R2GAN [50]$^\diamond$ | 2.0 | 39.1 | 71.0 | 81.7 | 2.0 | 40.6 | 72.6 | 83.3 | 13.9 | 13.5 | 33.5 | 44.9 | 12.6 | 14.2 | 35.0 | 46.8 |
| MCEN [14] | 2.0 | 48.2 | 75.8 | 83.6 | 1.9 | 48.4 | 76.1 | 83.7 | 7.2 | 20.3 | 43.3 | 54.4 | 6.6 | 21.4 | 44.3 | 55.2 |
| ACME [42]$^\diamond$ | 1.0 | 51.8 | 80.2 | 87.5 | 1.0 | 52.8 | 80.2 | 87.6 | 6.7 | 22.9 | 46.8 | 57.9 | 6.0 | 24.4 | 47.9 | 59.0 |
| SCAN [43] | 1.0 | 54.0 | 81.7 | 88.8 | 1.0 | 54.9 | 81.9 | 89.0 | 5.9 | 23.7 | 49.3 | 60.6 | 5.1 | 25.3 | 50.6 | 61.6 |
| DaC [13] | - | - | - | - | - | - | - | - | 5.9 | 24.4 | 49.4 | 60.5 | - | - | - | |
| DaC [13]$^\diamond$ | 1.0 | 55.9 | 82.4 | 88.7 | - | - | - | | 5.0 | 26.5 | 51.8 | 62.6 | - | - | - | - |
| Ours ($\mathcal{L}_{pair}$) | 1.0 | 58.3 | 86.2 | 91.8 | 1.0 | 59.6 | 86.1 | 92.2 | 4.1 | 26.8 | 54.7 | 66.5 | 4.0 | 27.6 | 55.1 | 66.8 |
| Ours ($\mathcal{L}_{pair} + \mathcal{L}_{rec}$) | 1.0 | 59.1 | 86.9 | 92.3 | 1.0 | 59.1 | 87.0 | 92.7 | 4.0 | 27.3 | 55.4 | 67.3 | 4.0 | 27.8 | 55.6 | 67.3 |
| Ours ($\mathcal{L}_{pair} + \mathcal{L}_{rec}$)$^\diamond$ | 1.0 | **60.0** | **87.6** | **92.9** | 1.0 | **60.3** | **87.6** | **93.2** | 4.0 | **27.9** | **56.4** | **68.1** | 4.0 | **28.3** | **56.5** | **68.1** |

Table 3: **Comparison with existing methods.** medR (↓), Recall@k (↑) are reported on the Recipe1M test set. $\diamond$ indicates that methods use all training samples in Recipe1M for training as opposed to using paired image-recipe samples only.

### 4.5. Comparison to existing works

We compare the performance of our method with existing works in Table 3, where we take our best performing model on the validation set, and evaluate its performance on the test set. For comparison, we provide numbers reported by authors in their respective papers. When trained with paired data only, our model $\mathcal{L}_{pair} + \mathcal{L}_{rec}$ achieves the best results compared to recent methods trained using the same data, achieving an image-to-recipe R1 of 27.3 on 10k-sized rankings (c.f. 24.4 DaC [13], 23.7 SCAN [43], and 20.3 MCEN [14]). When we incorporate the additional unpaired data with no images, it makes a further improvement in the retrieval accuracy (R1 of 27.9, and R5 of 56.4), while still outperforming the state-of-the-art method of DaC$^\diamond$ [13], which jointly embeds pre-trained recipe embeddings (trained on the full training set) and pre-trained image representations using triplet loss. Compared to previous works, we use raw recipe data as input (as opposed to using partial recipe information, or pre-trained embeddings), and train the model with a simple loss functions that are directly applied to the output embeddings and intermediate representations. Our model $(\mathcal{L}_{pair} + \mathcal{L}_{rec})^\diamond$ achieves state-of-the-art results for all retrieval metrics (medR and recall) and retrieval scenarios (image-to-recipe and recipe-to-image) for 10k-sized rankings, while being conceptually simpler and easier to train both in terms of data preparation and optimization compared previous works.

