Ingredient Prediction from Food Images using Multi-modal

Ujwal Pratap Krishna, Donghee Lee, Pinn Prugsanapan,

Abstract

The rapid growth of social media platforms, recipe apps, and online food communities has led to an explosion of food images shared online. Automatically extracting detailed information about the ingredients used in these food images has become a challenging and important task for various applications, including dietary analysis, menu planning, and personalized cooking recommendations. This project presents a novel deep-learning approach for ingredient prediction from food images.

Our approach which can be found in Github repository, involves multi-modal networks, particularly the Salesforce's Blip series, which combines a visual encoder with a language model. n this research, one innovative aspect we explore is the modification of the instructBlip model to an encoder-only language model architecture, specifically tailored for multi-label classification tasks. This adaptation addresses the limitations of the original decoder-based structure of instructBlip, which was less effective for tasks requiring the identification of multiple labels, such as ingredient prediction from food images. For a comprehensive study, we plan to compare it with a baseline model using TF_{set} from Inversecooking [9], which uses different multi-modal architecture. We found that using pre-trained BERT LLM with a Classification head in InstructBLIP architecture[4] improves the performance over the baseline. Our model performs better with IoU and F1 scores of 34.01 and 50.76 compared to the TF_{set} model which is based on transformer architecture and has IoU and F1 scores of 31.80 and 48.26.

1 Introduction

The exponential growth of digital platforms and the widespread sharing of culinary creations on social media have ushered in an era where food photography has become a ubiquitous form of visual expression. Capturing and sharing images of appetizing dishes has become a cultural phenomenon, with individuals, food enthusiasts, and professionals alike showcasing their culinary skills through food imagery. In this digital gastronomic landscape, each image holds not only the aesthetics of gastronomy but also hidden valuable information—its ingredients.

In recent years, significant advancements in visual recognition tasks, such as natural image classification, object detection, and semantic segmentation, have greatly enhanced the capabilities of computer vision systems. However, when it comes to the domain of food recognition, unique challenges arise. Food and its components exhibit high intraclass variability, and they undergo substantial deformations during the cooking process. Moreover, ingredients in cooked dishes are often occluded, and they come in diverse colors, forms, and textures. Successful visual ingredient detection requires higher-level reasoning and prior knowledge. Previously, the image-to-recipe problem has been framed as a retrieval task[3, 2, 1], where a recipe is retrieved from a fixed dataset based on image similarity scores in embedding space, However, such systems heavily rely on the dataset's size, diversity, and the quality of learned embeddings, often failing when a matching recipe for the image query does not exist in the static dataset. To overcome these limitations, an alternative approach is presented in this project: introduce a model that directly predicts ingredients from images, rather than relying on retrieval-based methods.

The contributions of our paper can be summarized as:

- We introduce a multi-modal ingredient prediction system that harnesses the state-of-the-art InstructBLIP framework, incorporating a modification: the transition of the language model to an encoder-only architecture. This adjustment is specifically targeted to enhance the model's proficiency in multi-label classification tasks.
- We exhaustively evaluate this system against the baseline model TF_{set} developed in Inverse Cooking by Amaia Salvador et al.[9]

2 Related work

2.1 Multi-label Classification

Substantial research in Deep Learning has aimed to improve multi-label classification by modeling label dependencies. One early technique was to create a powerset [11] of all label combinations, but this does not scale due to exponentially growing computational costs for large label spaces like in food ingredient prediction. Modeling the full joint probability distribution is also prohibitively expensive. Methods based on classifier chains, such as Bayesian probabilistic classifier chains [3], decompose the joint distribution into conditional distributions using a chain rule to alleviate computational demands. However, the chain order introduces implicit ordering. More recent CNN-RNN approaches [12] also capture conditional label dependencies in an ordered manner and face potential ordering bias. Overall, existing multi-label classification techniques struggle to effectively and efficiently account for dependencies among large label sets. New approaches are needed to overcome computational and ordering issues while preserving model accuracy. Most multi-label classification models need to make a prediction for every potential label, which ignores relationships between labels. Some recent research by S Hamid Rezatofighi et al. [7, 8] has explored using joint embeddings for the inputs and labels to maintain label correlations, or predicting the number of labels instead of the labels themselves, though this still assumes independence. There are several different types of loss functions that have been studied for multi-label classification objectives, such as binary logistic loss, cross-entropy loss on the label distribution, mean squared error loss on the label distribution, and ranking-based losses. Recent large-scale studies highlight the effectiveness of modeling the full label distribution directly using a target distribution loss. This suggests capturing the label correlations and co-occurrence statistics through the target distribution is a promising approach for multi-label classification [6].

