Universiteit
Leiden

Master Computer Science

Cross-Modal Food Retrieval with Vision-Language Pre-training

Name:	Yaqiong Gao
Student ID:	s2478846
Date:	11/07/2023
Specialisation:	Data Science
1st supervisor:	Dr. Anna V. Kononova
2nd supervisor:	Dr. Suzan Verberne

Master's Thesis in Computer Science

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Abstract

Cross-modal food retrieval has gained a lot attention recently since the large-scale dataset Recipe1M [27] was released. Meanwhile, Vision-language Pre-training (VLP) is also in a fast developing phase, where it has been proved that can improve the performance of many vision-language downstream tasks. In addition, current studies on cross-modal food retrieval focus on the dual-stream approach, whereas VLP follows a single-stream approach. In this work, we investigate whether the single-stream approach can work for the Recipe1M dataset on the cross-modal retrieval task by fine-tuning two VLP models, Oscar [14] and ViLT [10], as well as training a recipe vision-language (RecipeVL) model from scratch following the architectures of the two VLP models. We use the latest work, H-Transformer [28], of the original authors who released the Recipe1M as our dual-stream comparative model. The experimental results show that the single-stream approach can produce comparable performance on the cross-modal food retrieval. Furthermore, we use a explainable framework, VL-CheckList [43], to evaluate our methods in three aspects, namely object, attribute, and relation. Our code is available at `https://github.com/chloeeegao/cross-modal-retrieval`.

Contents

List of Figures

List of Tables

1 Introduction

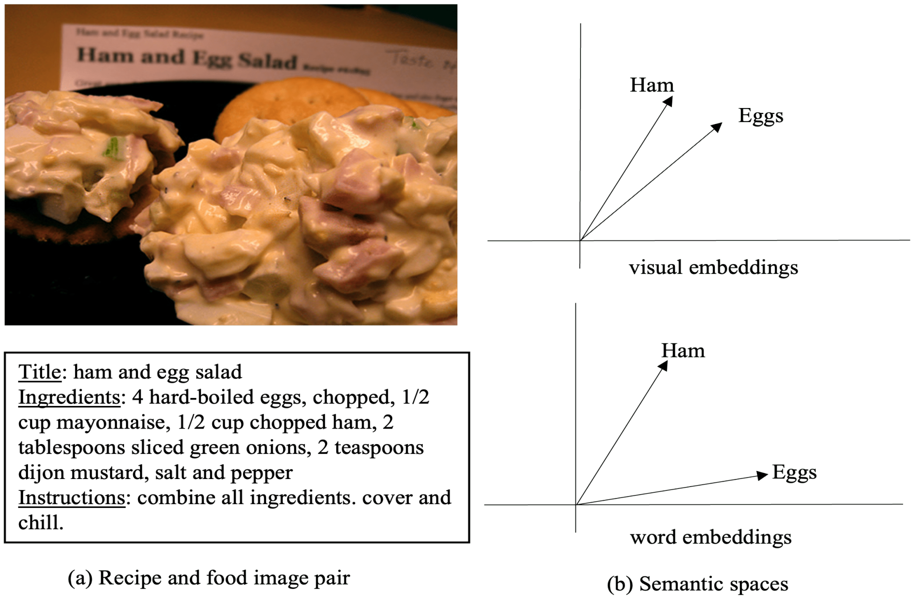
In the real world, human beings generally learn about objects through five senses including vision, hearing, smell, taste, and touch. In such ways, people can obtain comprehensive information about the same object from different aspects. It has been shown that about 80% to 85% of the information people obtain is from vision [23]. Visual information includes facial expressions, body language, text, visual symbols, color, diagrams, images, videos, and so on. Human beings can naturally integrate different information to learn about the same visual object. For instance, when we were kids, parents often use fruit flash cards which contain a fruit name and its corresponding image on each card to teach us recognize different fruits. After a period of learning, we can tell the fruit name based on its image or draw a outline of the fruit given a fruit name. This is a simple example of how human beings learn about objects from different aspects.

On the other hand, nowadays with the more accessible and faster network, multimedia information which describe the same events or topics, are growing fast on the Internet. For instance, a news website not only has textual descriptions about the headline but also has images or videos about it for a full demonstration [36]. Furthermore, smart devices, especially cellphones and tablets, have been widely used in our daily life, which has reshaped our lifestyles. People tend to post more multimedia information on social media and also prefer to obtain more multimedia information when searching online. It has been suggested that like humans, machines can also learn more comprehensive information about the same visual object from different aspects. All of above introduced a new challenge which is how to fuse information from different modalities, namely *multimodal representation learning*. Modality refers to the form of information presented[17], which typically includes text, audio, images, videos, etc. In this work, we will focus on *cross-modal representation learning* between text and images.

In the past decade, benefiting from the rapid development of deep learning, computer vision (CV) and natural language processing (NLP) have achieved high quality performance in many downstream tasks, such as object detection, image classification, sentiment analysis, and machine translation. However, each of these tasks only involves a single modality. As multimodal representation learning steadily draws more attention, many more complex tasks that involve multiple modalities, such as image captioning, text-to-image generation, visual question answering (VQA), natural language for visual reasoning (NLVR), cross-modal retrieval, and multimodal sentiment analysis, have been studied in the recent years. In this work, we will tackle the cross-modal retrieval task in the food domain using the Recipe1M dataset [27], in which cooking recipes are used as queries to retrieve food images or vice versa. The dataset contains more than 1 million cooking recipes and 800K food images.

1.1 Problem Statement

Nowadays, latent representation is widely used to extract information from raw data, which usually can preserve more useful information and also is more flexible to downstream tasks than traditional task-specific feature engineering when dealing with large-scale and noisy unstructured data [17]. As mentioned above, benefiting from current well-developed deep learning architectures, such as ResNets [6] and BERT [3], a more powerful unimodal



Title: ham and egg salad
Ingredients: 4 hard-boiled eggs, chopped, 1/2 cup mayonnaise, 1/2 cup chopped ham, 2 tablespoons sliced green onions, 2 teaspoons dijon mustard, salt and pepper
Instructions: combine all ingredients. cover and chill.

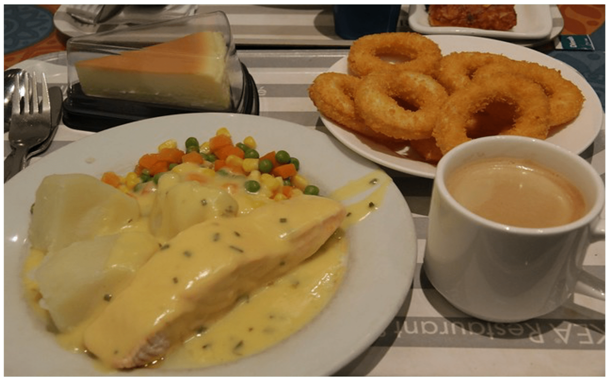
(a) Recipe and food image pair

(b) Semantic spaces

Figure 1: (a) An example from the Recipe1M [27] dataset. (b) Pseudo word vectors and visual vectors in the semantic space.

representation can improve performance on many downstream tasks. For instance, as BERT [3] was introduced, the performance of many NLP downstream tasks took a big leap, which also started a trend of utilizing pre-trained models.

Likewise, a more effective cross-modal representation will also boost the performance in vision-language downstream tasks. Since our work focuses on the cross-modal retrieval task and only involves text and image modality, the later mentioned cross-modal representation are all meant for these two modalities. There are three existing main challenges in cross-modal representation learning. The first one is the **semantic gap between modalities**, specifically different modalities often capture different aspects of information which leads to the semantic information preserved in each modality can be unbalanced. For instance, as shown in Figure 1(a), the cooking recipe contains quantity information about each ingredient but this information is not explicitly shown in the paired food image. On the other hand, the food image has some biscuits and a black plate in the background but those are not described in the paired cooking recipe. The second challenge is the **heterogeneity gap between modalities**, specifically different modalities have distinct data distributions and characteristics. For instance, as shown in Figure 1(b) 'ham' and 'eggs' are closer in the visual semantic space than in the text semantic space. The third challenge is the **lack of paired labelled data**, specifically cross-modal representation learning is naturally a weakly-supervised learning problem due to the lack of explicit alignments between modalities. As the example shown in Figure 1, the cooking recipe and the paired food image are manually made to be a pair but explicit alignments between words in text and corresponding regions in the paired image are not provided. Therefore, it requires high volumes of paired data for model training.



Caption: a plate with creamy chicken and vegetables, a side of onion rings, a cup of coffee and a slice of cheesecake.

Figure 2: An example from COCO [16] dataset.

With the availability of large image-text corpora, such as COCO [1] and Flickr30K [2], Vision-Language Pre-training (VLP) is becoming a prevalent topic in the field. Recent researches also show that VLP models, such as Oscar [14] and ViLT [10], have improved performance on many vision-language downstream tasks. In this work, we will fine-tune the two VLP models on the cross-modal retrieval task using the Recipe1M dataset. In addition, we also train a recipe vision-language (RecipeVL) from scratch following the architecture of the two VLP models.

1.2 Research Questions

The cooking recipes in the Recipe1M dataset are long text as shown in Figure 1, which often include more semantic information than the paired image as discussed in Section 1.1. Meanwhile, the text data in the image-text corpora VLP models are trained on are short image captions. For instance, the example in Figure 2 is from COCO, in which the image-text pair is also related to food but the textual description just simply lists the visible objects in the image. Thus in this manner, the typical text input length of VLP models ranges from 40 to 70 tokens [3]. In addition, current researches of cross-modal food retrieval focus on the dual-stream approach, but VLP models follow a single-stream approach. The details of the two approaches are presented in Section 2.1. Therefore, we propose to extract key information from recipes in order to reduce the semantic gap between modalities and also be able to fine-tune VLP models using their pre-trained weights. The main research questions of this work are as follows:

1. **Does information extraction from cooking recipes help narrow the semantic gap in cross-modal representation learning?**

2. **Can fine-tuning VLP models on the Recipe1M dataset help improve the performance of the cross-modal retrieval task?**

[1] https://cocodataset.org

[2] https://www.kaggle.com/datasets/hsankesara/flickr-image-dataset

[3] Tokens refer to the units into which text is divided. In VLP models, *BertTokenizer* is often used which is a subword-level tokenization. For example, word 'unhappiness' will be splitted into 'un' and 'happiness', which are identified as two subword tokens.

3. **Can single-stream approaches work better than dual-stream approaches on the Recipe1M dataset?**

1.3 Contributions

In summary, the main contributions of this work are as follows:

- We propose to extract key information from cooking recipes to ease the learning of semantic alignment. Specifically, we extract ingredients with corresponding cooking methods from recipe instructions using named entity recognition and dependency parsing techniques.

- We replicate Salvadaor et al.'s work [28] and retrain the H-Transformer using the extracted information from recipes.

- We fine-tune two VLP models, Oscar and ViLT, on the cross-modal retrieval task using the extracted information from recipes on a subset of the Recipe1M dataset.

- We train a recipe vision-language (RecipeVL) model from scratch following the architectures of the two VLP models. Our experimental results show that the single-stream approach can produce comparable performance to current researches of the dual-stream approach on the cross-modal food retrieval.

- We evaluate our methods using a recent explainable framework called VL-CheckList [43] for a more comprehensive understanding and also help to provide some insights for future improvements.

1.4 Thesis Structure

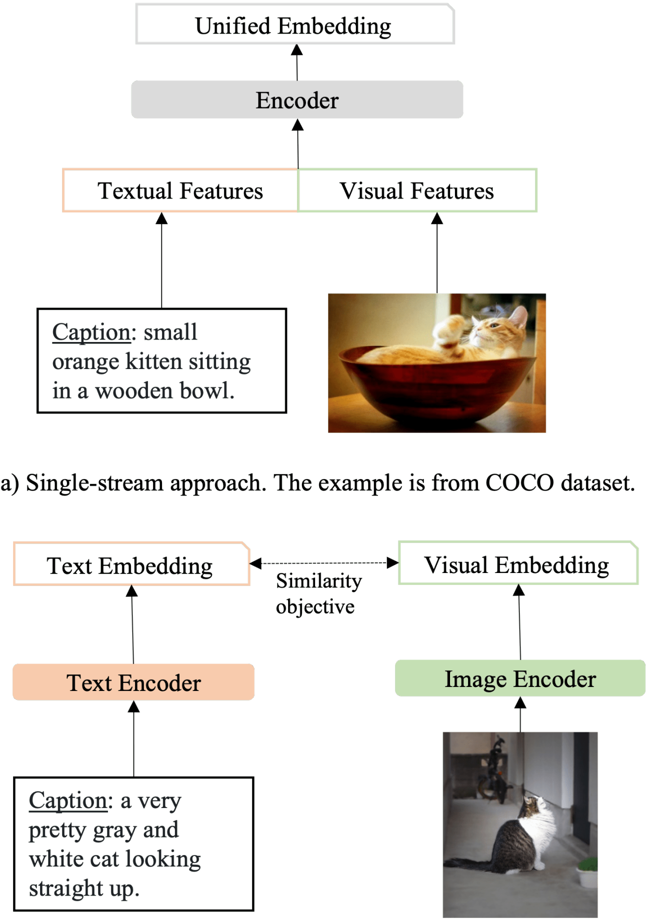
The remainder of this work is organized as follows: Section 2 introduces two approaches in cross-modal representation learning, related work on the Recipe1M dataset, and details about the two VLP models. Section 3 provides a preliminary analysis and statistics about the Recipe1M. Section 4 explains our proposed methods in details. Section 5 presents our experiments and results. Finally, we address our research questions and discuss the potential improvements along future work in Section 6 and conclude the thesis in Section 7.