### 4.6. Testing with incomplete data

Training with our self-supervised triplet loss on recipe components allows us to easily test our model in missing data scenarios. Once trained, our proposed projection layers described in Section 3.4 allow our model to hallucinate any recipe component from the others, e.g. in case that the title is missing, we can simply take the average of the two respective projected vectors from the ingredients and the instructions: $e_{ttl}$ as $(g_{ing \to ttl}(e_{ing}) + g_{ins \to ttl}(e_{ins}))/2$ (see

| | medR | R1 | R5 | R10 |
|---|---|---|---|---|
| No title | 6.0 | 22.7 | 48.4 | 60.4 |
| Hallucinated $e_{ttl}$ | 5.0 | 24.2 | 51.2 | 63.1 |
| No ingredients | 10.2 | 16.0 | 38.3 | 50.2 |
| Hallucinated $e_{ing}$ | 10.1 | 16.6 | 39.1 | 50.8 |
| No instructions | 6.0 | 22.3 | 48.0 | 59.8 |
| Hallucinated $e_{ins}$ | 6.0 | 23.1 | 49.4 | 61.1 |
| Title only | 35.5 | 6.0 | 18.9 | 28.4 |
| Hallucinated $e_{ing}$, $e_{ins}$ | 35.8 | 6.6 | 20.0 | 29.3 |
| Ingredients only | 8.3 | 19.2 | 42.5 | 53.9 |
| Hallucinated $e_{ttl}$, $e_{ins}$ | 8.0 | 19.4 | 43.5 | 55.3 |
| Instructions only | 15.0 | 13.1 | 32.6 | 43.8 |
| Hallucinated $e_{ttl}$, $e_{ing}$ | 13.9 | 14.0 | 34.1 | 45.4 |

Table 4: **Testing with missing data.** Image-to-recipe retrieval results reported on the test set of Recipe1M. Results reported on rankings of size $10k$.

| | medR | R1 | R5 | R10 |
|---|---|---|---|---|
| DaC (ResNeXt-101) [13] | 4.0 | 30.0 | 56.5 | 67.0 |
| ResNet-50 | 4.0 | 27.9 | 56.4 | 68.1 |
| ResNeXt-101 | 4.0 | 28.9 | 57.4 | 69.0 |
| ViT | **3.0** | **33.5** | **62.2** | **72.9** |

Table 5: **Comparison of different image encoders.** Image-to-recipe retrieval results reported on the test set of Recipe1M. Results reported on rankings of size $10k$.

Figure 4: **Qualitative results.** Each row includes the query (image or recipe) on the left (highlighted in blue), followed by the top $K = 5$ retrieved recipes. The correct retrieved element is highlighted in green.

Figure 3 for reference). We pick the model trained with $\mathcal{L}_{pair} + \mathcal{L}_{rec}$ and evaluate its image-to-recipe performance when some recipe component features are replaced with their hallucinated versions. In Table 4, we compare models using hallucinated features with respect to the ones in Table 2, i.e. those that ignore those inputs completely during training. In all missing data combinations, we see a consistent improvement over the cases where the missing data is not used during training. Results suggest that using all recipe components during training can improve performance even when some of them are missing at test time.

### 4.7. Image Encoders

We report the performance of our best model $(\mathcal{L}_{pair} + \mathcal{L}_{rec})^{\diamond}$ using different image encoders in Table 5. For comparison with [13], we train our model with ResNeXt-101 image encoder. For $R\{5, 10\}$, we achieve favourable performance with respect to [13] while sharing the same medR score when using the same encoder. We also experiment with the recently introduced Visual Transformer (ViT) [12] as image encoder, and achieved substantial improvement for all metrics: medR of 3.0 and $R\{1, 5, 10\}$ improvement of 3.5, 5.7 and 5.9 points, respectively compared to the best reported results so far on Recipe1M (Table 5, row 1).

### 4.8. Qualitative results

Figure 4 shows some qualitative image-to-recipe and

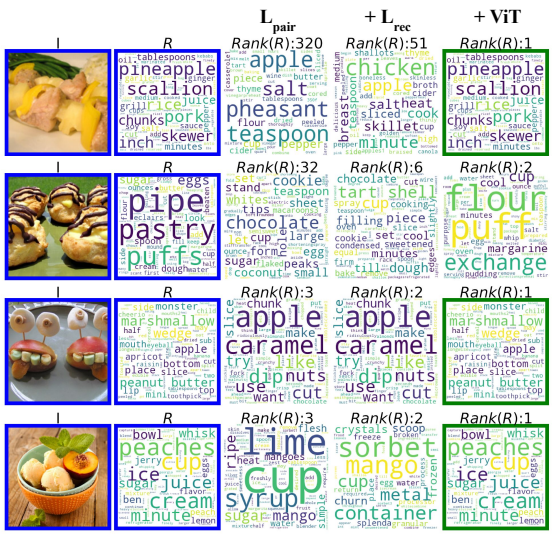



Figure 5: **Incremental improvements.** Each row includes top-1 retrieved recipes for different methods. From left to right: a) Query Image, b) True Recipe, c) $\mathcal{L}_{pair}$, d) $(\mathcal{L}_{pair}+\mathcal{L}_{rec})^{\diamond}$, and e) $(\mathcal{L}_{pair}+\mathcal{L}_{rec})^{\diamond}$ (ViT).