2.2 InverseCooking

Salvadore et al.[9] introduced a comprehensive suite of neural network models for generating cooking recipes from food images, with a focus on ingredient prediction as well as recipe generation. As our work is focused on ingredient prediction from food images, we further examine the modeling approaches proposed by Salvador et al. specifically for this task. In particular, we study their models designed for ingredient prediction which treat ingredients as either ordered lists or unordered sets. The list-based models include feed-forward convolutional networks trained with different losses: binary cross-entropy, intersection over union, dice coefficient, and Tanimoto distance. These capture ingredients as multi-label outputs but do not model inter-dependencies. The authors also train sequential transformer models to predict ingredient lists, imposing order and capturing dependencies through the language modeling approach or adding ingredient shuffling as data augmentation. For set-based models, the authors propose a set transformer architecture that leverages the auto-regressive transformer to model ingredient co-occurrences while satisfying order invariance in its set predictions.

The modeling strategies described above are: FF_{BCE}, FF_{IOU}, FF_{DC}, FF_{TD}, TF_{list}, $TF_{list+shuf}$, and TF_{set}. These models are designed for ingredient prediction and the modeling strategies are implemented as the following:

1. FF_{BCE} (Feed-Forward with Binary Cross-Entropy): A feed-forward convolutional neural network that employs Binary cross-entropy loss function.

2. FF_{IOU} (Feed-Forward with Intersection over Union): A feed-forward neural network that employs Intersection over Union (IOU) loss function.

3. FF_{DC} (Feed-Forward with Dice Coefficient): A feed-forward neural network that employs Dice coefficient loss function.

4. FF_{TD} (Feed-Forward with Tanimoto Distance): A feed-forward neural network that employs Tanimoto distance loss function.

5. TF_{list} (Transformer-based for Ingredient List Prediction): Unlike the previous feed-forward models, TF_{list} employs a transformer architecture, allowing it to capture dependencies between ingredients.

6. $TF_{list+shuf}$ (Transformer-based for Ingredient List Prediction with Shuffling): Similar to TF_{list}, it is a transformer-based model for ingredient list prediction. However, it incorporates a shuffling mechanism during training to improve diversity in ingredient predictions.
7. TF_{set} (Transformer-based for Ingredient Set Prediction): employs a transformer architecture like TF_{list} but is tailored for ingredient sets, allowing more flexibility in predictions without considering ingredient order.

In this project, we are using the IoU and F1 evaluation value(Table 1) from the model TF_{set} which is the best-performed model from the paper[9] as our base-line.

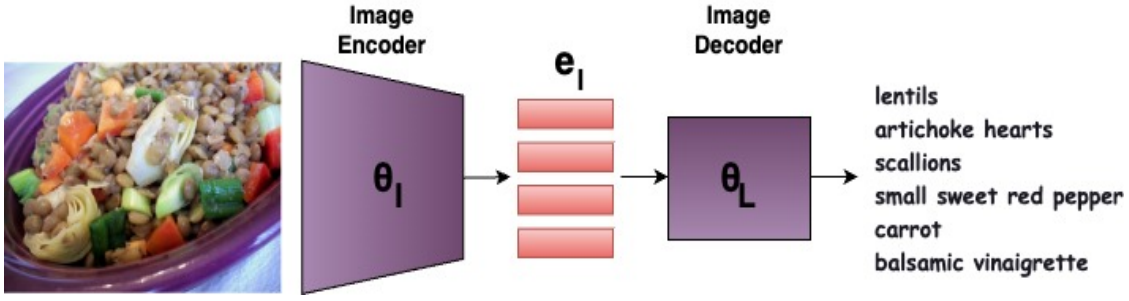

Figure 1: InverseCooking Ingredient Prediction Model as depicted in *[Amaia Salvador et al.,2019,"Inverse Cooking: Recipe Generation from Food Images"]*[9]: "Extracts image features e_I with the image encoder, parametrized by θ_I. Ingredients are predicted by θ_L", which is one of the above-mentioned models