2 Background and Related Work

In this section, we provide an overview of two common approaches in cross-modal representation learning and discuss related works on the Recipe1M dataset. We also introduce the details about the two VLP models, namely Object-Semantics Aligned Pre-training (Oscar) [14] and Vision-and-Language Transformer (ViLT) [10].

2.1 Cross-modal Representation Learning

Cross-modal representation learning aims to build embeddings using information from different modalities. Since our task is to learn the cross-modal representation from text and image modality, our survey mainly focuses on these two modalities while current researches also involve audio, video, and other modalities. Recent works involving text and image modality can be divided into two categories [17]: (1) fuse information from different modalities into unified embeddings, known as *single-stream approach.* (2) build embeddings from different modalities in a common semantic space based on similarity metrics, known as *dual-stream approach.* Figure 3 illustrates the basic architectures of the two approaches.



(a) Single-stream approach. The example is from COCO dataset.

(b) Dual-stream approach. The example is from COCO dataset.

Figure 3: Two cross-modal representation learning approaches.

Specifically, (1) in the single-stream approach, text and visual features are concatenated together as the input to the encoder. The parameters of the encoder are optimized through training objectives. The output of the encoder will be the unified embedding which fuses information from text and image modality. In Figure 3(a), the unified embedding will contain the information from 'small orange kitten sitting in a wooden bowl' and the paired image, which can be used for further vision-language downstream tasks. (2) in the dual-stream approach, text and visual features are fed into the text and image encoder separately. The outputs of the two encoders are projected into a common semantic space, where the most similar embeddings will be closer than others and the most dissimilar embeddings will be distant from others. The parameters of both encoders are optimized jointly through similarity objectives. In Figure 3(b), the text embedding of 'a very pretty gray and white cat looking straight up' and the visual embedding of the paired image should be close to each other in the common semantic space after training.

2.2 Related Work on Recipe1M Dataset

Recipe1M dataset was created and released by Salvador et al. [27] in 2017. The authors proposed to build cross-modal embeddings for cooking recipes and food images by jointly training two encoders (JE) based on the cosine similarity between two modalities in the common space with semantic regularization. Long-short term memory (LSTM) [7] and ResNet-50 [6] were used as text and image encoder in JE. This is the first work presenting the cross-modal retrieval task in the food domain. Chen et al. [2] argued that recipe search is different from other cross-modal retrieval task because it requires understanding the textual descriptions of cooking procedure to predict the possible consequence on visual appearance. The authors proposed to use attention mechanism to align the attended words and sentences in a recipe to their corresponding image features. Besides attention modeling on cooking recipes, on top of JE, Chen et al. used all three components (title, ingredients, instructions) in a recipe and also instead of using the penultimate layer of ResNet-50 , pool5 features are extracted for image representation. The cross-modal retrieval result was improved by 6% on Recall@1 (1K test set). Later, Wang et al. [35] proposed a novel framework, adversarial cross-modal embedding (ACME), in which three parts are added on top of JE: (1) a triplet loss with hard sampling to reduce the high variance in images, (2) an adversarial loss to align the feature distribution between modalities, (3) a cross-modal translation consistency loss to reduce the information loss. It achieved state-of-the-art (SOTA) performance on the Recipe1M dataset in 2019. Afterwards, Wang et al. [34] proposed a semantic-consistent and attention-based networks (SCAN), in which a semantic consistency loss was introduced to reduce the intra-class variance of food data representations along with a self-attention applied on LSTM to learn discriminative recipe features without images. It then outperformed previous mentioned models.

Besides above efforts on learning objectives, other works on better unimodal representation before cross-modal learning were also investigated at the same time. Cross-modal hierarchical embeddings for food domain (CHEF) proposed by Pham et al. [21] used a tree-structured LSTM as the text encoder to learn the complex functional and hierarchical relationships between images and text. CHEF can identify the main ingredients and cooking actions in the recipe without explicit supervision. Sugiyama and Yanai [32] argued that photos with different serving styles and different plates can be associated with the

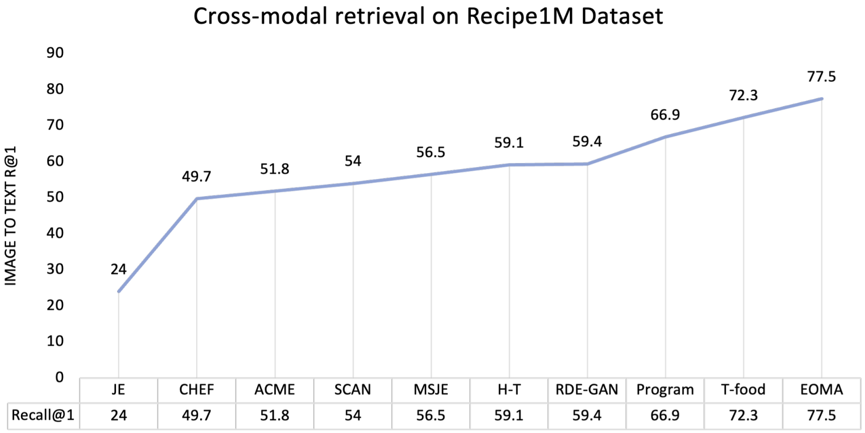


Figure 4: Recall@1 on 1K test set of recent studies on the Recipe1M dataset.

same recipe. They proposed a recipe disentangled embedding GAN [4] (RDE-GAN), in which images are disentangled into a recipe image embedding and a dish shape embedding. In this manner, the image embedding only contains recipe-related information. Xie et al. [40] proposed a multi-modal semantic enhanced joint embedding approach (MSJE), in which term-frequency and inverse document frequency (TF-IDF) features were extracted from the title, ingredients, and instructions to capture the significant key terms. Moreover, they concatenated the frequency feature to the recipe sequence feature learned from LSTM and TF-IDF enhanced category semantics were incorporated to the image feature learned from ResNet-50. These methods all achieved competitive results shown in Figure 4.

After Transformer [33] architecture has proved to be a efficient technique in the NLP field, studies of using it as the text encoder have been investigated extensively. The original authors who released the Recipe1M dataset proposed a hierarchical recipe Transformer (H-Transformer) [28] with a self-supervised loss to make a full use of the recipe-only data. It achieved SOTA results in 2021 with a simplified method compared to all aforementioned methods. Its ablation study also showed that utilizing vision Transformer (ViT) [11] as the image encoder can produce a better result than ResNet-50.

Furthermore, in this competitive field there are several novel works on the Recipe1M dataset published last year. Shukor et al. [31] proposed a new retrieval framework, Transformer decoders with multimodal regularization for cross-modal food retrieval (T-Food), which exploited the interaction between modalities along with a novel regularization. Plus, it used a VLP model, CLIP [25], as the image encoder. Papadopoulos et al. [20] proposed to represent cooking recipes and food images as cooking programs in order to capture cooking semantics and sequential relationships between actions. Specifically, besides a text and image encoder, the model also jointly generated programs as sequence of commands using a program decoder conditioned on the image and text features. Xie et al. [39] argued that cooking recipes contain the descriptions of numerous events while such event sequence information is lacking in the image, thereby it is a challenging to learn the fine-grained alignment. They proposed a novel framework, event-oriented modality alignment (EOMA), in which the significance of each event was captured and combined with the identified key event elements to learn the discriminative text embeddings for recipes and the image embeddings were enhanced by the shared event tags. EOMA is the current SOTA model on the cross-modal retrieval task using the Recipe1M dataset.

In conclusion, all aforementioned methods are dual-stream approaches. As we can see in Figure 4, the performance of cross-modal retrieval task on the Recipe1M dataset has been improved a lot since JE was introduced. However, the single-stream approach has never been investigated on the Recipe1M dataset, to the best of our knowledge.

2.3 Vision-Language Pre-training

In this section, two VLP models, Object-Semantics Aligned Pre-training (Oscar) for vision-language tasks [14] and Vision-and-Language Transformer (ViLT) without convolution or region supervision [10], are introduced in details along with two commonly used pre-training objectives.

2.3.1 Pre-training Objectives

Oscar and ViLT both are single-stream approaches, in which text features and image features are concatenated together to feed into the encoder and the output embedding is the unified embedding which fuses information from the text and image modality. In this approach, masked language modeling (MLM) and image text matching (ITM) are the commonly used objectives.

Masked Language Modeling This is similar to the masked language modeling in BERT [3] that the masked tokens in text can be recovered from its surroundings. In cross-modal setting, the masked text tokens also leverage the information in the image context. The masked text tokens are randomly chosen with the probability of 0.15. The objective is to predict the masked text tokens t_{masked} from its contextualized text tokens $\boldsymbol{t}$ also plus all image features tokens $\boldsymbol{v}$ by minimizing the negative log-likelihood. The MLM loss is as follow:

$$L_{MLM} = -E_{(t,v)} \log_p(t_{masked}|\boldsymbol{t}, \boldsymbol{v}) \tag{1}$$

Image Text Matching The output of the special token [CLS] contains the fused vision-language information. The fused embedding is projected through a single linear layer over binary class considered as a classifier $f(\cdot)$ to predict whether the input text $\boldsymbol{t}$ and image $\boldsymbol{v}$ is a pair (y=1) or not (y=0). The paired image is randomly replaced by another sample from the dataset with the probability of 0.5. The ITM loss is defined as:

$$L_{ITM} = -E_{(t,v)} \log_p(y|f(\boldsymbol{t}, \boldsymbol{v})) \tag{2}$$

2.3.2 Oscar

Oscar was released in July, 2020 by Microsoft Corporation. It was pre-trained on a huge public corpus consisting of 6.5 million image-text pairs and fine-tuned on six vision-language downstream tasks including image-text retrieval, image captioning, novel object

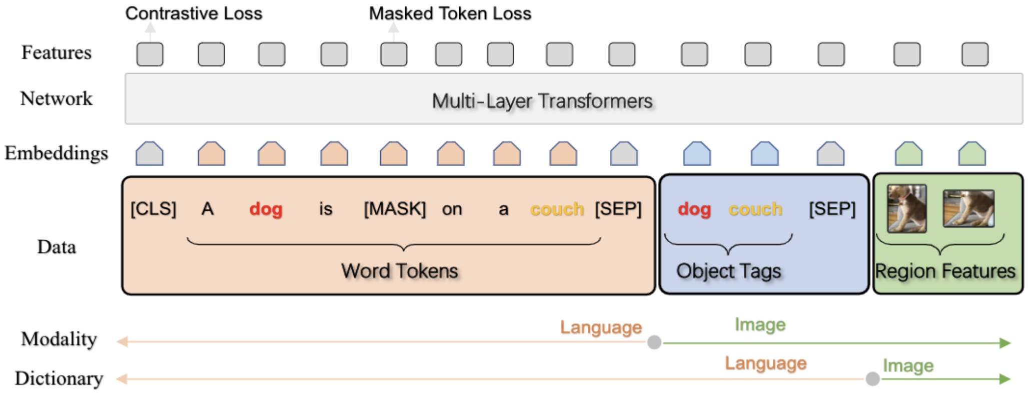


Figure 5: Illustration of Oscar from [14]. The input triple consists of word tokens $\boldsymbol{w}$, object tags $\boldsymbol{q}$, and region features $\boldsymbol{v}$. The input can be understood from two perspectives: modality view and dictionary view.

captioning (NoCaps), visual question answering (VQA), visual reasoning and compositional question answering (GQA), and natural language visual reasoning for real (NLVR2). The statistics of its pre-training corpus are shown in Table 1. Unlike other VLP models which used image-text pairs as inputs, the input of Oscar is a text-tag-image triple. The authors proposed that object tags detected in images can be used as anchor points to ease the learning of semantic alignments. The idea was motivated by the observation that the prominent objects in an image can be accurately detected and also mentioned in the paired text.

