FIRE: Food Image to REcipe generation

Prateek Chhikara[1,2], Dhiraj Chaurasia[1,2], Yifan Jiang[1,2], Omkar Masur[1], and Filip Ilievski[2,3]
[1]University of Southern California, USA, [2]Information Sciences Institute, USA
[3]Vrije Universiteit Amsterdam, Netherlands
{pchhikar,chaurasi,yifjia}@isi.edu, omasur@usc.edu, f.ilievski@vu.nl

Abstract

Food computing has emerged as a prominent multidisciplinary field of research in recent years. An ambitious goal of food computing is to develop end-to-end intelligent systems capable of autonomously producing recipe information for a food image. Current image-to-recipe methods are retrieval-based and their success depends heavily on the dataset size and diversity, as well as the quality of learned embeddings. Meanwhile, the emergence of powerful attention-based vision and language models presents a promising avenue for accurate and generalizable recipe generation, which has yet to be extensively explored. This paper proposes `FIRE`, a novel multimodal methodology tailored to recipe generation in the food computing domain, which generates the food title, ingredients, and cooking instructions based on input food images. `FIRE` leverages the BLIP model to generate titles, utilizes a Vision Transformer with a decoder for ingredient extraction, and employs the T5 model to generate recipes incorporating titles and ingredients as inputs. We showcase two practical applications that can benefit from integrating `FIRE` with large language model prompting: recipe customization to fit recipes to user preferences and recipe-to-code transformation to enable automated cooking processes. Our experimental findings validate the efficacy of our proposed approach, underscoring its potential for future advancements and widespread adoption in food computing.

1. Introduction

Food is not only a vital source of sustenance but also an integral part of our cultural identity, defining our lifestyle, traditions, and social interactions [31]. As the well-known saying goes, *"Tell me what you eat, and I will tell you who you are,"* [28] emphasizing the idea that an individual's dietary choices reflect their identity [30]. Moreover, a person's physical appearance and cognitive abilities often bear evidence of their dietary habits, as the selection of nutritious food contributes to the overall well-being of both the body and mind [46]. The advent of social media enables anyone to share captivating visuals of personal experiences related to the delectable food they consume. A simple search for hashtags like `#food` or `#foodie` yields millions of posts, underscoring the immense value of food in our society [17]. The significance of food accompanied by its large amounts of publicly available data has inspired food computing tasks [31] that associate visual depictions of dishes with symbolic information. An ambitious goal of food computing is to produce the recipe for a given food image, with applications such as food recommendation according to user preferences, recipe customization to accommodate cultural or religious factors, and automating cooking execution for higher efficiency and precision [33].

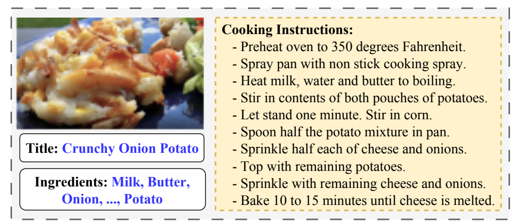

Figure 1. Given a potentially unseen image, our method **FIRE** generates a corresponding recipe consisting of a title, ingredients, and cooking instructions.

Generating detailed recipe information or cooking procedures solely from a food image presents a considerable challenge [40, 41]. Food computing has been of interest to the computer vision (CV) community, whose efforts to use image processing for food quality assurance can be traced back to 1996 [11]. State-of-the-art food image processing methods [15, 27, 41] use deep learning techniques to extract ingredients from images with limited success. Meanwhile, a popular natural language processing (NLP) application has been recipe generation, a procedural task of creating recipes based on a flexible set of ingredients as inputs. Typical models for recipe generation include [29, 41, 51]. We note that prior work has not connected the dots between the CV and NLP research in order to provide an end-to-end system that generates recipes from images. Moreover, current methods for food computing have not caught up with the most recent advances in NLP and CV, featuring diffusion models and language modeling.

This paper presents a novel multimodal methodology that we call **FIRE** (**F**ood **I**mage to **RE**cipe generation). **FIRE** is designed to generate comprehensive recipe output for the food computing domain, including food titles, ingredients, and cooking instructions, based on input food images as shown in Figure 1. We leverage recent advancements in CV and language modeling to employ state-of-the-art (SotA) techniques that have demonstrated exceptional performance in various vision and language tasks. **FIRE** connects the dots between SotA models, using BLIP [21] for title generation, a Vision Transformer [9] with a decoder for ingredient extraction, and the T5 [37] model for cooking instruction generation. Furthermore, we highlight two practical applications that can benefit from integrating **FIRE** with prompting large language models: *recipe customization* for personalized recipe adaptation, and *recipe-to-code generation*, enabling automated cooking processes. The contributions of the paper are as follows:

1. We leverage the capabilities of Vision Transformers (ViT) [9] to get expressive embeddings from food images, which are subsequently fed into an attention-based decoder to extract the ingredients of the recipe.
2. We present an end-to-end pipeline for generating recipe titles and cooking instructions, utilizing SotA vision (BLIP) and language (T5) models, respectively.
3. Our multimodal approach outperforms the existing work based on two evaluation metrics: (a) set metrics for ingredient extraction and (b) document-level metrics for cooking instruction generation.
4. We showcase the ability of **FIRE** to support two novel food computing applications: *Recipe Customization* and *Recipe to Code Generation*, through integration with few-shot prompting of large LMs.

We organize this paper as follows; Section 2 of the paper gives a detailed overview of the related work in the field of food computing and its gap against SotA models. We describe our proposed methodology (**FIRE**) in Section 3. Section 4 describes the experimental setup we follow to obtain the results, which are presented in Section 5. Section 6 illustrates two advanced applications that can benefit from our proposed approach. Finally, we conclude our paper in Section 7 with future research directions. We make all of our code available to stimulate work on recipe generation.

2. Related Work

2.1. Food Computing

Recently, the importance of food and the availability of extensive multimodal food datasets, such as Food-101 [4], Recipe1M [26], and Recipes242k [39], have enabled computational research on food computing tasks [31]. We review prior work on food recognition and recipe generation.