2.3 InstructBLIP

Recent advancements have shown encouraging outcomes by integrating vision and language models based on the foundation of pre-trained BLIP models. Our methodology involves multi-modal networks, specifically the Salesforce's Blip series, which merges a visual encoder with a language model[5]. This is akin to the approach adopted by Wenliang Dai et al.[4]. In their research, they carried out a systematic and comprehensive study on vision-language instruction tuning based on the pre-trained BLIP-2 models. They collected 26 publicly available datasets, encompassing a wide range of tasks and capabilities, and converted them into an instruction-tuning format. They also introduced an instruction-aware Query Transformer, which extracts informative features tailored to the given instruction. Their models achieved state-of-the-art zero-shot performance across all 13 held-out datasets, significantly outperforming BLIP-2 and larger Flamingo models.
InstructBLIP enhances BLIP by adding instruction conditioning blocks, enabling the model to follow natural language instructions to perform image manipulation iteratively. The initial InstructBLIP work focused on text-based image editing, allowing realistic manipulation of attributes like adding/removing objects, scene editing, color, and texture changes, etc. Subsequent works have adapted InstructBLIP for further multimodal tasks including text-based video editing, text-to-image generation, and guided image synthesis. However, the application of InstructBLIP for food analysis and recipe generation tasks remains relatively unexplored. Our research examines the effectiveness of InstructBLIP variants trained specifically for predicting cooking ingredients from food images.

We demonstrate that InstructBLIP's cross-modal foundations coupled with instruction tuning provide an effective approach for not only extracting ingredients but also generating full recipes conditioned on the visual inputs. Comparisons to prior food-centric models including the TFset ingredient prediction baseline[9] provide important insights and effective performance evaluation of our approach. Our experiments open up new opportunities for deploying InstructBLIP in recipe-oriented domains beyond its original image and video editing applications.

2.4 CLIP

Clip models are neural network architectures focused on contrastive language-image pre-training. They were introduced in a 2021 paper "Learning Transferable Visual Models From Natural Language Su-

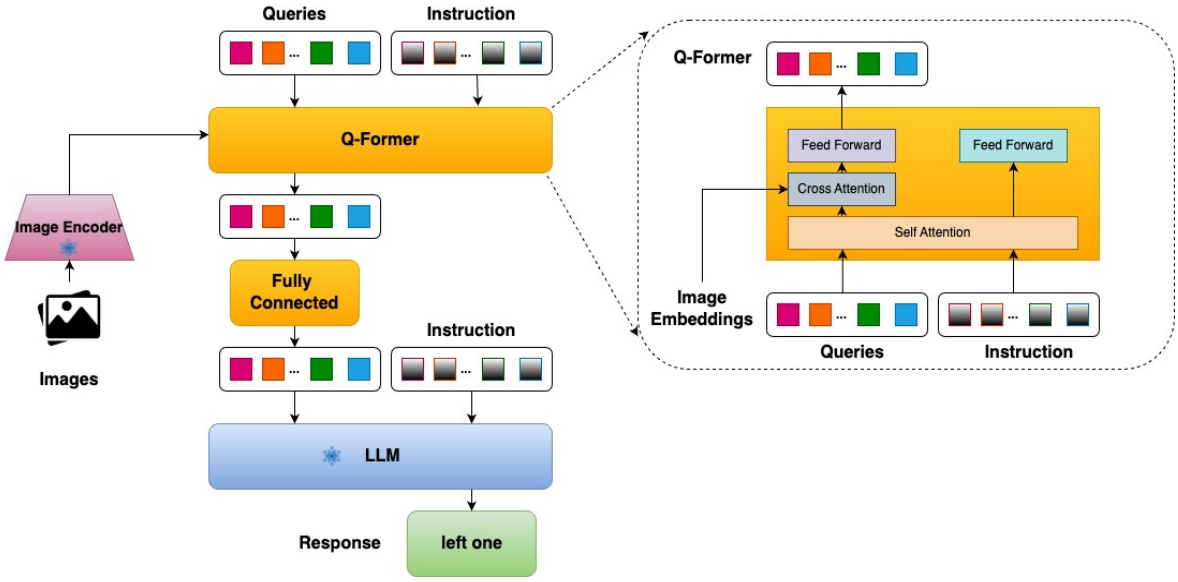

Figure 2: Model architecture of InstructBLIP as depicted in [*Wenliang Dai et al., 2023, "InstructBLIP: Towards General-purpose Vision-Language Models with Instruction Tuning"*][4]. "The Q-Former extracts instruction-aware visual features from the output embeddings of the frozen image encoder, and feeds the visual features as soft prompt input to the frozen LLM.[4]