Figure 5 illustrates the overview of Oscar. The input triple consists of word tokens $\boldsymbol{w}$, object tags $\boldsymbol{q}$, and region features $\boldsymbol{v}$, where $\boldsymbol{w}$ represents the sequence of word embeddings of the text, $\boldsymbol{q}$ is the word embeddings of object tags detected from the image, and $\boldsymbol{v}$ represents the set of region vectors of the image. Specifically, the object tags $\boldsymbol{q}$ and region features $\boldsymbol{v}$ are generated by Faster R-CNN [26]. Each region extracted from an image denotes as $(\boldsymbol{v'}, \boldsymbol{z})$, where $\boldsymbol{v'} \in \mathbb{R}^P$ is a P-dimensional vector and $\boldsymbol{z}$ is the R-dimensional region position vector (*i.e.*, R =4 or 6). $\boldsymbol{v'}$ and $\boldsymbol{z}$ are concatenated to form the position-

Data Source	# images	# captions	Oscar	ViLT
COCO[16]	113K	567K	✓	✓
Flicker30k [42] (train)	29K	145K	✓	
SBU [19]	867K	867K	✓	✓
VQA [5] (train)	83K	444K	✓	
GQA [9] (bal-train)	79K	1026K	✓	
VG-QA [12]	48K	484K	✓	
VG [12]	108K	5.41M		✓
GCC [29]	3.01M	3.01M	✓	✓
Total (image/text)			4.1M/6.5M	4.1M/9.8M

Table 1: Statistics of pre-training corpus in Oscar and ViLT.

sensitive region feature vector and then projected through a linear layer to have the same dimension as word embeddings. The word embeddings of $\boldsymbol{w}$ and $\boldsymbol{q}$ are initialized using pre-trained BERT. From Figure 5 we can see that the input can be understood from two perspectives: (1) dictionary view: $\boldsymbol{w}$ and $\boldsymbol{q}$ share the linguistic semantic space while $\boldsymbol{v}$ lies in the visual semantic space; (2) modality view: $\boldsymbol{q}$ and $\boldsymbol{v}$ are extracted from the image modality and $\boldsymbol{w}$ is considered as text modality. In addition, the masked token loss (MLM loss) is used for dictionary view where word tokens in ($\boldsymbol{w}$, $\boldsymbol{q}$) are randomly replaced with the probability of 0.15. The contrastive loss (ITM loss) is used for modality view where ($\boldsymbol{q}$, $\boldsymbol{v}$) is replaced by a different tag sequence and corresponding image sampled from the corpus with the probability of 0.5. The full pre-training objectives of Oscar is $L_{pre-training} = L_{MLM} + L_{ITM}$. Oscar is initialized with parameters $\boldsymbol{\theta}_{BERT}$ and the position-sensitive region features are linear projected via matrix $\boldsymbol{W}$. Therefore, the trainable parameters are $\boldsymbol{\theta} = \{\boldsymbol{\theta}_{BERT}, \boldsymbol{W}\}$. In fine-tuning phase, image-text retrieval task is considered as a binary classification problem, in which the final fused embedding [CLS] is used to predict whether the given pair is a true pair or not. The probability score is used to rank the image-text pairs given a query.

2.3.3 ViLT

ViLT was introduced by Kim et al. in 2021. They argued that current VLP models heavily rely on the image feature extraction process which requires much more computation than the multimodal interaction steps; and the predefined visual vocabulary limited the upper bound of the expressive power of the visual encoder. Therefore, they proposed a minimal VLP model without convolution or region supervision which was up to tens of times faster than other VLP models while also can have competitive performance on the downstream tasks. Figure 6 illustrates ViLT model overview. It was pre-trained on 9.8 million image-text pairs and the statistics of its pre-training corpus are shown in Table 1.

In ViLT, the visual features are extracted from a pre-trained ViT. Specifically, images are sliced into patches and transformed to patch embeddings, which drastically simplifies the visual embedding step to the level of textual embedding. The patch size of 32 is used in ViLT, in such manner a 224 x 224 image is sliced into (224/32) x (224/32) = 7 x 7 patches and so the visual features consist of 49 patch tokens plus one class token. Instead of initializing from pre-trained BERT, the text features are learned from scratch because they argued the pre-trained model for single modality does not guarantee performance

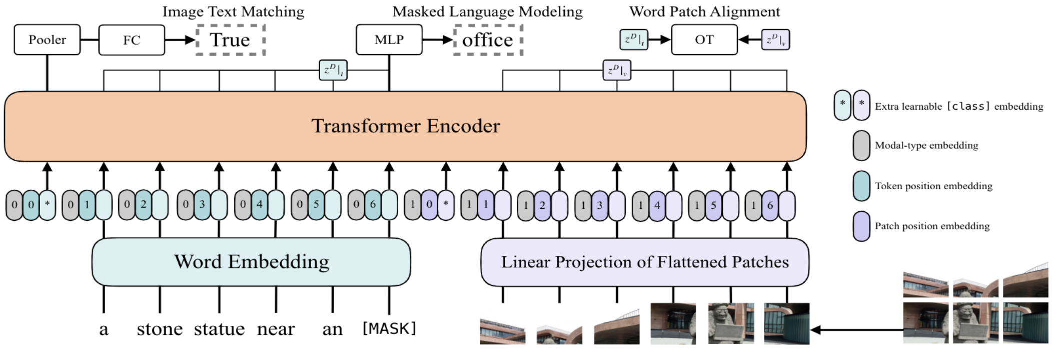


Figure 6: Illustration of ViLT from [10].

gain for vision and language tasks. Moreover, the interaction transformer weights are initialized with parameters from pre-trained ViT. They stated that such initialization can exploit the power of the cross-modal interaction layers to process visual features. ViLT also uses the two commonly used pre-training objectives described above but it adds two components to enhance the cross-modal learning. The first component is word patch alignment (WPA) which computes the alignment score between textual subset and visual subset using the inexact proximal point method for optimal transports (IPOT). The approximate wasserstein distance is mutltiplied by 0.1 and added to the ITM loss. The second component is whole word masking which means all consecutive subword tokens of one word will be masked. The authors argued that if not all tokens of a whole word are masked, the model might only rely on the nearby unmasked subword tokens to predict the masked subword token rather than using the information from image context. In fine-tuning phase, ViLT processes the image-text retrieval task as a binary classification. Specifically, 15 negative texts were randomly sampled from the corpus and the model was fine-tuned with cross-entropy loss that maximized the scores on positive pairs. In addition, the similarity score head used during fine-tuning is initialized from the pre-trained ITM head of ViLT.

3 Data

Recipe1M [27] is a large-scale corpus of recipe data which contains over 1 million cooking recipes and 800 thousands food images. The data were originally scraped from over 24 popular cooking websites. Specifically, relevant text was extracted from the raw HTML and the linked images were downloaded from the same page. Each recipe consists of 3 components which are the title, a list of ingredients used for the dish, and a sequence of instructions about how to prepare the dish. The recipe text is provided as free text. The images are provided as RGB in JPEG format. The dataset [4] was partitioned into training, validation, and test sets, and its statistics are shown in Table 2. 70% of the data is training set and the rest is split equally into validation and test set.

Partition	# recipe	# images
Training	720,639	619,508
Validation	155,036	133,860
Test	154,045	134,338
Total	1,029,720	887,706

Table 2: Number of samples in training, validation and test sets.

3.1 Preliminary Analysis

According to the original paper [27], the dataset has approximately 0.4% duplicate recipes and 2% duplicate images. The exact duplicates or recipes that shared the same image are removed. In addition, due to the nature of the data sources, not all recipes in the dataset have images. There are approximate 34% recipes have associated images, which means there are over 340K image-text pairs. The statistics of the recipes with images are shown in Table 3. Since models for cross-modal learning are trained on image-text pairs, that portion of the dataset will be used in this work.

Partition	# recipe	# images
Training	238,408	471,475
Validation	51,119	100,808
Test	51,304	100,297
Total	340,831	672,580

Table 3: Number of recipes have images and number of images per partition.

Figure 7 (a) and (b) display the distribution of number of ingredients and instructions per recipe, respectively. On average, a recipe comprises 9 ingredients and 10 instructions. In addition, the majority of recipes just have one associated image that illustrates the final look of the dish. From Figure 7 (c), we can see that there are about 290K recipes associated with one single image while a few recipes have more than one associated image that demonstrates each cooking step in the instructions.

[4] http://im2recipe.csail.mit.edu/dataset/download/

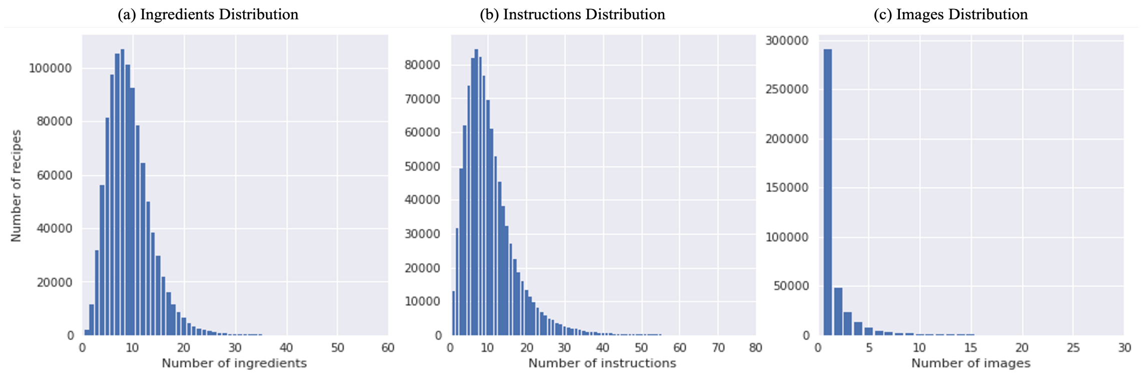


Figure 7: Distribution of the number of ingredients, instructions, and images per recipe.

Furthermore, in order to analyze the text length of cooking recipes we also count the number of words including punctuation in each component. The statistics are shown in Table 4. On average, in a recipe: the title component has 4 words; the ingredient component has 53 words; the instruction component has 133 words. In total, the average number of words in a recipe is 191 words. As we can see in Table 4 there are also some outliers which contain more than 1K words. Such samples usually include sub-recipes for the dishes.

Component	Median	Mean	Min	Max
Title	4	4	1	36
Ingredient	58	53	1	766
Instruction	108	133	1	3016
Total	163	191	3	3260

Table 4: Statistics for the number of words per recipe component.

4 Methods

In this section, we first present our proposed method for extracting ingredients and corresponding cooking methods from recipes. And then, we explain how to use the extracted information as inputs to retrain H-Transformer [28]. Then, we explain how to fine-tune the two VLP models, Oscar [14] and ViLT [10], using the extracted information on the cross-modal retrieval task. At last, we present the recipe vision-language (RecipeVL) model following the architectures of the two VLP models.

4.1 Information Extraction

As we discussed in the introduction, cooking recipes are long text which normally contain more semantic information than food images. Therefore, in order to narrow the semantic gap between two modalities, we propose to extract key information from recipes. On the other hand, it is also for the purpose of using pre-trained weights of VLP models to fine-tune on the Recipe1M dataset. Specifically, food ingredients and corresponding cooking methods are extracted from the instruction component using named entity recognition (NER) and dependency parsing techniques. Both techniques are implemented in *spaCy* [5].

The instruction component of a recipe is a sequence of sentences describing the steps about how to prepare a dish as shown in Figure 9. There are two reasons we want to extract food ingredients and cooking methods as key information of a recipe: 1) the ingredients are the main objects can be seen in the food images. In the Recipe1M, the paired images are usually the final look of the dish. 2) cooking methods applied to the ingredients can alter the their appearances in the finished dish. For instance, the final appearances of scrambled eggs and fried eggs are different.

4.1.1 Named Entity Recognition

We train a NER model to identify 'food' entities in the instructions by following the notebook [6] shared by Isaac Aderogba on DeepNote. It used the food data from the USDA's Branded Food's dataset[7], which contains 42,018 food names after filtering out names more than 3 words. Since the distribution of one-worded, two-worded, and three-worded food entities is not even, the dataset is further filtered to contain 45% for 1-worded foods, 30% for two-worded foods, and 25% for three-worded foods, which results in 3,311 food entities in total. The training data consists of 501 food entities [8] from the filtered results plus 436 randomly chosen sample text from *nltk*'s article corpus [9]. The remaining food entities [10] is the test data used for evaluation. After training, we obtain a food NER with 96.23% accuracy, which is the average over the three types food entities. The NER can identify commonly used cooking tools, such as pan or skillet, and it also can identify some states of ingredients, such as brown or tender.