Food Recognition is an image-to-text task requiring models to detect food categories in a food image. Recognition of food items can offer people comprehensive information and a better understanding of unfamiliar dishes, thereby improving other food-related applications as well [1, 34]. Previous works focus on extracting deep representations of food [15,27,41,55]. Martinel *et al.* [27] adapted a slice convolution block in the residual network to capture features in images. Salvador *et al.* [41] proposed InverseCooking, an encoder-decoder framework to output the title of the food. Wang *et al.* [50] also utilized images to get the recipe by treating it as an image captioning task. Notably, earlier architectures are constrained as they tend to emphasize global features rather than local features and can not detect the ingredients overlapping in the image [19]. Instead, we employ a SotA vision encoder, ViT [9], to enhance the extraction of the local semantic segmentation.

Recipe Generation is a more complex text-to-sequence task generating food recipes based on the ingredients provided. To solve this task, models must possess knowledge of food composition, ingredients, and cooking procedures to perform the task accurately. Early attempts at recipe generation were constrained by limited model capacity and structure, leading to solutions that relied on information retrieval techniques [52,56]. Wang *et al.* [52] developed a novel similarity and filtering algorithm to increase the search accuracy. Xie *et al.* [56], leveraged the cooking flow and eating features with other domain knowledge to enhance the searching process. More recent work relies on encoder-decoder structures to generate recipes [41, 51] with multimodal settings. Salvador *et al.* [41] presented a framework that utilizes encoded image and ingredients representations in recipe generation. Wang *et al.* [51] added tree structures within the encoder-decoder process to incorporate structure-level information. In contrast to prior unimodal work, our approach uses images as input and generates titles and ingredients as an intermediate representation, and uses them to generate recipes.

2.2. State-of-the-Art Models

We review state-of-the-art models that have not found a broad application in food computing tasks to date.

Image to Text Models gradually play an important role in vision language tasks with the development of deep learning [20]. Models follow a main pipeline that encodes image input into intermediate stages and then decodes them into text output [49, 57]. Vinyals *et al.* [49] encoded the input image into a global visual vector through CNN and then applied RNN to generate captions, whose generalization on multi-task also supported the effect of the encoder-decoder pipeline. Xu *et al.* [57] further built an attention mechanism to pick the most related subregion vectors rather than depending only on the global visual vector. De-

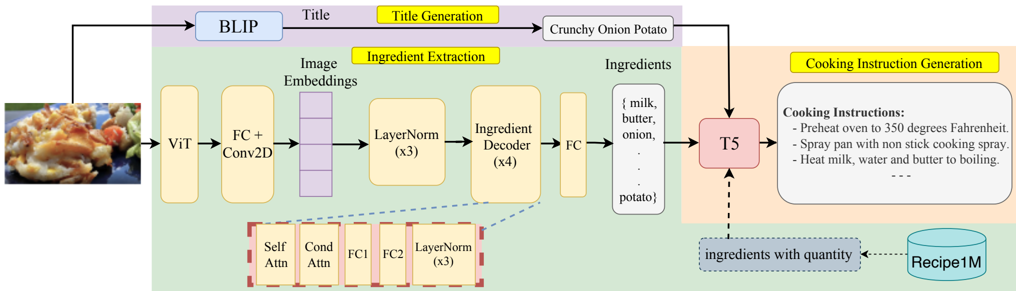

Figure 2. Proposed architecture to extract ingredients, and generate the recipe title and cooking instructions from a food image. (*Ingredients with quantity is passed during the train time only*)

spite the progress in the encoder-decoder pipelines, their ability is limited by only emphasis on single modularity input. Recent research switched to unimodality with the birth of large image-text datasets [13, 58]. CLIP [35] modified GPT-2 [36] to obtain text features from textual input and used image-text contrastive learning, which trained the model with the similarity between the image and text. ALBEF [22] utilized ViT (pre-trained on large amounts of data and transferred to multiple mid-sized or small image recognition benchmarks) [9] as an image encoder and BERT [8] as a text encoder to extract information with attention. One additional multimodal encoder was built on the extraction output, which additionally incorporated masked-language-modeling loss (MLM) to enhance the image-text interactions. Based on previous work, BLIP model [21], designed to achieve unified vision-language understanding and generation, reached state-of-the-art results on various vision-language tasks in a zero-shot manner [60]. By employing a captioner to generate synthetic captions and applying a filtering mechanism, BLIP maximizes the utilization of noisy web data. In our work, we leverage BLIP and ViT to generate food titles and ingredients separately.

Text (Sequence) to Sequence Models takes a sequence of text as input and maps it into a succession of another sequence [59]. For this task, previous work either built recurrent neural networks (RNNs) [2, 7, 45] or attention-based models [8, 36, 37, 48]. GPT-2 [36] is based on a transformer-decoder to perform tasks on various fields in a zero-shot setting, while T5 (text-to-text encoder-decoder model) [37] transformed text-related tasks in a text-to-sequence format to enhance its general ability. With the recent progress of large pre-trained LMs, prompting has become a popular and efficient approach to tackle many NLP tasks [23]. More specifically, few-shot prompting provided input-output mapping in demonstration to guide LMs to prompt the specific structure. Chain-of-thought (CoT) [54] with Self-Consistency [53] reached state-of-art on commonsense reasoning and symbolic reasoning, even compared to supervised models. Madaan *et al.* [25] used graph demonstration to hint LMs generate complex Python classes for reasoning and state tracking. Capitalizing on these advancements, we employ T5 to generate cooking instructions from food ingredients and titles. To overcome resource and time constraints posed by large-scale food dataset, we also show how few-shot prompting can support food computing applications. Our work analyzes how to generate a curated recipe or convert recipe into a structured program flow [33] for further application with the help of prompting.

3. Proposed Methodology (`FIRE`)

FIRE consists of three components: `(1)` title generation from food images by using state-of-the-art image captioning, `(2)` ingredient extraction from images using vision transformers and decoder layers with attention, and `(3)` cooking instruction generation based on the generated title and extracted ingredients using an encoder-decoder model.