pervision" by Radford et al.

The key innovation with clip models is that they are trained on a large dataset of text-image pairs to learn multimodal representations, without the need for explicit object labels. Specifically, the model tries to predict if an image-text pair matches or doesn't match. The contrastive approach allows the model to learn powerful associations between visual concepts and language.

3 Data set

In this project, we utilized a dataset known as Recipe1M[10]. This dataset can be divided into two main parts. The first part contains fundamental information, including recipe titles, lists of ingredients, and step-by-step instructions for preparing various dishes. The second part builds upon the first one and includes images associated with these recipes, which are provided in RGB JPEG format. Additionally, some recipes in this dataset are labeled with course categories such as appetizer, side dish, or dessert.

Upon analyzing the dataset, we found that, on average, a recipe typically consists of nine ingredients and is accompanied by ten instructions. Approximately half of the recipes in the dataset have images that show the fully prepared dish. It's worth noting that the dataset contains a small percentage (around 0.4%) of duplicate recipes and about 2% of duplicate images, where different recipes may share the same image. Excluding the duplicate recipes, approximately 20% of the recipes have non-unique titles but differ in terms of a median of 16 ingredients. Additionally, there's a small subset (0.2%) of recipes that share the same ingredients but tend to be relatively simple, typically containing a median of six ingredients.

For the purpose of our experiments, we took care to remove exact duplicates and recipes that shared the same image to prevent any overlap between the training and test subsets of the dataset. Roughly 70% of the dataset is designated for training, while the remaining portion is evenly split between the validation and test sets. The data distribution in this dataset follows a heavy-tailed pattern, meaning that a relatively small number of ingredients and recipes account for a significant portion of occurrences. On one end of the spectrum, there are recipes with just one step, often instructing to "Combine all ingredients." On the other end, there are lengthy recipes and ingredient lists associated with sub-recipes, which are considered outliers.

Regarding the images in the dataset, specific collections of recipes include a substantial number of user-submitted images. For instance, popular recipes like chocolate chip cookies have a notably higher

number of images compared to the average. Notably, 25% of the images are associated with only 1% of the recipes, and half of all images are related to just 10% of the recipes. The second layer of the dataset, focusing on unique recipes, consists of approximately 333,000 recipes.

Ingredient	Quantity
lentils	1 1/4 cups
artichoke hearts	8 1/2 ounce
scallions	4
small sweet red pepper	1
carrot	1
balsamic vinaigrette	1/4 cup

Figure 3: Recipe1M Sample "Warm Lentil Salad"

4 Methodology

4.1 Data Preprocessing

In our research, we adopted the data preprocessing methodology from the inversecooking framework[9]. Our dataset comprises 720,639 training, 155,036 validation, and 154,045 test recipes. Each recipe includes a list of ingredients and, in some cases, an accompanying image. For our experiments, we specifically utilized recipes that included images, excluding those with fewer than two ingredients or two instructions. This filtering process yielded approximately 250,000 training, 54,000 validation, and 54,000 test samples.

Given that the dataset was compiled from various cooking websites, it presents a high degree of unstructured data with extremely specific ingredient descriptions. For instance, 'cheese' is represented in 400 distinct variations, while 'olive oil' is listed under various labels such as 'olive oil', 'virgin olive oil', and 'Spanish olive oil'. To address this, we condensed the original ingredient vocabulary of over 16,000 entries into 1,488 unique ingredient clusters, replicating the approach used in inversecooking.

For the purposes of the instructBlip experiment, we employed BERT's pretrained tokenizer for processing the Q-former input. Additionally, we utilized the backbone language model's tokenizer for parsing the inputs directed to the language model.