[5] https://spacy.io/

[6] https://deepnote.com/@isaac-aderogba/Spacy-Food-Entities-2cc2d19c-c3ac-4321-8853-0bcf2ef565b3

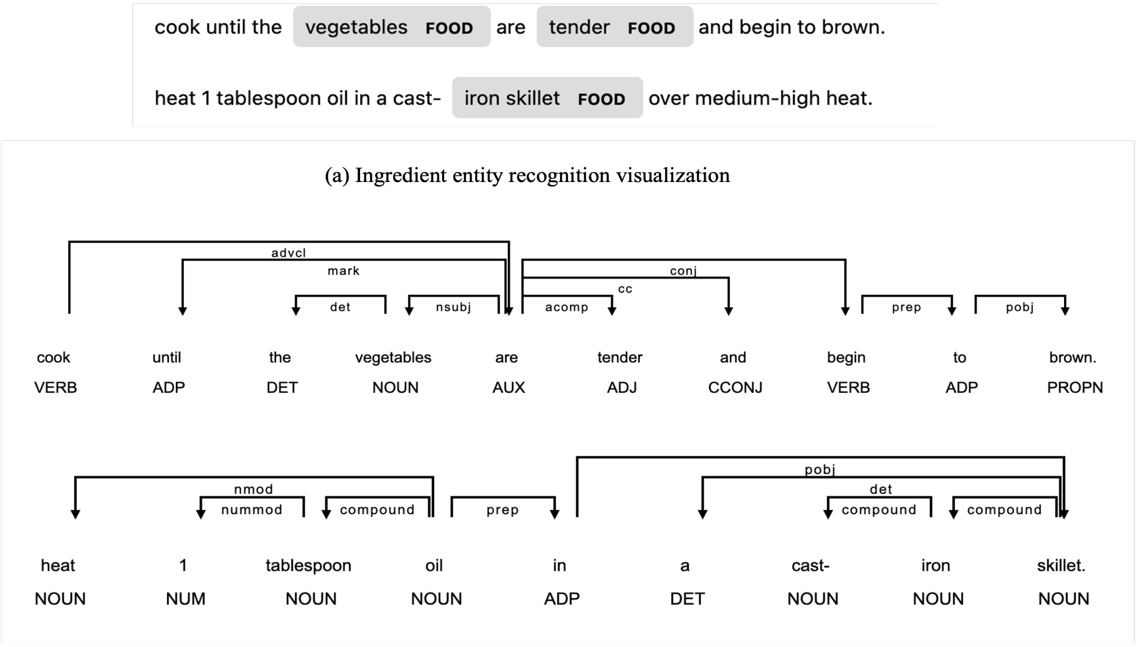
[7] https://fdc.nal.usda.gov/download-datasets

[8] One-worded, two-worded, three-worded foods equally have 167 entities respectively.

[9] https://www.nltk.org/api/nltk.corpus

[10] 1036 for one-worded foods, 268 for two-worded foods, and 245 for three-worded foods.

In addition, we also use the extracted ingredients from the ingredient component to compile a pattern-ruled ingredient NER. This ingredient extraction procedure (e.g. '1/2 teaspoons pepper' → 'pepper') was done in the first work of Salvador's [27] with 99.5% accuracy. Finally, we apply the two food and ingredient NERs to each sentence in the instructions so words tagged FOOD or ING are extracted as food ingredient entities. As shown in the Figure 8(a), 'oil', 'iron skillet', 'vegetables', and 'tender' are extracted.



(a) Ingredient entity recognition visualization

(b) Dependency parse visualization

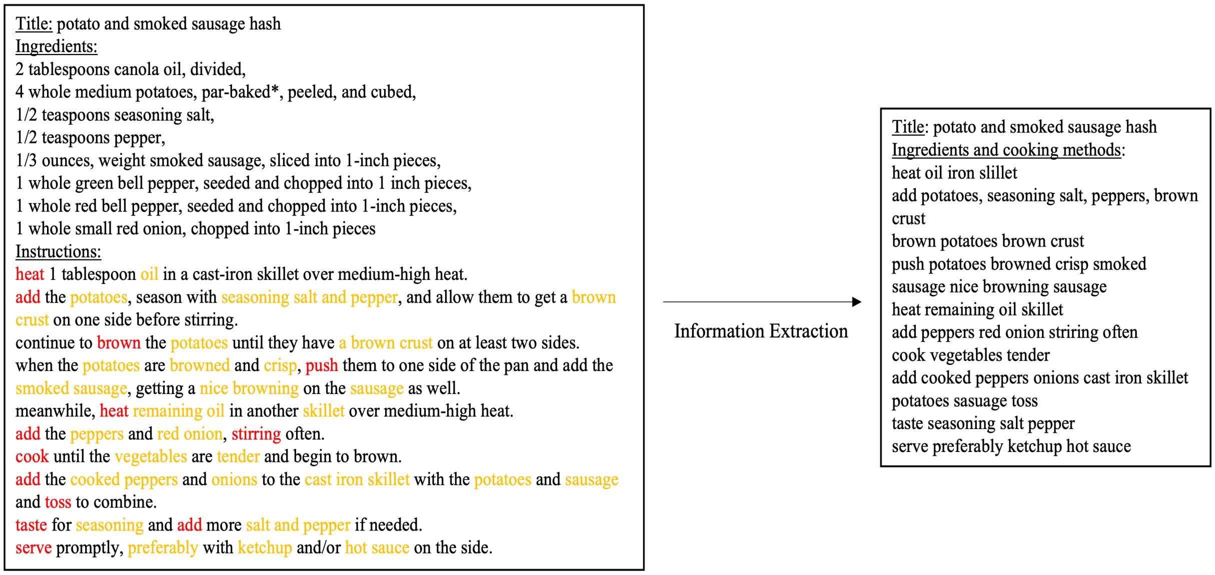
Figure 8: Visualization of (a) named entity recognition and (b) dependency parsing. The example is the first and eighth sentence in the instructions shown in Figure 9.

4.1.2 Dependency Parsing

Since our data is only related to recipes and in order to improve the extraction accuracy, we first manually create a cooking verb list consisting of 154 verbs shown in Table 13 that are widely used in the food domain. Then, we apply spaCy's dependency parser [11] to each sentence in the instructions. As we observe, the cooking methods are usually the root word in the sentence or have a compound or modifier relationship with labelled food ingredient entities. Hence, in each sentence of the instructions if the word's dependency role is root, or compound, or modifier, and meanwhile its lemmatization is also in the cooking verb list then the word is extracted as cooking method for the corresponding entity. As shown in Figure 8(b), although 'heat' was tagged as noun but it is the modifier for 'oil' and also in the defined verb list, so extracted as cooking method; 'cook' is the root word and in the list, so extracted. The extracted information for the two sentences are 'heat oil iron skillet' and 'cook vegetables tender'.

Our proposed information extraction procedure is a rule-based approach because there is no annotations for cooking methods or food ingredients in the Recipe1M dataset. The

[11] The spaCy version we use is 2.3.9 and the parser is a variant of the non-monotonic arc-eager transition-system described by Honnibal el at. [8].



Title: potato and smoked sausage hash
Ingredients:
2 tablespoons canola oil, divided,
4 whole medium potatoes, par-baked*, peeled, and cubed,
1/2 teaspoons seasoning salt,
1/2 teaspoons pepper,
1/3 ounces, weight smoked sausage, sliced into 1-inch pieces,
1 whole green bell pepper, seeded and chopped into 1 inch pieces,
1 whole red bell pepper, seeded and chopped into 1-inch pieces,
1 whole small red onion, chopped into 1-inch pieces
Instructions:
heat 1 tablespoon oil in a cast-iron skillet over medium-high heat.
add the potatoes, season with seasoning salt and pepper, and allow them to get a brown crust on one side before stirring.
continue to brown the potatoes until they have a brown crust on at least two sides.
when the potatoes are browned and crisp, push them to one side of the pan and add the smoked sausage, getting a nice browning on the sausage as well.
meanwhile, heat remaining oil in another skillet over medium-high heat.
add the peppers and red onion, stirring often.
cook until the vegetables are tender and begin to brown.
add the cooked peppers and onions to the cast iron skillet with the potatoes and sausage and toss to combine.
taste for seasoning and add more salt and pepper if needed.
serve promptly, preferably with ketchup and/or hot sauce on the side.

Title: potato and smoked sausage hash
Ingredients and cooking methods:
heat oil iron slillet
add potatoes, seasoning salt, peppers, brown crust
brown potatoes brown crust
push potatoes browned crisp smoked sausage nice browning sausage
heat remaining oil skillet
add peppers red onion stirring often
cook vegetables tender
add cooked peppers onions cast iron skillet potatoes sasuage toss
taste seasoning salt pepper
serve preferably ketchup hot sauce

Figure 9: Information extraction from recipe instructions. The text in red color are cooking methods and the text in yellow color are ingredient entities.

example in Figure 9 demonstrates after extracting the recipe can be represented using its title and ingredients with corresponding cooking methods. The recipe title is considered as the shared mutual information between text and image modality so it is kept . The example recipe originally consists of 285 words including punctuation, after extracting the recipe only contains 57 words but also keeps the key information about how to make the dish. It is worth to mention the order of instructions is also kept since some instructions might have sequential relations.

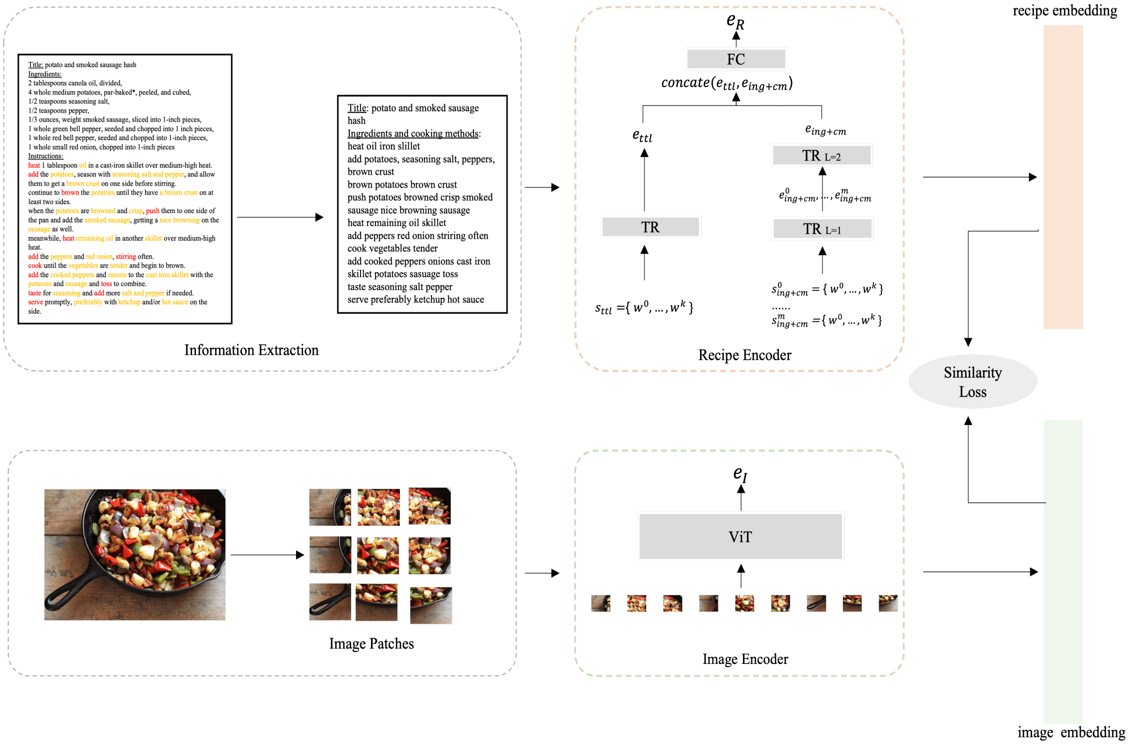
4.2 H-Transformer

We replicated Hierarchical Transformer (H-Transformer) [28] with the released code [12]. In order to explore the expressive power of the extracted information from recipes and also for later comparison to VLP models, we retrain H-Transformer with the extracted information. The model overview is shown in Figure 10. It is a dual-stream approach that recipe and image embeddings are learned separately from a text and image encoder, and the training process is end-to-end. In this approach, recipe and image embeddings are projected into the common semantic space and the parameters of two encoders are optimized through a similarity objective.

4.2.1 Recipe Encoder

A recipe x_R has 3 components: title r_{ttl}, ingredients r_{ing}, and instructions r_{ins}. In terms of representation, recipe title is a single sentence so $r_{ttl} = s_{ttl} = (w^0, ..., w^k)$ while both ingredients and instructions are lists of sentences so $r_{ing} = (s^0_{ing}, ..., s^n_{ing})$ and $r_{ins} = (s^0_{ins}, ..., s^m_{ins})$. In H-Transformer, title r_{ttl} is encoded with a Transformer TR so $e_{ttl} = TR(r_{ttl})$ is the average of the outputs of the Transformer at the last layer. Sentences in ingredients and instructions are first encoded with a $TR_{L=1}$ to obtain a sentence-

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



Title: potato and smoked sausage hash
Ingredients:
2 tablespoons canola oil, divided,
4 whole medium potatoes, par-baked*, peeled, and cubed,
1/2 teaspoons seasoning salt,
1/2 teaspoons pepper,
1/3 ounces, weight smoked sausage, sliced into 1-inch pieces,
1 whole green bell pepper, seeded and chopped into 1 inch pieces,
1 whole red bell pepper, seeded and chopped into 1-inch pieces,
1 whole small red onion, chopped into 1-inch pieces
Instructions:
heat 1 tablespoon oil in a cast-iron skillet over medium-high heat.
add the potatoes, season with seasoning salt and pepper, and allow them to get a brown crust on one side before stirring.
continue to brown the potatoes until they have a brown crust on at least two sides.
when the potatoes are browned and crisp, push them to one side of the pan and add the smoked sausage, getting a nice browning on the sausage as well.
meanwhile, heat remaining oil in another skillet over medium-high heat.
add the peppers and red onion, stirring often.
cook until the vegetables are tender and begin to brown.
add the cooked peppers and onions to the cast iron skillet with the potatoes and sausage and toss to combine.
taste for seasoning and add more salt and pepper if needed.
serve promptly, preferably with ketchup and/or hot sauce on the side.