recipe-to-image retrieval using our learned embeddings using the best performing model from Table 5[4]. Our model is able to find recipes that include relevant ingredient words to the query food image (*e.g.* *bread* and *garlic* in the first row, *salmon* in the fifth row). Figure 5 shows examples of the performance improvement of our different models. When adding our proposed recipe loss $\mathcal{L}_{rec}$, and replacing the image model with ViT, the rank of the correct recipe ($Rank(R)$) is improved, as well as the relevance of the top retrieved recipe with respect to the correct one in terms of the common words. These results indicate that our proposed model not only improves retrieval accuracy, but also returns more semantically similar recipes with respect to the query.

## 5. Conclusion

In this work, we study the cross-modal retrieval problem in the food domain by addressing different limitations from previous works. We first propose a textual representation model based on hierarchical Transformers outperforming LSTM-based recipe encoders. Secondly, we propose a self-supervised loss to account for relations between different recipe components, which is straightforward to add on top of intermediate recipe representations, and significantly improves the retrieval results. Moreover, this loss allows us to train using both paired and unpaired recipe data (i.e. recipes without images), resulting in further boost in performance. As a result of our contributions, our method achieves state-of-the-art results in the Recipe1M dataset.

[4]Recipes are shown as word clouds (https://amueller.github.io/word_cloud/) for simplicity.

## References


[1] Mustafa Sercan Amac, Semih Yagcioglu, Aykut Erdem, and Erkut Erdem. Procedural reasoning networks for understanding multimodal procedures. *arXiv preprint arXiv:1909.08859*, 2019.

[2] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101–mining discriminative components with random forests. In *ECCV*, 2014.

[3] Micael Carvalho, Rémi Cadène, David Picard, Laure Soulier, Nicolas Thome, and Matthieu Cord. Cross-modal retrieval in the cooking context: Learning semantic text-image embeddings. In *ACM SIGIR Conference on Research & Development in Information Retrieval*, 2018.

[4] Khyathi Chandu, Eric Nyberg, and Alan W Black. Storyboarding of recipes: Grounded contextual generation. In *ACL*, 2019.

[5] Jing-Jing Chen and Chong-Wah Ngo. Deep-based ingredient recognition for cooking recipe retrieval. In *ACM MM*. ACM, 2016.

[6] Jing-Jing Chen, Chong-Wah Ngo, and Tat-Seng Chua. Cross-modal recipe retrieval with rich food attributes. In *ACM MM*. ACM, 2017.

[7] Jing-Jing Chen, Chong-Wah Ngo, Fu-Li Feng, and Tat-Seng Chua. Deep understanding of cooking procedure for cross-modal recipe retrieval. In *ACM MM*, 2018.

[8] Mei-Yun Chen, Yung-Hsiang Yang, Chia-Ju Ho, Shih-Han Wang, Shane-Ming Liu, Eugene Chang, Che-Hua Yeh, and Ming Ouhyoung. Automatic chinese food identification and quantity estimation. In *SIGGRAPH Asia 2012 Technical Briefs*, 2012.

[9] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

[10] Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*, 2015.

[11] Xin Chen, Hua Zhou, and Liang Diao. Chinesefoodnet: A large-scale image dataset for chinese food recognition. *CoRR*, abs/1705.02743, 2017.

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[13] Mikhail Fain, Andrey Ponikar, Ryan Fox, and Danushka Bollegala. Dividing and conquering cross-modal recipe retrieval: from nearest neighbours baselines to sota. *arXiv preprint arXiv:1911.12763*, 2019.

[14] Han Fu, Rui Wu, Chenghao Liu, and Jianling Sun. Mcen: Bridging cross-modal gap between cooking recipes and dish images with latent variable model. In *CVPR*, 2020.

[15] Jiuxiang Gu, Jianfei Cai, Shafiq R Joty, Li Niu, and Gang Wang. Look, imagine and match: Improving textual-visual cross-modal retrieval with generative models. In *CVPR*, 2018.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[17] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 1997.

[18] Yan Huang and Liang Wang. Acmm: Aligned cross-modal memory for few-shot image and sentence matching. In *ICCV*, 2019.

[19] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *CVPR*, 2015.

[20] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[21] Ryan Kiros, Yukun Zhu, Russ R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Skip-thought vectors. In *NeurIPS*, 2015.

[22] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012.

[23] Kuang-Huei Lee, Xiaodong He, Lei Zhang, and Linjun Yang. Cleannet: Transfer learning for scalable image classifier training with label noise. In *CVPR*, 2018.