4.2 Pretrained InstructBlip with Decoder

In light of the recent success of decoder-only models such as LLaMA and GPT in delivering impressive performances, we initially explored a decoder-only multimodal approach for our task. We conducted experiments using the pretrained model of instructBlip, developed by Salesforce. A notable feature of the Blip series is its unique training approach, which focuses solely on the Q-former, without training the vision encoder and the language model. This method leverages the pre-existing knowledge of well-trained models like ViT, LLaMA, and Flan-T5. By training only the Q-former, instructBlip effectively utilizes the foundational knowledge of these models, enabling the q-former to integrate and apply this knowledge effectively.

The Q-former is trained individually for each language model, utilizing 26 different datasets and 11 task categories. This model is tailored for general visual-language tasks such as question answering and image captioning, with the goal of generating text-based responses. Therefore, its language backbone is comprised of decoder-only models. InstructBlip supports various language models including Vicuna-7b, Vicuna-13b, Flan-T5-XL, and Flan-T5-XXL. Due to resource limitations, our experiments were conducted with Vicuna-7b.

4.3 InstructBlip with BERT

Our initial experiments led to the conclusion that decoder models are not well-suited for multi-label classification tasks. Consequently, we shifted our focus to encoder-only language models. In this phase,

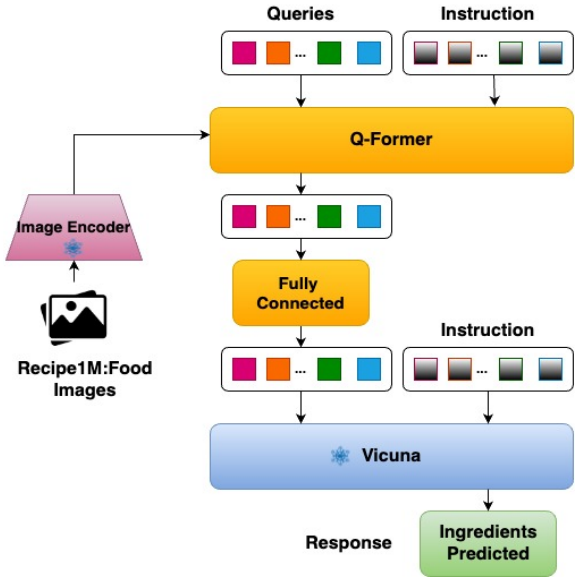

Figure 4: Model architecture of InstructBLIP with Vicuna LLM used for zero-shot inference in our experiments.

we employed BERT-large as our encoder language model. Since instructBlip was originally designed exclusively with decoder models, we made some structural modifications to the model and trained it from scratch for this experiment. To align the dimension sizes between the Q-former and BERT, we adjusted the language projection layer. Furthermore, we attached a classification head to the BERT head, projecting it onto 1,488 ingredients. For each ingredient, Binary Cross Entropy Loss was utilized as the loss function.

The model was trained for a total of 25 epochs using the complete training dataset. We set the batch size to 128. This approach allowed us to explore the effectiveness of an encoder-only language model, like BERT, in conjunction with instructBlip's capabilities, specifically targeting the intricacies of ingredient prediction from food images. This experiment represents a significant deviation from instructBlip's original decoder-based framework, emphasizing the versatility and adaptability required in advanced machine learning applications.

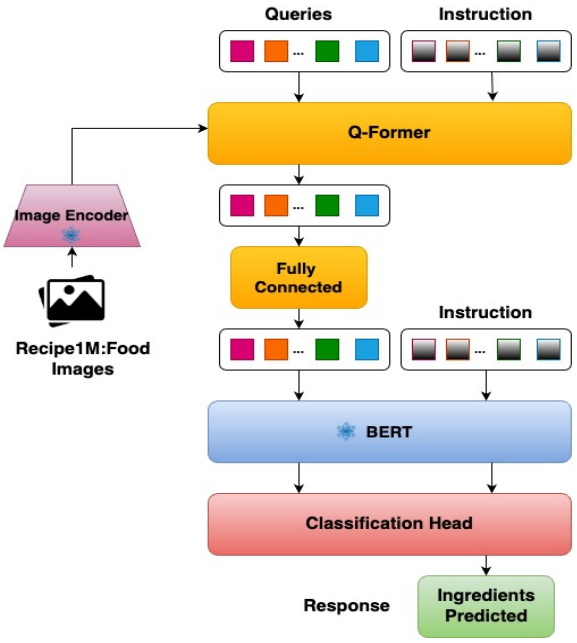

Figure 5: Model architecture for InstructBLIP with a classification head attached to BERT that projects the embeddings onto 1488 ingredients.