3.1. Title Generation

We generate recipe titles from food images using the BLIP model, a state-of-the-art image captioning approach. In our initial experiments with the off-the-shelf BLIP model, we observed promising results, yet, BLIP's prediction accuracy was lower because of the domain shift between its training data and the food domain. Namely, BLIP tends to capture extraneous details impertinent to our goal because it was originally designed to provide a comprehensive image caption for a wide variety of settings. As an illustration, when presented with an image of a muffin, BLIP produced the description *'a muffin positioned atop a wooden cutting board'*. To better align the generated captions with recipe titles, we fine-tune the BLIP model using a subset of the Recipe1M dataset. We restrict our tuning to 10% of the training dataset, as fine-tuning on the entire dataset is computationally intensive [18]. We observe that the fine-tuned version of BLIP shows promising improvements in generating accurate, aligned, and pertinent titles for food images. The fine-tuned BLIP captions the same example image with a shorter string *'muffin'*, removing the additional extraneous information.

3.2. Ingredient Extraction

Extracting ingredients from a given food image presents challenges due to the inherent complexity and variability of food compositions. Unlike generating titles or captions, determining a comprehensive and accurate list of ingredients requires a deep understanding of food characteristics, textures, and interactions. Additionally, the visual appearance of certain ingredients may overlap, whereas others may not be visible at all, leading to potential ambiguities and difficulties in discerning specific components. For instance, in a food image that contains a dish with melted cheese on top, from visual appearance alone it may be challenging to determine if the cheese used is mozzarella, cheddar, or any other type. These nuances motivate need for an architecture capable of retaining expressive embeddings from food images. While finetuned BLIP successfully generated accurate titles for food images, our analysis showed that using it for ingredient extraction led to significant hallucinations in the output. As BLIP is primarily trained for image captions, it struggles to generate ingredient lists accurately. To address this challenge, we develop an ingredient extraction pipeline (shown in Figure 2) built on top of the one proposed by [41].

Feature Extractor: We extract the image's features by employing a vision transformer (ViT). ViT's attention mechanism enables for effective handling of feature representations with stable and notably high resolution. This capability precisely meets the requirements of dense prediction tasks such as ingredient extraction from food images [63]. Furthermore, transformer-based approaches exhibit minimal reliance on the inductive bias, facilitating effective interaction and integration of long-range information. Unlike conventional CNNs, the output of a ViT is sequential; therefore, we use a fully connected (FC) linear layer to reshape the output and pass it to a 2D convolution (Conv2D) layer.

Ingredient Decoder: The feature extractor produces image embeddings. We pass these image embeddings through three normalization layers (*layerNorm*) and subsequently feed the output into our ingredient decoder responsible for extracting ingredients. The decoder consists of four consecutive blocks, each comprising multiple sequential layers: self-attention, conditional attention, two fully connected layers, and three normalization layers. In the last step, the decoder output is processed by a fully connected layer with a node count equivalent to the vocabulary size, resulting in a predicted set of embeddings.

Given a corpus with ingredients and recipes corresponding to food images, we construct a dictionary D consisting of N possible ingredients. Each recipe, r_i, is associated with a set S, comprising K ingredients selected from this dictionary. Given that the order of the ingredients does not affect the resulting recipe, we represent the ingredients as a set rather than a list. In other words, we exploit codependencies among ingredients without penalizing for prediction order. We represent the ingredient set S using a binary vector, s, of dimension N, where s_i = 1 if $s_i \in S$, and 0 otherwise. Consequently, our training dataset consists of m pairs of image and ingredients sets: $\{(x_i, s_i)\}_{i=0}^{m}$. In this case, the goal is to predict $\hat{s}$ from an image x by maximizing the following objective:

$$\arg\max_{\theta_{img},\theta_{ing}} \sum \log p(\hat{s}_i = s_i | x_i; \theta_{img}, \theta_{ing}) \quad (1)$$

where θ_{img} and θ_{ing} represent the learnable parameters for the image encoder and ingredient decoder, respectively. While there may exist certain dependencies among the ingredients, such as the common combination of salt and pepper, these dependencies do not exert a dominant influence. Consequently, we can reasonably assume independence between the ingredients and factorize them as follows:

$$\sum_{j=1}^{N} \log p(\hat{s}_{ij} = s_{ij} | x_i) \leftarrow p(\hat{s}_i = s_i | x_i) \quad (2)$$

Our decoder makes ingredient predictions sequentially until it encounters an end-of-sequence (EOS) token. To mitigate the impact of the order of the ingredients, we aggregate the outputs separately across different time steps and use max pooling at the end to obtain the ingredient set. This enables training the model using binary cross-entropy loss ($loss_{ingr}$) between the predicted ingredients (after pooling) and the ground truth. However, since the EOS information is lost during pooling, we use a custom EOS loss ($loss_{eos}$). This loss calculates the binary cross-entropy between the predicted EOS probabilities at all time steps and the corresponding ground truth. Furthermore, to enhance performance, we incorporate a cardinality L1 penalty ($loss_{card}$), which constrains the length of the predicted ingredients to be close to the ground truth ingredients. We empirically find that integrating the $loss_{card}$ leads to better performance.

$$loss = \alpha_1 \times loss_{ingr} + \alpha_2 \times loss_{eos} + \alpha_3 \times loss_{card} \quad (3)$$

where, α_1=100 , α_2=1 , and α_3=1 are the hyper-parameters.