[24] Jiatong Li, Fangda Han, Ricardo Guerrero, and Vladimir Pavlovic. Picture-to-amount (pita): Predicting relative ingredient amounts from food images. *arXiv preprint arXiv:2010.08727*, 2020.

[25] Chang Liu, Yu Cao, Yan Luo, Guanling Chen, Vinod Vokkarane, and Yunsheng Ma. Deepfood: Deep learning-based food image recognition for computer-aided dietary assessment. In *ICOST*, 2016.

[26] Yang Liu and Mirella Lapata. Hierarchical transformers for multi-document summarization. *ACL*, 2019.

[27] Austin Meyers, Nick Johnston, Vivek Rathod, Anoop Korattikara, Alex Gorban, Nathan Silberman, Sergio Guadarrama, George Papandreou, Jonathan Huang, and Kevin P Murphy. Im2calories: towards an automated mobile vision food diary. In *ICCV*, 2015.

[28] Simon Mezgec and Barbara Koroušić Seljak. Nutrinet: A deep learning food and drink image recognition system for dietary assessment. *Nutrients*, 9(7), 2017.

[29] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *ICLR*, 2013.

[30] Weiqing Min, Shuqiang Jiang, Linhu Liu, Yong Rui, and Ramesh Jain. A survey on food computing. *ACM Computing Surveys (CSUR)*, 52(5):1–36, 2019.

[31] Weiqing Min, Linhu Liu, Zhiling Wang, Zhengdong Luo, Xiaoming Wei, Xiaolin Wei, and Shuqiang Jiang. Isia food-500: A dataset for large-scale food recognition via stacked global-local attention network. In *ACM MM*, 2020.

[32] Chong-Wah Ngo. Deep learning for food recognition. In *SoICT*, 2017.

[33] Taichi Nishimura, Atsushi Hashimoto, and Shinsuke Mori. Procedural text generation from a photo sequence. In *Proceedings of the 12th International Conference on Natural Language Generation*, 2019.

[34] Ferda Ofli, Yusuf Aytar, Ingmar Weber, Raggi al Hammouri, and Antonio Torralba. Is saki# delicious?: The food perception gap on instagram and its relation to health. In *ICWWW*, 2017.

[35] Siyuan Pan, Ling Dai, Xuhong Hou, Huating Li, and Bin Sheng. Chefgan: Food image generation from recipes. In *ACM MM*, 2020.

[36] Amaia Salvador, Michal Drozdzal, Xavier Giro-i Nieto, and Adriana Romero. Inverse cooking: Recipe generation from food images. In *CVPR*, 2019.

[37] Amaia Salvador, Nicholas Hynes, Yusuf Aytar, Javier Marin, Ferda Ofli, Ingmar Weber, and Antonio Torralba. Learning cross-modal embeddings for cooking recipes and food images. In *CVPR*, 2017.

[38] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *ACL*, 2018.

[39] Jonathan C Stroud, David A Ross, Chen Sun, Jia Deng, Rahul Sukthankar, and Cordelia Schmid. Learning video representations from textual web supervision. *arXiv preprint arXiv:2007.14937*, 2020.

[40] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

[41] Hao Wang, Guosheng Lin, Steven CH Hoi, and Chunyan Miao. Structure-aware generation network for recipe generation from images. *ECCV*, 2020.

[42] Hao Wang, Doyen Sahoo, Chenghao Liu, Ee-peng Lim, and Steven CH Hoi. Learning cross-modal embeddings with adversarial networks for cooking recipes and food images. In *CVPR*, 2019.

[43] Hao Wang, Doyen Sahoo, Chenghao Liu, Ke Shu, Palakorn Achananuparp, Ee-peng Lim, and Steven CH Hoi. 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.

[44] Xin Wang, Devinder Kumar, Nicolas Thome, Matthieu Cord, and Frederic Precioso. Recipe recognition with large multimodal food dataset. In *ICMEW*, 2015.

[45] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017.

[46] Semih Yagcioglu, Aykut Erdem, Erkut Erdem, and Nazli Ikizler-Cinbis. Recipeqa: A challenge dataset for multimodal comprehension of cooking recipes. *arXiv preprint arXiv:1809.00812*, 2018.

[47] Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2, 2014.

[48] Xingxing Zhang, Furu Wei, and Ming Zhou. Hibert: Document level pre-training of hierarchical bidirectional transformers for document summarization. *ACL*, 2019.

[49] Bin Zhu and Chong-Wah Ngo. Cookgan: Causality based text-to-image synthesis. In *CVPR*, 2020.

[50] Bin Zhu, Chong-Wah Ngo, Jingjing Chen, and Yanbin Hao. R2gan: Cross-modal recipe retrieval with generative adversarial network. In *CVPR*, 2019.