Title: potato and smoked sausage hash
Ingredients and cooking methods:
heat oil iron slillet
add potatoes, seasoning salt, peppers, brown crust
brown potatoes brown crust
push potatoes browned crisp smoked sausage nice browning sausage
heat remaining oil skillet
add peppers red onion striring often
cook vegetables tender
add cooked peppers onions cast iron skillet potatoes sasuage toss
taste seasoning salt pepper
serve preferably ketchup hot sauce

Information Extraction
e_R
FC
$concate(e_{ttl}, e_{ing+cm})$
e_{ttl}
e_{ing+cm}
TR L=2
$e^0_{ing+cm}, \dots, e^m_{ing+cm}$
TR
TR L=1
$s_{ttl} = \{ w^0, \dots, w^k \}$
$s^0_{ing+cm} = \{ w^0, \dots, w^k \}$
......
$s^m_{ing+cm} = \{ w^0, \dots, w^k \}$
Recipe Encoder
recipe embedding
Similarity Loss
Image Patches
e_I
ViT
Image Encoder
image embedding

Figure 10: Illustration of H-Transformer using the extracted information. The image-text pair example is from the Recipe1M dataset.

level embedding for each sentence in ingredients and instructions respectively, and then those embeddings are encoded with a second $TR_{L=2}$ to obtain a single embedding for ingredients and instructions respectively. This is so-called Hierarchical Transformer HTR, in which each level TR has the same architecture (2 layers, 4 heads, D=512) but different parameters. Hence, $e_{ing} = HTR(r_{ing})$ and $e_{ins} = HTR(r_{ins})$. The recipe embedding $e_R = FC(concat(e_{ttl}, e_{ing}, e_{ins}))$ with D = 1024.

In our case, we use recipe title r_{ttl} and the extracted information r_{ing+cm} (ingredients along with corresponding cooking methods) as input as shown in Figure 10 so that recipe embedding $e_R = FC(concat(e_{ttl}, e_{ing+cm}))$ with D = 1024 where $e_{ttl} = TR(r_{ttl})$ and $e_{ing+cm} = HTR(r_{ing+cm})$.

4.2.2 Image Encoder

In the ablation study of [28], it was shown that Vision Transformer (ViT) has a relatively better performance than previous commonly used ResNet-50. Therefore, in our work we use a pre-trained ViT [13] as the image encoder to learn image embeddings e_I. The output of the last layer before the classifier layer in ViT is projected to the common space through a single linear layer. The dimension of image embedding e_I is set to 1024.

[13] This ViT is vit-base-patch16-224 and pre-trained on the ImageNet-21k dataset.

4.2.3 Similarity Loss

A bi-directional triplet hinge-loss objective is used in H-Transformer as shown in Equation 4. The main component of the loss function is L_{cos} as shown in Equation 3, where a, p, and n denote anchor, positive, and negative samples, $c(\cdot)$ denotes cosine similarity, and m is the margin (empirically set to 0.3). In Equation 4, e_R and e_I denotes recipe embedding and image embedding; $n = i$ means the same sample from different modalities (e.g. recipe and image embeddings of the image-text pair) and $n = j$ means different samples. In addition, $\delta(i, j) = 1$ if $i \neq j$ otherwise 0. Hence, ($e_R^{n=i}$, $e_I^{n=i}$) is a positive pair, and ($e_R^{n=i}$, $e_I^{n=j}$) and ($e_I^{n=i}$, $e_R^{n=j}$) are negative pairs. In training process, for a batch size B, the loss for one sample i is the average of all losses considering all other samples in the batch as negatives.

$$L_{cos}(a,p,n) = max(0, c(a,n) - c(a,p) + m) \tag{3}$$

$$L_{bi} = \frac{1}{B}\sum_{j=0}^{B}(L_{cos}(e_R^{n=i}, e_I^{n=i}, e_I^{n=j}) + L_{cos}(e_I^{n=i}, e_R^{n=i}, e_R^{n=j}))\delta(i,j) \tag{4}$$

4.3 Fine-tuning VLP Models

As we discussed in Section 1.2, the pre-trained weights of VLP models are not suitable for lengthy text such as recipes. Therefore, we use the extracted key information from recipes as text input. In this section, we explain how to fine-tune Oscar and ViLT on the cross-modal retrieval task. Both models are introduced in Section 2.3.

4.3.1 Oscar

The input of Oscar is a triplet consisting of word tokens w, object tags q, and image features v. The sequence length of discrete tokens (w, q) and $\boldsymbol{v}$ are 70, and 50 tokens, respectively. The image features used in Oscar are object region features extracted from Faster R-CNN. The object tags are also detected from Faster R-CNN. However, in our case there is no labelled image regions in the Recipe1M dataset to train a object detection model. Therefore, we use the pre-trained ViT [14] to extract image features, which is the same as in H-Transformer. In this manner, images are sliced into image patches and then flatted and linearly projected to patch embeddings v before feeding into Oscar. The word tokens and object tags (w, q) are word embeddings from pre-trained BERT. In our case, we use the extracted recipes as word tokens and consider main ingredients and food category as object tags. Following the motivation in Oscar, the main ingredients are explicitly visible objects in food images. And food categories are the shared mutual information between text and image modality. Following the image-text retrieval downstream task in Oscar, we fine-tuned it on the Recipe1M through the ITM objective by randomly select a different image/text sample from the dataset with 0.5 probability.

Main Ingredients We consider the visible ingredients in food images as the main ingredients. Generally, ingredients used for seasoning or flavour, such as salt, sugar, spices,

[14] This ViT is vit-base-patch32-224 and pre-trained on the ImageNet-21k dataset.

salt	sugar	vinegar	pepper
sauce	cumin	spice	oregano
flakes	nutmeg	cayenne	cinnamon
ginger	paprika	rub	blend
tamari	mesquite	seasoning	flour
powder	butter	rosemary	thyme
cloves	garlic	hickory	hanout

Table 5: Invisible ingredients list

or some sauces, are very hard to be seen in a food image. We define them as invisible ingredients and we manually create an invisible ingredients list shown in Table 5. Therefore, the ingredients not in this list are all considered as main ingredients and used as object tags.

Food Category In the early work of cross-modal food retrieval, food category was used as regularization to align the semantic gap between modalities because it is a mutual information shared by cooking recipes and food images. We use the food category generated from [27]. The authors first assigned Food-101 [1] categories to recipes that contain them in the title. And then the top 2,000 most frequent bigrams generated from the recipe titles in the training set are used to assign food category to which recipe contains the most frequent bigrams in the title. There are 1,047 categories obtained, which cover over 50% of the image-text pairs in the dataset. The remaining recipes without a category are assigned to a *background* class. The food category is used as object tags.

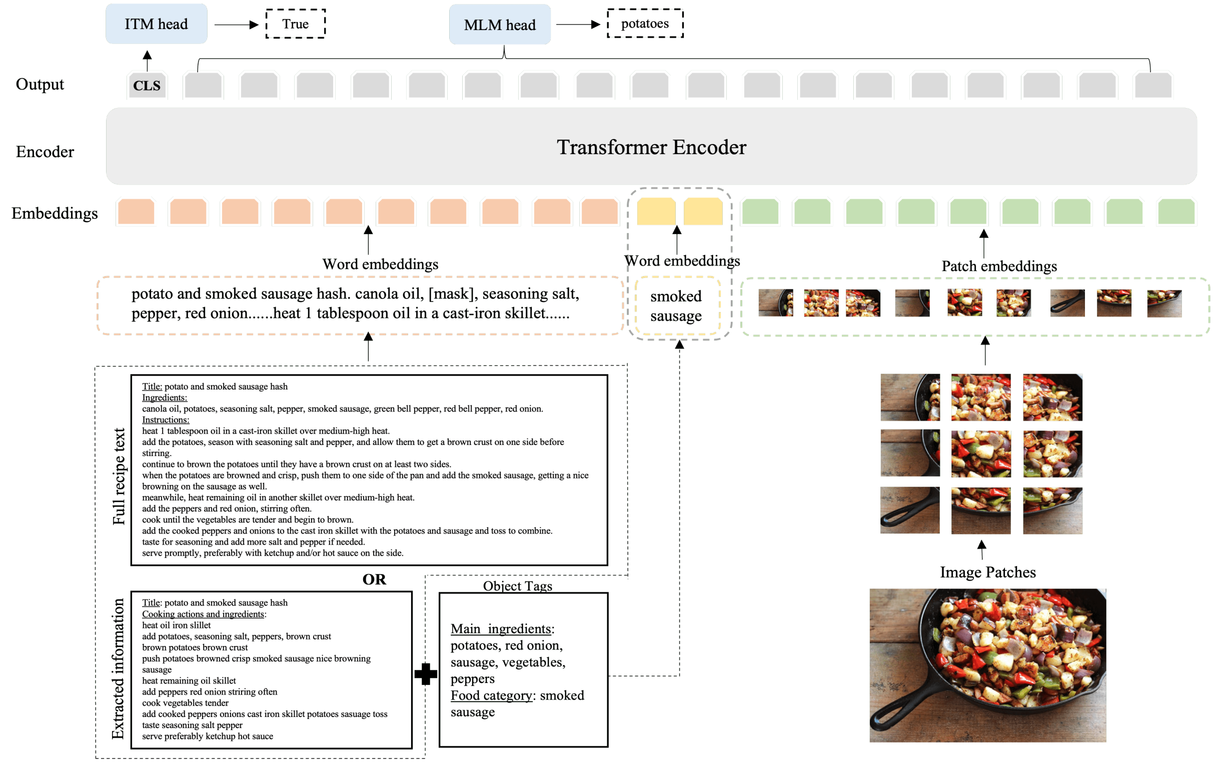
4.3.2 ViLT

The pre-trained weights of the Oscar were learned from image regions which are manually changed to image patches in our case. Fine-tuning Oscar could not yield desirable results. Therefore, we also want to explore another VLP model that is pre-trained on image patches. So here comes ViLT, which was designed to be a minimal VLP model without convolution or region supervision and also can obtain competitive performance to other advanced VLP models. The text input length in ViLT is 40 tokens so that we also use the extracted recipes as text inputs. The image features are also extracted from the pre-trained ViT [15]. Following the retrieval downstream task in ViLT, we fine-tune ViLT on the Recipe1M by randomly selecting 15 text negative samples for each data and rank them with the pre-trained ITM head. And then, the model is tuned with cross-entropy loss that maximizes the scores on positive pairs.

4.4 Our Proposed Recipe Vision-Language Model

Both Oscar and ViLT are pre-trained for vision-language downstream tasks so they aim to produce a general cross-modal representation for text and image modality. Moreover, as we discussed above, their pre-trained weights are limited and not suitable for downstream tasks using long text such as recipes.

[15] This ViT is vit-base-patch32-384 and pre-trained on the ImageNet-21k dataset.



potato and smoked sausage hash. canola oil, [mask], seasoning salt, pepper, red onion......heat 1 tablespoon oil in a cast-iron skillet......

Title: potato and smoked sausage hash
Ingredients:
canola oil, potatoes, seasoning salt, pepper, smoked sausage, green bell pepper, red bell pepper, red onion.
Instructions:
heat 1 tablespoon oil in a cast-iron skillet over medium-high heat.
add the potatoes, season with seasoning salt and pepper, and allow them to get a brown crust on one side before stirring.
continue to brown the potatoes until they have a brown crust on at least two sides.
when the potatoes are browned and crisp, push them to one side of the pan and add the smoked sausage, getting a nice browning on the sausage as well.
meanwhile, heat remaining oil in another skillet over medium-high heat.
add the peppers and red onion, stirring often.
cook until the vegetables are tender and begin to brown.
add the cooked peppers and onions to the cast iron skillet with the potatoes and sausage and toss to combine.
taste for seasoning and add more salt and pepper if needed.
serve promptly, preferably with ketchup and/or hot sauce on the side.

Title: potato and smoked sausage hash
Cooking actions and ingredients:
heat oil iron slillet
add potatoes, seasoning salt, peppers, brown crust
brown potatoes brown crust
push potatoes browned crisp smoked sausage nice browning sausage
heat remaining oil skillet
add peppers red onion striring often
cook vegetables tender
add cooked peppers onions cast iron skillet potatoes sasuage toss
taste seasoning salt pepper
serve preferably ketchup hot sauce

Main ingredients: potatoes, red onion, sausage, vegetables, peppers
Food category: smoked sausage

Figure 11: Illustration of RecipeVL. Food images are sliced into patches then linearly projected to patch embeddings. Recipe text has two scenarios: 1) directly using full recipe text; 2) using extracted recipes plus object tags defined in the Section 4.3.1.

Therefore, we follow their model architectures to build a recipe vision-language (RecipeVL) model for the food domain using the numerous recipe-image pairs in the Recipe1M. Figure 11 illustrates the overview of RecipeVL. Following ViLT, we initialize the interaction transformer weights from a pre-trained ViT [16]. The input z is the concatenation of text embeddings of cooking recipes and image embeddings of food images. Specifically, the text embedding $\bar{t}$ is the summation of word embeddings t, position embedding matrix $T^{pos} \in \mathbb{R}^{(n+1)\times H}$, and corresponding modal-type embedding vector $t^{type} \in \mathbb{R}^{H}$. The image embedding $\bar{v}$ is the summation of patch embeddings v, position embedding matrix $V^{pos} \in \mathbb{R}^{(m+1)\times H}$, and corresponding modal-type embedding vector $v^{type} \in \mathbb{R}^{H}$. The contextualized embedding z is iteratively updated through D-depth transformer blocks until the final sequence z^D. The pooled representation p of the whole cross-modal input is the linear projection of the first index of z^D through $W_{pool} \in \mathbb{R}^{\mathbb{H}\times\mathbb{H}}$ with a hyperbolic tangent activation. The hidden size H is 768, the transformer block depth D is 12 and each block has 12 attention heads.