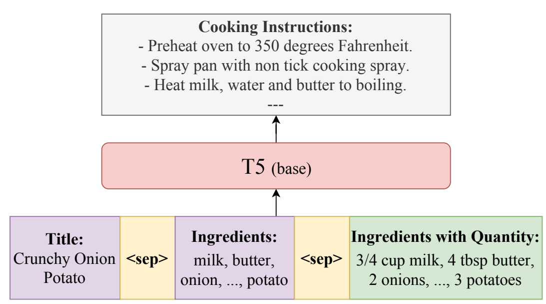

Figure 3. Generating cooking instructions for a title and a set of ingredients. (ingredients with quantity is present only during the fine-tuning of T5)

3.3. Cooking Instruction Generation

Considering the remarkable accomplishments of LMs in natural language applications like text generation and question answering [6], we pose cooking instruction generation as a language modeling task. Large LMs such as GPT [5], LLaMa [47], and Alpaca [38] are pre-trained with billions of tokens with multiple training objectives, which makes them capable of understanding language in context. Refining the LMs for downstream tasks has demonstrated remarkable outcomes in various NLP assessments. While we expect that large LMs would be capable of generating cooking instructions after fine-tuning, they require prohibitive computational resources given their large number of parameters. Given the available resources and our research objective, we adopt popular encoder-decoder model, T5 [37], for generating cooking instructions. We conduct all experiments using base T5 model with 220M parameters. During finetuning, we pass title and ingredients of the recipe as a formatted string (see Figure 3), inspired by prior work [61].

The T5 is finetuned on three inputs: title, ingredients, and ingredients with quantity to incorporate maximum information from the dataset. However, we do not have ingredients with quantity at inference time; hence we can pass only the title and ingredients. Moreover, excluding the quantity information from our model ensures a fair comparison with previous approaches and investigates whether our model's advantage stems from a well-structured architecture rather than relying solely on the augmentation of additional knowledge. By removing the influence of quantity information during inference, we aim to highlight the inherent capabilities of T5 and its ability to generate high-quality cooking instructions.

4. Experiment Setup

4.1. Dataset

Recipe1M is a large-scale dataset of over one million cooking recipes [26]. The dataset contains rich food-related information, including recipe titles, ingredients, cooking instructions, and nutritional information. The dataset itself is not provided publicly, instead, the authors provide a list of image URLs that can be scraped from the Web. While scraping this data, we encountered instances where we could not download images due to an expired URL or a corrupt image. Additionally, some of the recipes in the dataset did not have any accompanying images. Therefore, we only utilized recipes with at least one corresponding image available to us. Following the dataset filtering process, we obtained a training set comprising 259,932 samples, a development set containing 55,773 samples, and a test set consisting of 56,029 samples adequate for recipe analysis.

4.2. Baselines

Ingredient Extraction: We present a comparative analysis of **FIRE**'s ingredient extractor against two retrieval-based techniques: R_{I2L} and R_{I2LR} [42]. R_{I2L} learns joint embeddings between images and ingredient lists and uses them to retrieve the most relevant recipe within the embedding space. R_{I2LR} expands upon this approach by incorporating the joint embedding between recipe title, instructions, ingredients, and the corresponding food image to further enhance retrieval. We also compare our approach to two state-of-the-art generative models, namely FF_{TD} and InverseCooking [41]. FF_{TD} models the joint distribution of the ingredients set by utilizing the target distribution and greedily sampling from a cumulative distribution of sorted output probabilities until the sum of probabilities of selected elements exceeds a specified threshold. InverseCooking is an attention-based model that takes embeddings from ResNet50 as input and uses a transformer decoder architecture for ingredient generation.

Cooking Instruction Generation: As baselines, we utilize both InverseCooking [41] and Chef Transformer [29]. We specifically select these two baselines as their code was publicly available, and they work on recipe generation rather than retrieval from a database. InverseCooking is an end-to-end recipe generation model that takes food images as input and extracts ingredients, which along with image embeddings, are used to generate the title and the cooking instructions. Like our method, InverseCooking is also trained on Recipe1M. Chef Transformer is trained on the RecipeNLG [3] dataset and exclusively relies on ingredient inputs rather than food images for cooking instructions generation. Therefore, in the case of the Chef Transformer, we use ground truth ingredients for testing.

4.3. Evaluation Metrics

We evaluate **FIRE** on end-to-end cooking instruction generation and through ablation study on ingredient extraction. For end-to-end **recipe generation**, we employ document-level evaluation metrics: *SacreBLEU* and *RougeL* to assess the quality of our model's output. Since the output of **ingredient extraction** models is a set, we evaluate their performance using *F1-score/Dice score*, and *Jaccard/IoU* similarity index, computed for accumulated counts of true positives, false negatives, and false positives over the entire dataset.

5. Results & Analysis

5.1. End-to-End Recipe Generation

The results in Table 1 show that **FIRE** exhibits superior performance compared to the two SotA baselines, InverseCooking and Chef Transformer. These results demonstrate our proposed pipeline's ability to generate precise and coherent recipes, corroborating the effectiveness of **FIRE** and emphasizing the value of language generation models for high-quality recipe generation. These results also support our expectation that the **FIRE** method can generalize well without ingredient quantity information given at inference time, even when they were present during training. Meanwhile, we observe that training with extra information results in fewer hallucinations, especially regarding ingredi-

Table 1. Recipe generation comparison on the test dataset. We report mean with one standard deviation of 10 experiments. **Bold** represents the best model. ($^+$) represents the model tested on the ground truth title and ingredients to generate the recipe

	SacreBLEU	ROUGE L
Chef Transformer [29]	4.61 ± 0.32	17.54 ± 0.19
InverseCooking [41]	5.48 ± 0.21	19.47 ± 0.15
FIRE (without $loss_{card}$)	5.91 ± 0.17	20.87 ± 0.13
FIRE (ResNet50)	5.87 ± 0.10	20.49 ± 0.08
FIRE	6.02 ± 0.15	21.29 ± 0.10
FIRE$^+$	$\mathbf{7.29} \pm 0.11$	$\mathbf{25.17} \pm 0.07$

ents quantity (e.g., 2 tablespoons of salt) and cooking time (e.g., heat for 10-12 minutes).

FIRE with automatically extracted title and ingredients achieved a relative improvement over InverseCooking of 6% and 8% on SacreBLEU and RougeL scores, respectively. Notably, InverseCooking incorporates both image embeddings and automatically extracted ingredients during the cooking instruction generation phase. Meanwhile, **FIRE**'s instruction generation language model relies on the recipe title and ingredients only, which provide **FIRE** with informative signals to generate comprehensive recipes.