5 Experimentation and Evaluation

5.1 InverseCooking

Drawing inspiration from Inverse Cooking, we evaluate our models using Intersection over Union (IoU) and F1 Score. These metrics are computed by accumulating true positives (TP), false negatives (FN), and false positives (FP) across all images in the Recipe1M dataset splits. Using IoU and F1 provides a holistic view of model performance on ingredient prediction over the entire dataset, rather than just per-image metrics. This follows evaluation protocols established in the Inverse Cooking domain, allowing our results to be fairly compared to prior benchmark methods for recipe generation from food images. The accumulative counts used to calculate IoU and F1 allow for assessing how well a model predicts the comprehensive set of ingredients needed for a large collection of recipes.

Table 1 displays the evaluation results from Inverse Cooking, showing IoU and F1 scores for all their experimented models on the ingredient prediction task. In our work, we focus only on their TF_{set} model as our baseline. TF_{set} uses a set transformer architecture to model ingredient co-occurrences through the auto-regressive transformer, while also satisfying order invariance in its set prediction output. This makes it well-suited for modeling dependencies without imposing any order, aligning with our goal. Furthermore, TF_{set} achieves the overall best performance in Inverse Cooking, highlighting the benefits of its dependency modeling and order-invariant set prediction for this recipe generation task. By comparing it, we can fairly evaluate the efficacy of our proposed InstructBLIP variants on modeling such food ingredient relationships.

5.2 InstructBLIP

The evaluation of our experiments began with zero-shot testing using Vicuna-7b. The model tended to respond as if it were performing image captioning. When presented with a food image and prompted with *"Question: what are the ingredients I need to make this food? Answer:"*, the model often gave a general description of the dish or listed only a few visible ingredients, rather than detailing the specific ingredients required. This outcome was somewhat anticipated, as the nature of this task involves ingredients that can appear differently in shape or color across dishes, and many ingredients (like salt, pepper, oil) are not visually evident in the final dish image. As such, general visual-language models struggle to make accurate predictions in this context. As shown in Figure 6, there was a significant discrepancy between the model's predictions and the ground truth. Further, zero-shot testing with the

Model	IoU	F1
FF_{BCE}	17.85	30.30
FF_{IOU}	26.25	41.58
FF_{DC}	27.22	42.80
FF_{TD}	28.84	44.11
TF_{list}	29.48	45.55
$TF_{list+shuf}$	27.86	43.58
TF_{set}	**31.80**	**48.26**

Table 1: **Model Selection (val)** Global ingredient Evaluation Metrics: IoU & F1[9].

Groundtruth	Prediction	Zero-shot from Vicuna
Tomatoes	Tomatoes	Tomatoes
spaghetti	spaghetti	spaghetti
Onions	Red Onions	-
garlic cloves	garlic cloves	-
red wine vinegar	red wine vinegar	-
fresh basil	fresh basil	-
olive oil	olive oil	-
olives	-	-
-	black pepper	-
fresh parsley	-	-
capers	-	-
hot red pepper flakes	-	-
dried oregano	-	-
pecorino cheese	-	-

Figure 6: $instructBlip_{BERT}$ ingredient prediction example

Recipe1M test set (as depicted in Table 2) yielded low F1 and IoU scores of 0.13 and 0.079, respectively. Similar trends were observed with Flan-t5-xl, which even underperformed Vicuna in this regard.

Another challenge for decoder models in this task is set prediction. Decoders, being auto-regressive, predict the next token conditioned on the previously generated tokens. However, in set prediction, this training approach is not optimal. For instance, if the ingredients are 'salt', 'pepper', and 'onion', the order of appearance can affect the conditioning — 'pepper' would be conditioned on 'salt', and 'onion' on both 'salt' and 'pepper'. This can lead to confusion during training. Indeed, when instructBlip was trained on Recipe1M, both F1 and IOU scores deteriorated, and the model struggled to generate coherent sentences. This experiment demonstrated that even well-pretrained, large decoder models are unsuitable for multi-label classification tasks, both in zero-shot scenarios and fine-tuning.