[16] This ViT is vit-base-patch16-224 and pre-trained on the ImageNet-21k dataset.

$$\bar{t} = [t_{class}, t_1, ..., t_n] + T^{pos} + t^{type} \tag{5}$$

$$\bar{v} = [v_{class}, v_1, ..., v_m] + V^{pos} + v^{type} \tag{6}$$

$$z^0 = concat(\bar{t}, \bar{v}) \tag{7}$$

$$z^d = TR_{block}(z^0), \qquad d = 1, ..., D \tag{8}$$

$$p = tanh(z_0^D \cdot W_{pool}) \tag{9}$$

Since we want to investigate the expressive power of both extracted recipes and raw recipes, the text input of RecipeVL has two scenarios. The first scenario is using the full recipe text in order of title, ingredients, and instructions; the second is using the recipe title along the extracted cooking methods with associated food ingredients, plus the object tags defined in Section 4.3.1. The text sequence length is set to 300 and 100 tokens for the two scenarios, respectively. Since the patch size we use is 16 x 16 and we resize the input image to 224 x 224, the image sequence length is 197 = 1 + (224/16 + 224/16) patch tokens.

We train RecipeVL with the two commonly used objectives in VLP: image text matching (ITM) and masked language modeling (MLM). Specifically, for ITM, the paired image is randomly replaced with a different image from the dataset with 0.5 probability. The pooled feature p is projected through ITM head to logits over binary class. For MLM, the text tokens are randomly masked with 0.15 probability. Following ViLT, we also use whole word masking. The output z^D_{masked} is feed to MLM head to get the logits over vocabulary. Both logits are used to compute the negative log-likelihood loss., so $L = L_{MLM} + L_{ITM}$. In addition, when using extracted recipes as text inputs, following Oscar, inputs can be considered in two views (dictionary view and modality view). Hence, in this case, object tags should be corresponding to the paired image which means both of them are randomly replaced with a different sample.

5 Experiments and Results

In this section, we present our experiments on the methods introduced in the previous section with the implementation details. We also evaluate their performance on the cross-modal retrieval task. In addition, we use a novel explainable framework, VL-CheckList [43], to evaluate the fine-tuned ViLT and RecipeVL for a more comprehensive understanding.

5.1 Implementation Details

5.1.1 Dataset

Table 6 shows the number of image-text pairs used for training each method in our experiments. According to the preliminary analysis in Section 3.1, in most cases a recipe has only one associated food image, but if it has more than one image we then randomly choose up to 5 images per recipe during training. Moreover, we set the maximum number of ingredients and instructions used per recipe up to 20 respectively, which means when recipes have more than 20 ingredients or instructions, we take the first 20 in the text. Due to the training cost and time constraint, we fine-tuned Oscar and ViLT on a subset of the Recipe1M, which is the unique number of recipes in the training set. During evaluation, we randomly choose one image when a recipe has more than one associated image.

5.1.2 Evaluation Metrics

Following prior works, the cross-modal retrieval performance are measured using Recall@K where $K \in \{1, 5, 10\}$ on rankings of sample size $N = 1000$. Specifically, Recall@K indicates the percentage of all the queries for which the true matching item is retrieved amongst the top K. Therefore, the higher the Recall@K, the better the performance. In our experiments, we first randomly sample 5 different subsets of 1,000 image-text pairs from the test set, then we report the average Recall@K of the 5 test subsets.

5.1.3 Training Details

Following Salvador el at's work [28], for all experiments, food images are resized to 256 x 256 in their shortest edge and cropped to 224 x 224. During training, images are random cropped and horizontally flipped with 0.5 probability. During evaluation, images are center cropped.

ViLT and RecipeVL are trained on 2 NVIDIA 3090 GPUs with a batch size of 16 for 5 epochs and 30 epochs, respectively. Oscar and H-Transformer are trained on 5 NVIDIA 980 Ti GPUs plus 1 Titan X GPUs with a batch size of 32 for 10 epochs and a batch size of 128 for 30 epochs, respectively.

For fine-tuning Oscar and ViLT, we use their pre-trained weights from cross-modal retrieval task. H-transformer and RecipeVL are trained from scratch. Moreover, following ViLT, RecipeVL is fine-tuned for 5 more epochs on the validation set of the Recipe1M by randomly selecting 7 negative text samples for each image-text pair. AdamW optimizer are used for all experiments. Following training setting of each method, the base learning rate for H-transformer is $1e^{-4}$ with a step-wise rate decay of 0.1 every 10 epochs. The base

Methods	# Parameters	# Image-text pairs	# Training steps
H-Transformer	27M (recipe encoder) 87M (image encoder)	383,702	89,930
Oscar	157M	238,408	74,502
ViLT	111M	238,408	74,502
RecipeVL	134M	383,702	719,442

Table 6: Comparison of models in terms of parameters, training data and steps [17].

learning rate for Oscar is $2e^{-5}$ and linearly decayed to zero within total training steps. The base learning rate for ViLT and RecipeVL is $1e^{-4}$ with a linear decay of 0.01 but was first warmed up for 10% of the total training steps.

5.2 Cross-modal Retrieval Results

Methods	image-to-recipe			recipe-to-image		
	R@1	R@5	R@10	R@1	R@5	R@10
Extracted recipes						
H-Transformer	**50.3**	**81.6**	**89.3**	**49.9**	**82.0**	**88.3**
Fine-tuned Oscar	3.0	7.6	11.8	0.2	0.7	1.5
Fine-tuned ViLT	17.1	43.3	56.6	11.8	34.4	49.1
RecipeVL	48.6	78.7	87.6	47.1	78.4	87.2
Full recipes						
H-Transformer [18]	**58.3**	**86.2**	**91.8**	**59.6**	**86.1**	**92.2**
RecipeVL	56.2	84.8	90.7	56.0	84.2	90.3

Table 7: Performance comparison of RecipeVL with other methods on cross-modal retrieval task on 1K test set.

We compare the performance of RecipeVL with other methods by two groups, one is trained on extracted recipes and the other one is trained on full recipes. Table 7 shows the overall results. We can see that H-Transformer has the best performance among all the methods in both groups, but RecipeVL also achieved comparable results as a single-stream approach. Specifically, the overall performance of RecipeVL is about 2.25% lower than H-Transformer.

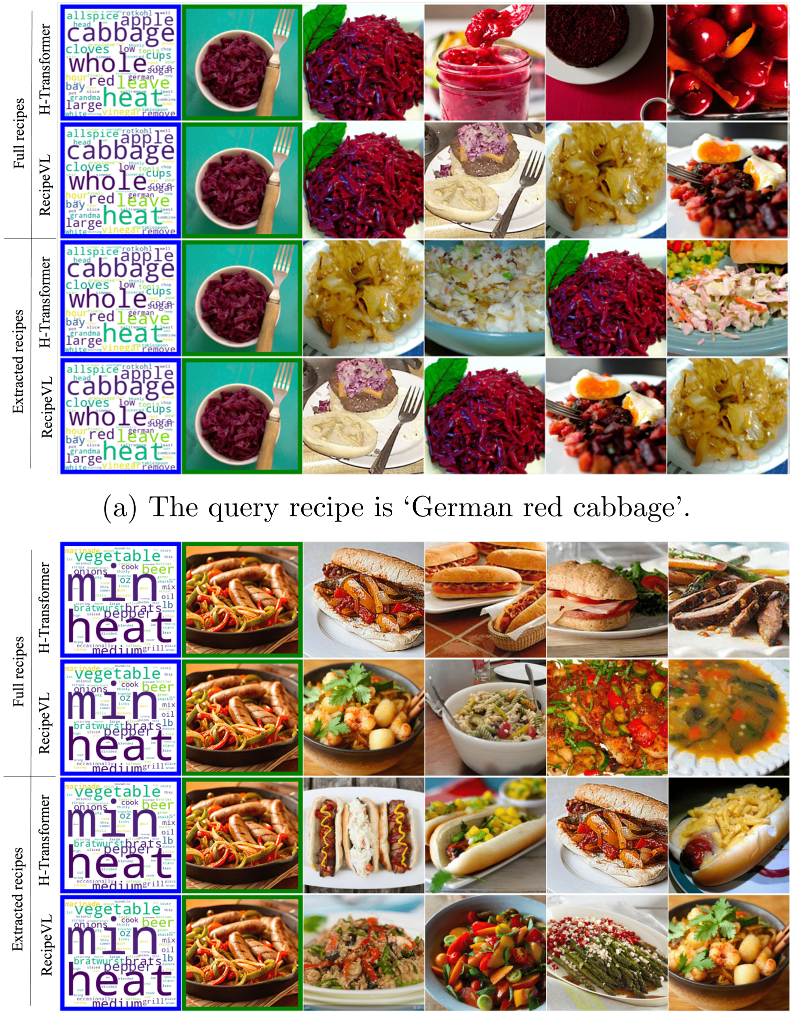
On the other hand, comparing the results of the same method in the two groups, the performance of H-Transformer and RecipeVL both decreased when using extracted recipes. Specifically, H-Transformer dropped by averagely 9.15% on R@1, 4.35% on R@5, and 3.05% on R@10 respectively on two sub cross-modal retrieval tasks while RecipeVL dropped by 8.25% on R@1, 5.95% on R@5, and 3.1% on R@10 respectively. We suspect that the attention distribution for the same ingredient in recipes changes after extracting,

[17] The training steps are calculated based on using 1 GPU and mentioned batch sizes.

[18] This results are quoted from H-Transformer's paper [28].

which makes extracted recipes less powerful to distinguish similar recipes. Meanwhile, our rule-based information extraction procedure cannot 100% accurately extract the correct ingredients and cooking methods, especially when the instructions are written in an unstructured manner. This can lead to information loss when using extracted recipes.

In addition, from Table 7 we also can see that the performance of both fine-tuned VLP models are significantly inferior compared to the other two methods, especially fine-tuned Oscar. We speculate the reason for such poor performance is that Oscar was pre-trained on image regions so when we change to image patches the pre-trained weights lost their advantages of learning image features. We fine-tuned both VLP models within reasonable epochs based on our computing resources. However, the performance of them are much worse compared to RecipeVL that was trained from scratch. It is worth to mention that we also experimented RecipeVL using extracted recipes without object tags and the results were worse, which indicates the effectiveness of the architecture of Oscar.



(a) The query recipe is ‘German red cabbage’.

(b) The query recipe is ‘bratwurst with saucy peppers & onions’

Figure 12: Recipe-to-image comparison between H-Transformer and RecipeVL on 1K test set. The left recipes are used as query and followed by the top 5 retrieved images in order. The image in green box is the ground truth. The recipe text for both examples is shown in Figure 13.

5.2.1 Qualitative analysis

In order to further compare H-Transformer and RecipeVL, we visualize two examples of recipe-to-image retrieval task from 1K test set as shown in Figure 12. We can see that the top 1 retrieved results for both examples are all the ground truth, but followed retrieved results are quite different among all the methods. In Figure 12 (a), the last two retrieved images for H-Transformer trained on full recipes (the first row) are irrelevant to ‘cabbage’ but are in red color while the retrieved results of H-Transformer trained on extracted recipes (the third row) are all relevant to ‘cabbage’ but three of them are not ‘red cabbage’. The retrieved results of RecipeVL (the second and fourth row) are all ‘cabbage’ and four of them are ‘red cabbage’. Interestingly, we can see that the ground truth image contains a fork, two of the retrieved images of RecipeVL also contain a fork, which does not show in any results of H-Transformer. This indicates that text embeddings learned from RecipeVL fused information from image modality. In Figure 12 (b), the retrieved results of H-Transformer are all shapely similar to the main visible ingredients ‘bratwurst’ in the ground truth image; while the results of RecipeVL tend to contain similar ingredients like red peppers and similar serving style.

Both examples’ retrieved results in Figure 12 demonstrate the differences between the single-stream approach and dual-stream approach. Specifically, in the dual-stream approach, the learned text/image embeddings with similar semantic information are closer than others in the common space. However, based on the retrieved results, the nearby image embeddings in the common space with similar colors or shapes can be noises. For the single-stream approach, the learned text/image embeddings fuse information from another modality, however, the fused information from another modality may not be helpful. For example, the third image in the fourth row shown in Figure 12 (b) are aligned with the mentioned ‘tomato’ and also with similar serving style to the ground truth image.