As Chef Transformer does not support image input, it uses ground-truth ingredients for cooking instruction generation. In comparison, **FIRE** faces realistic challenges due to noisy ingredient extraction. Yet, **FIRE** easily outperforms Chef Transformer, and the gap increases further when **FIRE** is also provided with a ground-truth title and ingredient set. The low performance of Chef Transformer on this task can be attributed to its architecture and its reliance on just ingredients without any title information. As a set of ingredients can correspond to multiple recipes, the title may be crucial for disambiguation and coherence. For example, both Stir-fried Ginger Chicken and Garlic Ginger Chicken Soup share the same set of ingredients (*chicken, garlic, soy sauce*, and *ginger*). Despite this commonality, this same set of ingredients leads to two entirely different recipes.

5.2. Ablation Study

Ingredient Extraction The results on the ingredient extraction task are shown in Table 2. The retrieval-based approaches (R_{I2L} and R_{I2LR}) yield poor results. This can be expected, given their reliance on the presence of an exact matching recipe in the static dataset and their dependence on the dataset size and diversity. The models FF_{TD}, InverseCooking, and **FIRE**, which employ conditional generation, exhibit relatively higher performance in capturing ingredient information from food images. Moreover, out of these three models, **FIRE** achieves the highest IoU and F1 scores among all of the models, surpassing the second-ranked InverseCooking model with a relative margin of 1.5% in terms of IoU and 1.4% in terms of F1 score. We attribute this improvement to **FIRE**'s superior feature extraction capability that uses ViT rather than ResNet50.

Table 2. Evaluation results on ingredient extraction using set metrics (IoU and F1). **Bold** represents the best model.

Model	IoU	F1
R_{I2L} [42]	18.92	31.83
R_{I2LR} [42]	19.85	33.13
FF_{TD} [41]	29.82	45.94
InverseCooking [41]	32.11	48.61
FIRE	**32.59**	**49.27**

Table 3. Impact of SotA feature extractors on ingredient extraction. All models are trained and tested on 10% dataset.

Model	IoU	F1
ResNet18	25.88	39.31
ResNet50	26.94	40.51
ResNet101	26.37	40.12
InceptionV3	25.31	38.92
ViT	**27.69**	**42.73**

Table 4. Comparison between zero-shot (ZS) and fine-tuned (FT) versions of BLIP and T5.

Model	ZS	FT
BLIP	17.89	37.72
T5	2.47	6.02

Image Feature Extraction To understand the observed ingredient extraction gap between **FIRE** and InverseCooking, we compare the impact of image feature extractors on ingredient extraction. We ablate our feature extractor (ViT) with state-of-the-art CNN models: ResNet18, ResNet50, ResNet101, and InceptionV3. Table 3 reveals that ViT outperforms the other feature extractors, demonstrating its superior ability to capture and represent food image features relevant to ingredient extraction. Furthermore, to assess the feature extractor's influence on the end-to-end FIRE pipeline, we substituted ViT with ResNet50. This change resulted in a performance decrease, as indicated in Table 1. This finding emphasizes the efficacy of leveraging the state-of-the-art feature extractor ViT for improved results in our food computing system.

Zero-shot vs Fine-tuned We compare the performance of zero-shot BLIP and T5 model against our fine-tuned model. The outcomes are detailed in Table 4. To assess the results for title generation by BLIP, we use a string similarity approach based on the *longest common subsequence* (LCS), as achieving an exact match is infrequent due to the vast array of recipe variations. For example, if the actual title is *'black bean and rice salad'* and BLIP predicts *'black bean and rice,'* then a conventional accuracy metric would yield zero, whereas the LCS score would be 0.76. For T5, we utilize the SacreBLEU metric. The results demonstrates that SotA BLIP and T5 models necessitate task-specific fine-tuning.

Cardinality Loss Complementing $loss_{eos}$ with $loss_{card}$ improves the model's ability to extract the correct ingredients from food images. In contrast, using only binary cross-entropy does not consider dependencies among elements in the set. We trained **FIRE**'s ingredient extraction model without cardinality loss ($loss_{card}$) to check the im-

Figure 4. Recipe prediction by **FIRE** for Pav Bhaji image.

pact of adding this loss in model. Without cardinality loss, we believe the model struggles in realizing correct number of ingredients, which leads to divergence from the ground truth, thus lowering performance as shown in Table 1.

5.3. Error Analysis

In order to gain further insight into the performance of our recipe generation method, we inspected its performance on individual images. As shown in Figure 3, **FIRE** is often able to generate a correct recipe for dishes similar to those present in the Recipe1M dataset. Meanwhile, we also study its ability to provide a recipe for Pav bhaji, a popular Indian dish that is not present in the Recipe1M dataset. **FIRE** generates a recipe for *'tomato and onion sandwich'* as shown in Figure 4. As expected, the generated recipe is unrelated to the intended dish. Other state-of-the-art models are also not able to predict the correct recipe. We acknowledge the need for improvement in our model to better generalize to novel recipes. Meanwhile, we highlight the importance of developing better evaluation metrics. Conventional evaluation metrics such as SacreBLEU and ROUGE, failed to capture the accuracy of the generated recipes and detect certain text hallucinations. Given the significant impact of even a single mistake on the final outcome of a dish, it is crucial to develop a robust metric that can reliably ensure the completion of the desired cooking task beyond text similarity. For additional examples of both successful and unsuccessful cases, please refer to the Appendix.

6. FIRE Applications

While **FIRE** achieves state-of-the-art performance on the ambitious task of generating recipes from images, we go a step further and investigate its integration into larger pipelines for food computing applications. Namely, considering the promise of few-shot prompting of large language models, we describe how **FIRE** and large LMs can be integrated to support recipe customization and recipe-to-machine-code generation. For both applications, we provide a potential pipeline with an illustrated recipe and we conduct a pilot study to investigate its potential.