Model	IoU	F1
TF_{set}(baseline)	31.80	48.26
$instructBlip_{vicuna}$(zero-shot)	7.9	13.0
$instructBlip_{BERT}$	**34.01**	**50.76**

Table 2: Comparative Evaluation of TF_{set} vs $InstructBlip_{BERT}$ and $InstructBlip_{vicuna}$

In our second experiment using BERT, Figure 8 shows that the loss converged well without over-

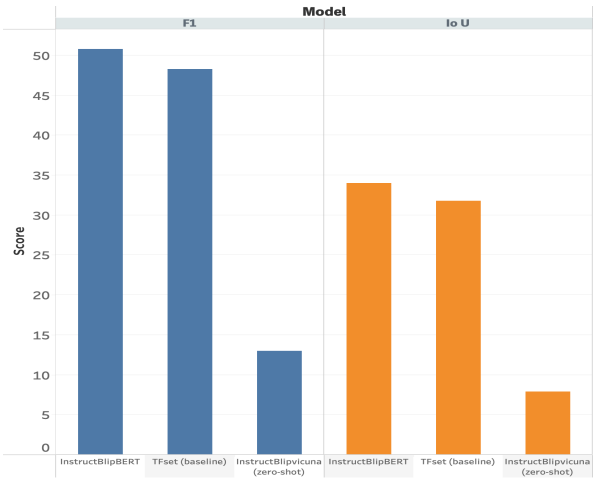

Figure 7: Visual Representation of IoU and F1 Comparison of TF_{set} model vs $InstructBlip_{BERT}$ and $InstructBlip_{vicuna}$