5.3 VL-CheckList

VL-CheckList [43] is a recent explainable framework to evaluate the capabilities of VLP models from three aspects: object, attribute, and relation. And the authors also further break down the three aspects into more fine-grained variables. Specifically, the attribute is composed of color, material, size, state, and action; the object is composed of size and location and further divided into large, medium, and small, and center, middle, and margin respectively; the relation is composed of action and spatial. Four datasets shown in Table 8 were used and categorized into three aspects, and then in the corpus for each image the paired text is rewritten to generate a negative sample for each aspect. For instance, in terms of color ‘the sheep is white.’ is rewritten to ‘the sheep is golden brown.’; or in terms of action ‘the child is brushing teeth.’ is rewritten to ‘the child is photographing teeth.’. The ITM head of a VLP model is used to predict if the given image-text pair is a true pair or not. In VL-CheckList, if the ITM score on the original text is higher than the score on the generated negative sample, they consider it as a positive output. In this manner, the accuracy is calculated by the number of positive outputs over the total number of test samples.

In our work, we want to have a comprehensive understanding of our methods not only relying on the performance of a single downstream task. Therefore, we use VL-Checklist

Name	Size	Adopt to
VG [12]	108K	Objects, Attribute, Relation
VAW [22]	72K	Attribute
HAKE [15]	104K	Objects, Relation
SWiG [24]	126K	Objects, Relation

Table 8: Datasets used in VL-CheckList

to evaluate our fine-tuned ViLT and RecipeVL. We pass H-Transformer due to its lack of ITM head and Oscar due its poor performance. The overall results are shown in Table 9, in which we can see that in terms of aspects, all the methods have better scores at object than other two aspects. Besides, RecipeVL has similar scores at relation and attribute while the fine-tuned ViLT performs better at attribute than relation. On the other hand, in terms of methods, the fine-tuned ViLT achieved the best score at object and attribute but the lowest at relation. RecipeVL (full) achieved better score than RecipeVL(info) at object but RecipeVL (info) is slightly better at attribute and relation.

Methods	Object	Attribute	Relation	Average
Fine-tuned ViLT	66.10	57.18	50.64	**57.97**
RecipeVL (info)	57.14	54.76	55.58	55.83
RecipeVL (full)	59.10	53.51	53.94	55.52

Table 9: Evaluation results of VL-CheckList for fine-tuned ViLT and RecipeVL. info denotes the model was trained on extracted recipes; full denotes the model was trained on full recipes.

5.3.1 Evaluation of object

Furthermore, scores of each variable for each aspect are reported. Table 10 shows the evaluation results of the object aspect. We can see that the fine-tuned ViLT achieved the best scores at each variable but all scores dropped compared to the results of ViLT reported in the original paper [43]. We speculate the reason to be the inherent characteristics of the data used. First, the datasets used in VL-CheckList are perfectly designed for VLP models, specifically the text are the image captions describing the objects in the paired image which are identical to the pre-training corpora of ViLT. In the Recipe1M dataset, the objects mentioned in the recipes are the more fine-grained ingredients. Thus, within limited training steps ViLT tends to learn fine-grained details of ingredients but may not fully capture the more nuanced information, which may also cause its potential loss of the original object recognition ability.

Similar to the findings in the original paper, the fine-tuned ViLT also tends to focus on large objects located in the centre of images. It also explains the poor performance of fine-tuned ViLT on the cross-modal food retrieval task. In contrast, although the scores of RecipeVL are not ideal due to the inherent differences of the data in the Recipe1M and VL-CheckList, we still can observe that **RecipeVL tends to focus on small and medium objects located in the middle of images**. This observation aligns with the

Methods	Object						Average
	Size			Location			
	Large	Medium	Small	Center	Middle	Margin	
Fine-tuned ViLT	72.24	63.20	60.80	71.78	66.52	62.05	**66.10**
RecipeVL (info)	56.14	58.14	58.78	55.91	57.43	56.40	57.14
RecipeVL (full)	58.18	59.40	59.47	58.39	60.12	59.05	59.10

Table 10: Evaluation of object. info denotes the model was trained on extracted recipes; full denotes the model was trained on full recipes.

Methods	Attribute					Average
	Color	Material	Size	State	Action	
Fine-tuned ViLT	62.78	59.44	54.49	55.44	53.76	**57.18**
RecipeVL (info)	56.79	53.03	59.14	53.09	51.77	54.76
RecipeVL (full)	54.63	52.84	55.29	52.30	52.50	53.51

Table 11: Evaluation of attribute. info denotes the model was trained on extracted recipes; full denotes the model was trained on full recipes.

characteristics of recipe data, where ingredients are the more fine-grained objects, and the presentation of the dishes typically occurs in the middle of images.

5.3.2 Evaluation of attribute

Table 11 presents the results of each variable for the attribute aspect. As we discussed in the previous subsection, the fine-tuned ViLT preserved some capability from pre-training, resulting in the best score on color, material, state, and action but those scores also decreased compared to the results of ViLT. It is worth to mention in the original paper, all compared VLP models are low on size due to the subjective description about size in natural language. Surprisingly, RecipeVL achieved higher score on size than the fine-tuned ViLT even though the recipe text lacks explicit words describing ingredient size. We suspect that **training on more fine-grained objects like ingredients can enhance the model's sensitivity on size attribute**.

Moreover, despite we expected RecipeVL could have better scores at other attributes, our dataset only focused on the food domain and so it may be challenging for RecipeVL to understand those attributes in a border spectrum.

5.3.3 Evaluation of relation

Table 12 presents the results for relation aspect. RecipeVL (info) achieved the best score on both action and spatial relation, while RecipeVL (full) outperforms the fine-tuned ViLT. We suspect the reason is recipe data contains more relation information. Specifically, we observe there are sequential relations between the instructions in recipes and also there are causal relations between cooking methods and ingredient appearances. Despite the inherent differences of the data in the Recipe1M and VL-CheckList, the scores of RecipeVL, although not optimal, are comparable to other VLP models evaluated in [43] which are pre-trained on millions of image-text pairs. **The scores of RecipeVL also indicates its ability to capture and understand the relational aspects of objects in different modalities**. In addition, we speculate the reason RecipeVL (info) achieves higher scores than RecipeVL (full) is that extracted recipes may have less noise on relation information. For example, some recipes end with sentences like 'enjoy the meal.', which is eliminated in extracted recipes but kept in full recipes.

Methods	Relation		Average
	Action	Spatial	
Fine-tuned ViLT	49.62	51.66	50.64
RecipeVL (info)	<u>55.80</u>	<u>55.35</u>	**55.58**
RecipeVL (full)	53.91	53.98	53.94

Table 12: Evaluation of relation. info denotes the model was trained on extracted recipes; full denotes the model was trained on full recipes.

6 Discussion

In this section, we address our research questions based on the experimental results; we discuss potential improvements and suggest some future work.

6.1 Research Questions

1. Does information extraction from cooking recipes help narrow the semantic gap in the cross-modal representation learning?

Based on our experimental results of the cross-modal retrieval task shown in Table 7, information extraction from recipes did not help narrow the semantic gap. Specifically, we experiment with both the dual-stream approach, H-Transformer, and the single-stream approach, RecipeVL, on the Recipe1M dataset but the performance of both declined when using extracted recipes. As we analyzed above, the main reason probably is our rule-based information extraction method cannot 100% accurately extract correct ingredients or cooking methods resulting in information loss. We also suspect the attention distribution of text changes after extracting, which makes the extracted recipes less powerful than full recipes to distinguish similar recipes in cross-modal retrieval task.

2. Can fine-tuning VLP models on the Recipe1M dataset help improve the performance of the cross-modal retrieval task?

Our experimental results show that fine-tuning VLP models can not help improve the performance of the cross-modal retrieval task on the Recipe1M dataset. Specifically, we fine-tuned Oscar and ViLT within reasonable epochs based on our computing resources. However, the results show the performance of both are quite deficient on the cross-modal retrieval task, especially Oscar. As we discussed earlier, there are a couple of reasons for such results. First of all, Oscar was pre-trained on image region features so that fine-tuning it on image patches did not work. Secondly, ViLT although was pre-trained on image patches, the results is still quite lower than what we expected in the first place. Incorporating the results of VL-CheckList, we found that recipe data contains more complex information than the image-text pairs the two VLP models trained on, which made them can not fully capture the characteristics of recipe data within reasonable training steps of fine-tuning.

3. Can single-stream approaches work better than dual-stream approaches on the Recipe1M dataset?

Although our results show the dual-stream approach outperforms the single-stream approach, we speculate the performance of RecipeVL can be further improved with additional enhancements or techniques, such as incorporating another training objective which can help increase the alignment between text and image modality.

6.2 Cross-modal Food Retrieval

Speaking of the application of cross-modal retrieval, the ultimate goal of the model is to retrieve the most relevant and best-matched results from one modality given the query from another modality within reasonable response time. In the dual-stream approach, the retrieved results from one modality are ranked based on distance metrics, such as cosine similarity or Euclidean distance, given a query from another modality. Corresponding embeddings for both modalities can be obtained beforehand through trained encoders. In the single-stream approach, the retrieved results are ranked based on probability predictions given image-text pairs. Therefore, there is an obvious drawback for single-stream approaches compared to dual-stream approaches. Given the same ranking size, the response time of single-stream approaches will depend on the computational complexity of the model, which also includes the data processing for different modalities, as well as any additional computations involved in the retrieval process. However, the response time of dual-stream approaches only relies on the distance computation between pre-obtained embeddings for different modalities. Hence, as the ranking size increases, the response time of single-stream approaches can become much slower than dual-stream approaches, which is not desirable in the application of cross-modal retrieval. Specifically, for instance, in our work, the evaluation time of RecipeVL for the 1K test set is over 50 minutes while it only takes less than 1 minute for H-Transformer.

6.3 Improvements and Future work

RecipeVL follows the basic architecture of VLP models as a single-stream approach and our experimental results demonstrate its effectiveness on the cross-modal retrieval task in the food domain, even though it did not outperform the dual-stream approach - H-Transformer. We believe RecipeVL can be further improved with additional enhancements or techniques as follows:

Train with a large batch size; or train on a larger recipe dataset. Due to our limited computing resources, we only train the model with a batch size of 16 for 30 epochs. A recent study [41] of the Recipe1M shows that large batch training can improve the model performance. In addition, it is worth to mention ViLT was pre-trained with a batch size of 4096. On the other hand, the latest model VLPCook [30] proposed by Shukor et al. is trained on Recipe1M+ [18] which contains over 13 millions food images. Its performance was improved by 3% compared to be trained on the Recipe1M.

Implement a re-ranking strategy and fine-tune for more training steps. We observe that the food images of different recipes using similar ingredients can look similar. And the Recipe1M dataset is collected from the cooking websites and tends to be noisy, in which positive pairs are not always strongly-correlated. Therefore, the ranking results only based on the probability predictions can be rough. Wang et al. [38] proposed a novel framework, AGREE, in which an entity alignment module was implemented during fine-tuning and also re-ranking upon VLP models. Their results showed AGREE can potentially help improve the performance of cross-modal retrieval task on fine-grained recipe-image pairs.

Enhance cross-modal alignment along extra modules or training objectives. The SOTA results in many vision-language downstream tasks have been updated constantly as new VLP models and methods are introduced, such as ALBEF [13] and BEiT

[37]. The fundamental principle is to increase the cross-modal alignment between modalities by adapting extra modules or training objectives upon previous VLP models. Likewise, RecipeVL can also be improved following similar procedures.

7 Conclusion

In this work, we study the cross-modal retrieval task in the food domain. Specifically, we experiment with the two commonly used approaches in cross-modal representation learning on the Recipe1M dataset: single-stream and dual-stream. We also propose to extract key information from recipes and use them to fine-tune Oscar and ViLT and re-train H-Transformer. Our results demonstrate that cross-modal food retrieval is a challenging task for VLP models within limited training epochs. However, the performance of RecipeVL that is trained from scratch and follows the basic architecture of VLP models demonstrates the effectiveness of single-stream approaches in the food domain, even through it did not outperform the dual-stream approach, H-Transformer. In addition, we use VL-CheckList to evaluate the capability of our methods in terms of object, attribute, and relation. We also discuss that in the real-life application of cross-modal food retrieval, single-stream approaches may cause response time problem compared to dual-stream approaches.

In conclusion, the cross-modal retrieval task in the food domain is challenging. Despite the requirement of large computing resource, the fine-grained ingredients and complex relationships between each cooking instruction and also between cooking methods and ingredient appearances are much harder to be captured than common image-text pairs. Regardless of single-stream or dual-stream approaches, future work discussed in the Section 6.3 can be potential directions for further investigations.

References

[1] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. “Food-101–mining discriminative components with random forests”. In: *European conference on computer vision*. Springer. 2014, pp. 446–461.

[2] Jing-Jing Chen et al. “Deep understanding of cooking procedure for cross-modal recipe retrieval”. In: *Proceedings of the 26th ACM international conference on Multimedia*. 2018, pp. 1020–1028.

[3] Jacob Devlin et al. “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *NAACL-HLT* (2019), pp. 4171–4186.

[4] Ian Goodfellow et al. “Generative adversarial networks”. In: *Communications of the ACM* 63.11 (2020), pp. 139–144.

[5] Yash Goyal et al. “Making the v in vqa matter: Elevating the role of image understanding in visual question answering”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 6904–6913.