(1) Recipe Customization
Recipe customization is crucial due to the connection between food, customs, and individual preferences. Additionally, it becomes essential when addressing allergies or dietary restrictions. Surprisingly, despite the evident demand, existing literature lacks dedicated efforts in the domain of recipe customization. We are inspired by the Computer Cooking Contest (CCC) [32], an annual event showcasing computational systems that generate novel and creative recipes, enabling participants to employ AI and computational creativity in exploring innovative food combinations and techniques. However, we cannot use CCC directly because judges perform its evaluation manually. Our work aims to bridge the research gap by enabling personalized recipe customization, considering individual taste profiles and dietary restrictions. To guide future research in this area, we showcase the ability of **FIRE** to support a recipe customization approach that focuses on a wide range of topics (e.g. ingredient replacement, taste adjustment, calories adjustment, cooking time adaptation) to test few-shot performance thoroughly. As shown in the purple part of Figure 5, we perform *ingredient removal* to trim the potatoes from the recipe. Two sentences related to potatoes are deleted in the modified version, and one sentence is modified to ensure consistency. Specifically, we perform *ingredient addition* to replace 'cheese' with 'cheddar cheese' and recognize that it should be added before baking, resulting in the modified sentence 'Sprinkle half each of cheddar cheese and onions.' We manually design four demonstrations to hint GPT-3 to solve the customization requirements.
Analysis To assess the effectiveness of recipe customization, we conducted a human evaluation with seven experts involving 10 recipes and their customizations. Evaluators rated four attributes: efficacy, coherence, soundness, and proportions and measurements, on a 0 to 4 scale (0: strongly disagree, 1: disagree, 2: neutral, 3: agree, 4: strongly agree). Table 1 (Appendix) shows the average experimental results. On average, each attribute has a high result (3.5 to 3.76) with high Fleiss [10] kappa inter-annotator agreement (0.78 to 0.92). The results indicate the promise of integrating our method with few-shot LM prompting. Albeit provided with a limited number of demonstrations, the model can handle complex examples like *Can you make the food with fewer calories?* and replace *milk* with *almond milk*. We refer the reader to the Appendix for further details about the design, the data, and the results of the pilot experiment.

(2) Generating Machine Code for Image-based Recipes
Converting recipes to machine code enables automation, scalability, and integration with various systems, reducing manual intervention, resulting in savings in labor costs and reducing human errors while preparing the food. To fa-

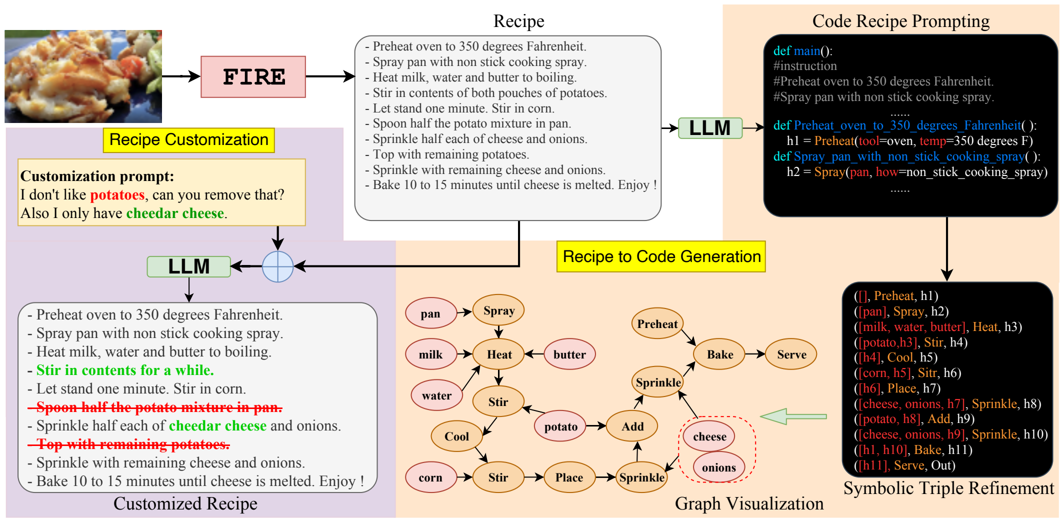

Figure 5. Applications of **FIRE**: Recipe Customization and Recipe to Code Generation.

cilitate this task, we combine **FIRE**'s recipe generation strength with the ability of large LMs to manipulate code-style prompts for structural tasks [25]. We show an example approach for generating Python-style code representations of recipes generated by **FIRE**, by prompting GPT-3 (orange part in Figure 5). This envisioned approach has two phases: code recipe prompting and symbolic triple refinement.

Code Recipe Prompting We convert the output of **FIRE** into a Python-style prompt and leverage GPT-3 to generate code representations as shown in Figure 5. Further, we refine these representations into symbolic triple representations within a predefined space. For all input recipes $r \in R$, we constructed corresponding Python-style prompting r^p and code representation c. For any new recipes r', the input to the prompting pipeline was $r_1^p \oplus c_1 \oplus \cdots \oplus r_k^p \oplus c_k \oplus r'$, where $k = 4$ was the number of demonstrations and the output code representation c' is completion result of GPT-3.

Symbolic Triple Refinement For further use in industrial applications, we refine code generation into symbolic triples (i, r, o), where i and o represent the input list and output of operations, and r represented the cooking instruction and parameter details. This allows for a more structured and standardized representation of the generated code, facilitating easier integration with various applications.

Analysis We conduct a similar human evaluation process focusing on how well ingredients, cooking instructions, and their descriptions are translated to code format on a scale of 0 (extremely poor) to 5 (excellent). Each property is rated on average between 4.27 and 4.47, with an inter-annotator agreement between 0.75 and 0.83. Despite the promising experiment results, few-shot prompting can produce hallucinations when tracking ingredients, especially in long contexts or when similar cooking tools are involved (e.g., saucepan, frying pan), which can be further explored by future work. We refer the reader to the Appendix for details about the design, the data, and the results of the pilot study.