[6] Kaiming He et al. “Deep residual learning for image recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[7] Sepp Hochreiter and Jürgen Schmidhuber. “Long short-term memory”. In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[8] Matthew Honnibal and Mark Johnson. “An Improved Non-monotonic Transition System for Dependency Parsing”. In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal: Association for Computational Linguistics, Sept. 2015, pp. 1373–1378. DOI: `10.18653/v1/D15-1162`. URL: `https://aclanthology.org/D15-1162`.

[9] Drew A Hudson and Christopher D Manning. “Gqa: A new dataset for real-world visual reasoning and compositional question answering”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 6700–6709.

[10] Wonjae Kim, Bokyung Son, and Ildoo Kim. “ViLT: Vision-and-Language Transformer Without Convolution or Region Supervision”. In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, 2021, pp. 5583–5594. URL: `http://proceedings.mlr.press/v139/kim21k.html`.

[11] Alexander Kolesnikov et al. “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”. In: (2021).

[12] Ranjay Krishna et al. “Visual genome: Connecting language and vision using crowdsourced dense image annotations”. In: *International journal of computer vision* 123.1 (2017), pp. 32–73.

[13] Junnan Li et al. “Align before fuse: Vision and language representation learning with momentum distillation”. In: *Advances in neural information processing systems* 34 (2021), pp. 9694–9705.

[14] Xiujun Li et al. “Oscar: Object-semantics aligned pre-training for vision-language tasks”. In: *European Conference on Computer Vision*. Springer. 2020, pp. 121–137.

[15] Yong-Lu Li et al. “Pastanet: Toward human activity knowledge engine”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 2020, pp. 382–391.

[16] Tsung-Yi Lin et al. “Microsoft coco: Common objects in context”. In: *European conference on computer vision.* Springer. 2014, pp. 740–755.

[17] Zhiyuan Liu, Yankai Lin, and Maosong Sun. *Representation learning for natural language processing.* Springer Nature, 2020.

[18] Javier Marın et al. “Recipe1m+: A dataset for learning cross-modal embeddings for cooking recipes and food images”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43.1 (2021), pp. 187–203.

[19] Tomas Mikolov et al. “Efficient estimation of word representations in vector space”. In: (2013).

[20] Dim P Papadopoulos et al. “Learning Program Representations for Food Images and Cooking Recipes”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 2022, pp. 16559–16569.

[21] Hai X Pham et al. “CHEF: Cross-Modal Hierarchical Embeddings for Food Domain Retrieval”. In: *35th AAAI Conference on Artificial Intelligence, AAAI 2021.* Association for the Advancement of Artificial Intelligence. 2021, pp. 2423–2430.

[22] Khoi Pham et al. “Learning to predict visual attributes in the wild”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 2021, pp. 13018–13028.

[23] Thomas Politzer. *Vision Is Our Dominant Sense.* Ed. by Brainline. 2008. URL: `https://www.brainline.org/article/vision-our-dominant-sense`.

[24] Sarah Pratt et al. “Grounded situation recognition”. In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IV 16.* Springer. 2020, pp. 314–332.

[25] Alec Radford et al. “Learning transferable visual models from natural language supervision”. In: *International Conference on Machine Learning.* PMLR. 2021, pp. 8748–8763.

[26] Shaoqing Ren et al. “Faster r-cnn: Towards real-time object detection with region proposal networks”. In: *Advances in neural information processing systems* 28 (2015).

[27] Amaia Salvador et al. “Learning Cross-modal Embeddings for Cooking Recipes and Food Images”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.* 2017.

[28] Amaia Salvador et al. “Revamping Cross-Modal Recipe Retrieval with Hierarchical Transformers and Self-supervised Learning”. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR).* 2021.

[29] Piyush Sharma et al. “Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning”. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers).* 2018, pp. 2556–2565.

[30] Mustafa Shukor, Nicolas Thome, and Matthieu Cord. *Vision and Structured-Language Pretraining for Cross-Modal Food Retrieval.* 2023. arXiv: `2212.04267 [cs.CV]`.

[31] Mustafa Shukor et al. “Transformer Decoders with MultiModal Regularization for Cross-Modal Food Retrieval”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 2022, pp. 4567–4578.

[32] Yu Sugiyama and Keiji Yanai. “Cross-Modal Recipe Embeddings by Disentangling Recipe Contents and Dish Styles”. In: *Proceedings of the 29th ACM International Conference on Multimedia.* 2021, pp. 2501–2509.

[33] Ashish Vaswani et al. “Attention is all you need”. In: *Advances in neural information processing systems* 30 (2017).

[34] Hao Wang et al. “Cross-modal food retrieval: learning a joint embedding of food images and recipes with semantic consistency and attention mechanism”. In: *IEEE Transactions on Multimedia* 24 (2021), pp. 2515–2525.

[35] Hao Wang et al. “Learning Cross-Modal Embeddings With Adversarial Networks for Cooking Recipes and Food Images”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.* 2019, pp. 11572–11581.

[36] Kaiye Wang et al. “A comprehensive survey on cross-modal retrieval”. In: *arXiv preprint arXiv:1607.06215* (2016).

[37] Wenhui Wang et al. *Image as a Foreign Language: BEiT Pretraining for All Vision and Vision-Language Tasks.* 2022. arXiv: `2208.10442 [cs.CV]`.

[38] Xiaodan Wang et al. “AGREE: Aligning Cross-Modal Entities for Image-Text Retrieval Upon Vision-Language Pre-trained Models”. In: *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining.* 2023, pp. 456–464.

[39] Zhongwei Xie et al. “Cross-Modal Retrieval between Event-Dense Text and Image”. In: *Proceedings of the 2022 International Conference on Multimedia Retrieval.* 2022, pp. 229–238.

[40] Zhongwei Xie et al. “Learning TFIDF Enhanced Joint Embedding for Recipe-Image Cross-Modal Retrieval Service”. In: *IEEE Transactions on Services Computing* (2021).

[41] Jing Yang, Junwen Chen, and Keiji Yanai. “Transformer-Based Cross-Modal Recipe Embeddings with Large Batch Training”. In: *MultiMedia Modeling: 29th International Conference, MMM 2023, Bergen, Norway, January 9–12, 2023, Proceedings, Part II.* Springer. 2023, pp. 471–482.

[42] Peter Young et al. “From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions”. In: *Transactions of the Association for Computational Linguistics* 2 (2014), pp. 67–78.

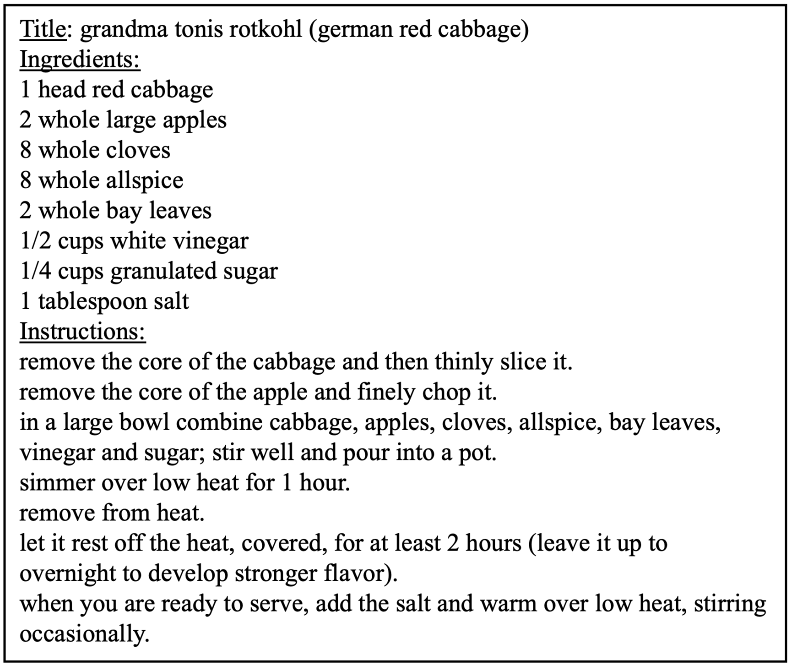
[43] Tiancheng Zhao et al. “VL-CheckList: Evaluating Pre-trained Vision-Language Models with Objects, Attributes and Relations”. In: *arXiv preprint arXiv:2207.00221* (2022).

A Cooking Verb List

add	bake	barbecue	baste	beat
blanch	blend	bring	boil	braise
bread	break	broil	brown	brush
candy	can	carve	char	check
chill	chop	clean	coat	combine
cook	cool	core	cover	cream
crisp	crush	cube	cut	debone
decorate	deep-fry	dehydrate	dice	dip
dissolve	drain	dress	drizzle	dry
drop	dust	emulsify	ferment	filet
flame	flambé	flip	fold	freeze
fry	garnish	glaze	grate	grill
grind	gut	heat	hull	infuse
julienne	knead	layer	level	liquefy
light	marinate	mash	measure	melt
mince	mix	mold	move	microwave
oil	pack	pan-fry	parboil	pare
peel	place	pickle	pierce	pinch
pit	poach	pop	pour	preheat
preserve	pressure-cook	prick	puree	push
put	reduce	remove	rinse	refrigerate
roast	roll	sauté	saute	serve
scald	scramble	scallop	score	sear
season	shred	simmer	sip	sift
skewer	slice	smoke	smooth	soak
soften	sprinkle	sous-vide	spatchcock	spice
spread	squeeze	steam	steep	stir
strain	stick	stuff	submerge	sweeten
swirl	taste	take	temper	tenderize
thicken	toast	top	toss	truss
thread	turn on	turn off	wash	weight
whip	whisk	wilt		

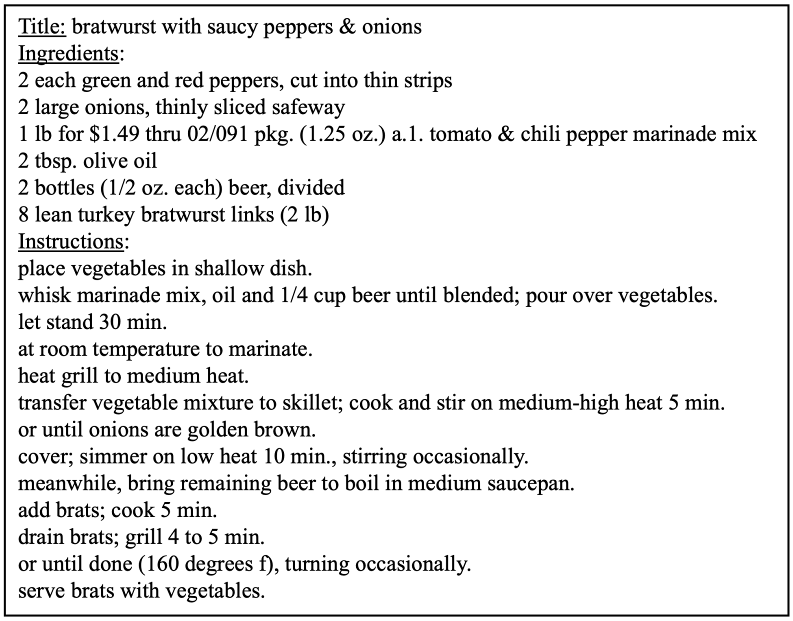
Table 13: List of cooking verbs

B Recipe-to-image examples



Title: grandma tonis rotkohl (german red cabbage)
Ingredients:
1 head red cabbage
2 whole large apples
8 whole cloves
8 whole allspice
2 whole bay leaves
1/2 cups white vinegar
1/4 cups granulated sugar
1 tablespoon salt
Instructions:
remove the core of the cabbage and then thinly slice it.
remove the core of the apple and finely chop it.
in a large bowl combine cabbage, apples, cloves, allspice, bay leaves, vinegar and sugar; stir well and pour into a pot.
simmer over low heat for 1 hour.
remove from heat.
let it rest off the heat, covered, for at least 2 hours (leave it up to overnight to develop stronger flavor).
when you are ready to serve, add the salt and warm over low heat, stirring occasionally.

(a) The recipe is 'German red cabbage'.



Title: bratwurst with saucy peppers & onions
Ingredients:
2 each green and red peppers, cut into thin strips
2 large onions, thinly sliced safeway
1 lb for $1.49 thru 02/091 pkg. (1.25 oz.) a.1. tomato & chili pepper marinade mix
2 tbsp. olive oil
2 bottles (1/2 oz. each) beer, divided
8 lean turkey bratwurst links (2 lb)
Instructions:
place vegetables in shallow dish.
whisk marinade mix, oil and 1/4 cup beer until blended; pour over vegetables.
let stand 30 min.
at room temperature to marinate.
heat grill to medium heat.
transfer vegetable mixture to skillet; cook and stir on medium-high heat 5 min.
or until onions are golden brown.
cover; simmer on low heat 10 min., stirring occasionally.
meanwhile, bring remaining beer to boil in medium saucepan.
add brats; cook 5 min.
drain brats; grill 4 to 5 min.
or until done (160 degrees f), turning occasionally.
serve brats with vegetables.

(b) The recipe is 'bratwurst with saucy peppers & onions'

Figure 13: Example recipes used in Figure 12.