7. Conclusion & Future Work

This paper introduced **FIRE**, a methodology tailored for food computing, focusing on generating food title, extracting ingredients, and generating cooking instructions solely from image inputs. We leveraged recent advancements in CV and language modeling to achieve superior performance against strong baselines. Furthermore, we demonstrated practical applications of **FIRE** for recipe customization and recipe-to-code generation, showcasing adaptability and automation potential of our approach. Experimental results validated the efficacy of **FIRE**, highlighting its promising prospects for future advancements and wide-ranging adoption in food computing. Inspired by our experiments, we list three challenges that should be addressed in future research:

1. A major limitation of both the proposed work and existing approaches lies in the absence of a reliable grounding mechanism [16] to ascertain the correctness of generated recipes. Conventional metrics are insufficient to capture this challenge. We propose to address this limitation by developing a metric that effectively captures the coherence and plausibility of generated recipes, providing a more comprehensive evaluation framework for recipe generation systems.
2. The diversity and availability of recipes are heavily dependent on the locations, climates, and religions [30, 44, 62], which prevent users from preparing food based on predefined recipes. One solution can be the injection of knowledge graphs [12, 43], which reflect the connection between the ingredients based on symbolic relations and contextual factors, thus informing the models about alternative ingredients.
3. Hallucination remains a critical challenge in recipe generation by natural language and vision models. We will investigate the possibility of incorporating methods for state tracking of participants [14,24] to enhance the production of reasonable and accurate results.

References

[1] Eduardo Aguilar, Beatriz Remeseiro, Marc Bolaños, and Petia Radeva. Grab, pay, and eat: Semantic food detection for smart restaurants. *IEEE Transactions on Multimedia*, 20(12):3266–3275, 2018. 2

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014. 3

[3] Michał Bień, Michał Gilski, Martyna Maciejewska, Wojciech Taisner, Dawid Wisniewski, and Agnieszka Lawrynowicz. Recipenlg: A cooking recipes dataset for semi-structured text generation. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 22–28, 2020. 5

[4] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101–mining discriminative components with random forests. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VI 13*, pages 446–461. Springer, 2014. 2

[5] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020. 4

[6] Prateek Chhikara, Ujjwal Pasupulety, John Marshall, Dhiraj Chaurasia, and Shweta Kumari. Privacy aware question-answering system for online mental health risk assessment. In *The 22nd Workshop on Biomedical Natural Language Processing and BioNLP Shared Tasks*, pages 215–222, Toronto, Canada, July 2023. Association for Computational Linguistics. 4

[7] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014. 3

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 3

[9] 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. 2, 3

[10] Joseph L Fleiss. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378, 1971. 7

[11] Sundaram Gunasekaran. Computer vision technology for food quality assurance. *Trends in Food Science & Technology*, 7(8):245–256, 1996. 1

[12] Filip Ilievski, Pedro Szekely, and Bin Zhang. Cskg: The commonsense knowledge graph. In *The Semantic Web: 18th International Conference, ESWC 2021, Virtual Event, June 6–10, 2021, Proceedings 18*, pages 680–696. Springer, 2021. 8

[13] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*, pages 4904–4916. PMLR, 2021. 3

[14] Yifan Jiang, Filip Ilievski, and Kaixin Ma. Transferring procedural knowledge across commonsense tasks. In *ECAI 2023 - 26th European Conference on Artificial Intelligence, September 30 - October 4, 2023, Kraków, Poland*, volume 372 of *Frontiers in Artificial Intelligence and Applications*, pages 1156–1163. IOS Press, 2023. 8

[15] Yoshiyuki Kawano and Keiji Yanai. Food image recognition with deep convolutional features. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication*, pages 589–593, 2014. 1, 2

[16] Kerry Brown. The Nature of Information, Semantics, and Effectiveness for Artificial Intelligence and Cognition. https://doi.org/10.31219/osf.io/dehkj. Accessed on June 14, 2023. 8

[17] Kiely Kuligowski. 12 Reasons to Use Instagram for Your Business. https://www.business.com/articles/10-reasons-to-use-instagram-for-business/. Accessed on May 12, 2023. 1

[18] Jae Myung Kim, A Koepke, Cordelia Schmid, and Zeynep Akata. Exposing and mitigating spurious correlations for cross-modal retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2584–2594, 2023. 3

[19] Fotios S. Konstantakopoulos, Eleni I. Georga, and Dimitrios I. Fotiadis. A review of image-based food recognition and volume estimation artificial intelligence systems. *IEEE Reviews in Biomedical Engineering*, pages 1–17, 2023. 2

[20] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017. 2

[21] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International Conference on Machine Learning*, pages 12888–12900. PMLR, 2022. 2, 3

[22] Junnan Li, Ramprasaath Selvaraju, Akhilesh Gotmare, Shafiq Joty, Caiming Xiong, and Steven Chu Hong Hoi. Align before fuse: Vision and language representation learning with momentum distillation. *Advances in neural information processing systems*, 34:9694–9705, 2021. 3

[23] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35, 2023. 3

[24] Kaixin Ma, Filip Ilievski, Jonathan Francis, Eric Nyberg, and Alessandro Oltramari. Coalescing global and local information for procedural text understanding. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 1534–1545, 2022. 8

[25] Aman Madaan, Shuyan Zhou, Uri Alon, Yiming Yang, and Graham Neubig. Language models of code are few-shot

commonsense learners. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, 2022. 3, 8

[26] Javier Marın, Aritro Biswas, Ferda Ofli, Nicholas Hynes, Amaia Salvador, Yusuf Aytar, Ingmar Weber, and Antonio Torralba. Recipe1m+: A dataset for learning cross-modal embeddings for cooking recipes and food images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(1):187–203, 2021. 2, 5

[27] Niki Martinel, Gian Luca Foresti, and Christian Micheloni. Wide-slice residual networks for food recognition. In *2018 IEEE Winter Conference on applications of computer vision (WACV)*, pages 567–576. IEEE, 2018. 1, 2

[28] Mary Brighton. Tell Me What You Eat and I Will Tell You Who You Are. https://www.hackensackmeridianhealth.org/en/HealthU/2018/02/07/tell-me-what-you-eat-and-i-will-tell. Accessed on Feb 12, 2023. 1

[29] Mehrdad Farahani and Kartik Godawat and Haswanth Aekula and Deepak Pandian and Nicholas Broad. Chef Transformer. https://huggingface.co/flax-community/t5-recipe-generation. Accessed on April 12, 2023. 1, 5, 6

[30] Weiqing Min, Bing-Kun Bao, Shuhuan Mei, Yaohui Zhu, Yong Rui, and Shuqiang Jiang. You are what you eat: Exploring rich recipe information for cross-region food analysis. *IEEE Transactions on Multimedia*, 20(4):950–964, 2017. 1, 8

[31] Weiqing Min, Shuqiang Jiang, Linhu Liu, Yong Rui, and Ramesh Jain. A survey on food computing. *ACM Comput. Surv.*, 52(5), sep 2019. 1, 2

[32] Nadia A Najjar and David C Wilson. Computer Cooking Contest. https://ceur-ws.org/Vol-2028/XXCCC17_preface.pdf. Accessed on June 15, 2023. 7

[33] Dim P. Papadopoulos, Enrique Mora, Nadiia Chepurko, Kuan Wei Huang, Ferda Ofli, and Antonio Torralba. Learning program representations for food images and cooking recipes, 2022. 1, 3

[34] Parisa Pouladzadeh and Shervin Shirmohammadi. Mobile multi-food recognition using deep learning. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 13(3s):1–21, 2017. 2

[35] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 3

[36] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. 3

[37] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020. 2, 3, 5

[38] Rohan Taori and Ishaan Gulrajani and Tianyi Zhang and Yann Dubois and Xuechen Li and Carlos Guestrin and Percy Liang and Tatsunori B. Hashimoto. Alpaca: A Strong, Replicable Instruction-Following Model. https://crfm.stanford.edu/2023/03/13/alpaca.html. Accessed on June 21, 2023. 4

[39] Markus Rokicki, Christoph Trattner, and Eelco Herder. The impact of recipe features, social cues and demographics on estimating the healthiness of online recipes. In *Proceedings of the international AAAI conference on web and social media*, number 1, 2018. 2

[40] Md. Shafaat Jamil Rokon, Md Kishor Morol, Ishra Binte Hasan, A. M. Saif, and Rafid Hussain Khan. Food recipe recommendation based on ingredients detection using deep learning, 2022. 1

[41] Amaia Salvador, Michal Drozdzal, Xavier Giró-i Nieto, and Adriana Romero. Inverse cooking: Recipe generation from food images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10453–10462, 2019. 1, 2, 4, 5, 6

[42] 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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3020–3028, 2017. 5, 6

[43] Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A. Smith, and Yejin Choi. Atomic: An atlas of machine commonsense for if-then reasoning. In *AAAI Conference on Artificial Intelligence*, 2019. 8

[44] Tiago Simas, Michal Ficek, Albert Diaz-Guilera, Pere Obrador, and Pablo R Rodriguez. Food-bridging: a new network construction to unveil the principles of cooking. *Frontiers in ICT*, 4:14, 2017. 8

[45] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27, 2014. 3

[46] Sutter Health. Eating Well for Mental Health. https://www.sutterhealth.org/health/nutrition/eating-well-for-mental-health. Accessed on March 24, 2023. 1

[47] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. 4

[48] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 3

[49] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3156–3164, 2015. 2

[50] Hao Wang, Guosheng Lin, Steven CH Hoi, and Chunyan Miao. Structure-aware generation network for recipe generation from images. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVII 16*, pages 359–374. Springer, 2020. 2

[51] Hao Wang, Guosheng Lin, Steven C. H. Hoi, and Chunyan Miao. Learning structural representations for recipe generation and food retrieval. *CoRR*, abs/2110.01209, 2021. 1, 2

[52] Liping Wang, Qing Li, Na Li, Guozhu Dong, and Yu Yang. Substructure similarity measurement in chinese recipes. In *Proceedings of the 17th international conference on World Wide Web*, pages 979–988, 2008. 2

[53] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022. 3

[54] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed H Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*. 3

[55] Hui Wu, Michele Merler, Rosario Uceda-Sosa, and John R. Smith. Learning to make better mistakes: Semantics-aware visual food recognition. In *Proceedings of the 24th ACM International Conference on Multimedia*, MM '16, page 172–176, New York, NY, USA, 2016. Association for Computing Machinery. 2

[56] Haoran Xie, Lijuan Yu, and Qing Li. A hybrid semantic item model for recipe search by example. In *2010 IEEE International Symposium on Multimedia*, pages 254–259, 2010. 2

[57] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057. PMLR, 2015. 2

[58] Lewei Yao, Runhui Huang, Lu Hou, Guansong Lu, Minzhe Niu, Hang Xu, Xiaodan Liang, Zhenguo Li, Xin Jiang, and Chunjing Xu. Filip: fine-grained interactive language-image pre-training. *arXiv preprint arXiv:2111.07783*, 2021. 3

[59] Hana Yousuf, Michael Lahzi, Said A Salloum, and Khaled Shaalan. A systematic review on sequence-to-sequence learning with neural network and its models. *International Journal of Electrical & Computer Engineering (2088-8708)*, 11(3), 2021. 3

[60] Jiarui Zhang, Mahyar Khayatkhoei, Prateek Chhikara, and Filip Ilievski. Using visual cropping to enhance fine-detail question answering of blip-family models. *arXiv preprint arXiv:2306.00228*, 2023. 3

[61] Chunting Zhou, Graham Neubig, Jiatao Gu, Mona Diab, Francisco Guzmán, Luke Zettlemoyer, and Marjan Ghazvininejad. Detecting hallucinated content in conditional neural sequence generation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1393–1404, 2021. 5

[62] Yu-Xiao Zhu, Junming Huang, Zi-Ke Zhang, Qian-Ming Zhang, Tao Zhou, and Yong-Yeol Ahn. Geography and similarity of regional cuisines in china. *PloS one*, 8(11):e79161, 2013. 8

[63] Shuangquan Zuo, Yun Xiao, Xiaojun Chang, and Xuanhong Wang. Vision transformers for dense prediction: A survey. *Knowledge-Based Systems*, 253:109552, 2022. 4