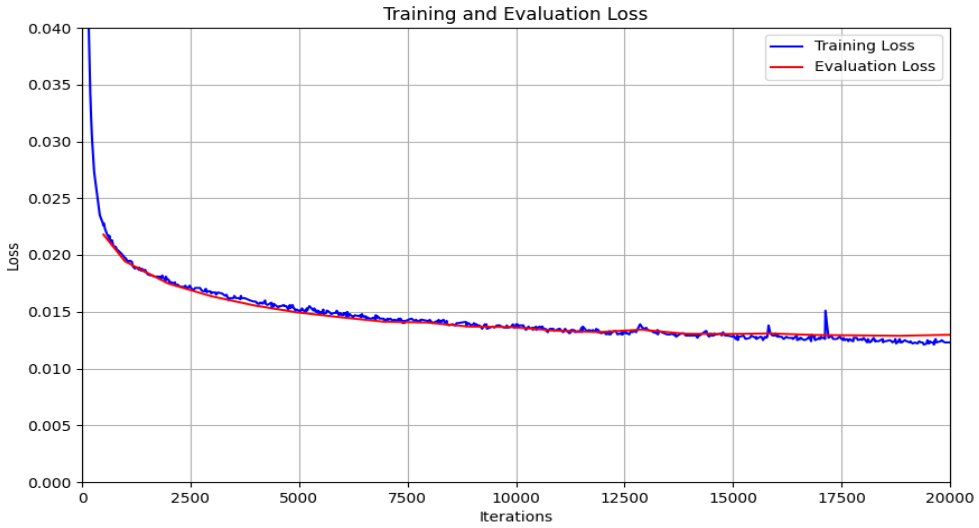

Figure 8: Iteration versus Loss in instructBlip Training and Evaluation

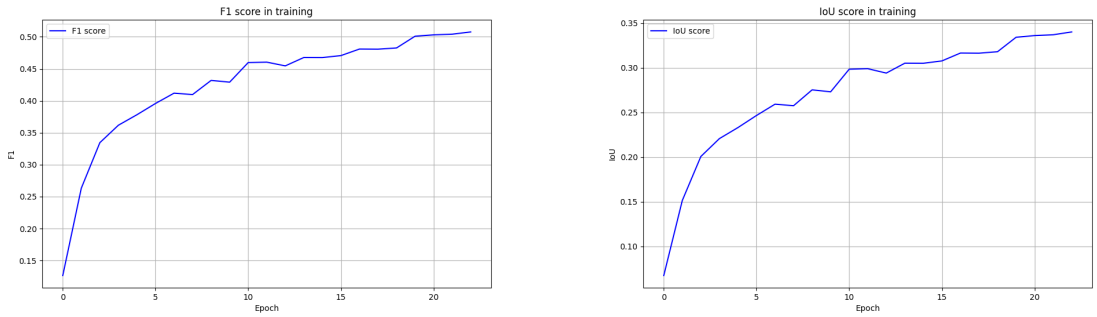

Figure 9: F1 and IoU score over time in training

fitting. Here is a generated example 6, where each ingredient name represents a cluster from the 16K ingredients condensed into 1488 clusters, named with the assistance of GPT-4. This figure illustrates

the model's capability to predict ingredients that are not visibly apparent in the image, indicating the use of language model knowledge to understand the context of ingredients and infer those not shown in the picture. As a result, both F1 and IOU scores were higher than the baseline set by inversecooking, as shown in Table 2.

Compared to the Vicuna-7b used in the decoder experiment, BERT required less than half the GPU memory, resulting in significantly faster training and inference speeds. This experiment highlighted that, even with a smaller size, encoders are more suited for multi-label classification tasks.

6 Conclusion

In this work, we conducted experiments to predict the list of ingredients from food images. This task is inherently a multi-label classification challenge. Our analysis revealed that, in the context of inverse cooking, employing a set transformer to model ingredient co-occurrence and satisfy order invariance yielded better results compared to using a normal transformer in an auto-regressive manner. Inverse Cooking requires training both the vision encoder and language model in two stages, involving numerous parameters, which necessitates a large dataset and extended training duration. Moreover, it cannot directly utilize pretrained vision encoders or language models, which is a notable limitation.

To overcome these drawbacks, we leveraged the instructBlip model, which effectively connects a pretrained vision encoder and language model through an additional module. Initially, instructBlip was based on a decoder architecture; however, decoders were found to be ill-suited for multi-label classification tasks. Large decoders like Vicuna-7b could predict ingredients in a zero-shot scenario, but they tended to identify only the most obvious ingredients visible in the food images and struggled to predict detailed ingredient lists even after training.

Therefore, we adopted a model linking a vision encoder with an encoder-only language model, following instructBlip's Q-former approach. Similar to instructBlip's training method, we trained only the Q-former, keeping the vision encoder and language model frozen. This strategy allowed us to harness the extensive knowledge encapsulated in the large, pretrained vision encoder and language model. The results of this approach demonstrated higher F1 and IoU scores compared to the baseline established by inverseCooking, highlighting the effectiveness of our modified instructBlip model in the context of ingredient prediction from food images.

7 Future Work

This research demonstrates promising results in applying InstructBLIP models for food image analysis and ingredient prediction. There's significant potential for future advancements that can build upon these initial findings in several areas:

- *Customized query transformer development:* Our existing Q-Former extract useful visual information from visual features and project it into text space. Designing a specialized query transformer tailored for food images and cooking recipe tasks could potentially improve performance.
- *Incorporating additional modalities:* Future multimodal models could extract information from food images, videos, and text to better mimic real-world cooking environments.
- *Applications for recipe generation interfaces:* Extending these models to also perform recipe generation may potentially benefit applications like cooking assistants, smart home devices, and other interactive tools could enhance user recipe ideation, planning, and creation.

7.1 Limitations

During the execution of this study, it's essential to acknowledge the limitations encountered, notably stemming from restricted GPU computational power. The constraints imposed by limited GPU resources significantly impacted the scale and efficiency of model training and experimentation. The computational demands of training complex neural network architectures, especially those employing transformer models and sophisticated loss functions, posed substantial challenges given the hardware

limitations. Due to restricted GPU availability, the experimentation process faced constraints in hyperparameter tuning, extensive training iterations, and exploring larger model architectures. These computational restrictions inevitably influenced the scope of the study, limiting the dataset sizes that could be processed effectively and the depth of exploration into certain model configurations. Despite these constraints, efforts were made to optimize resource utilization and explore methodologies balancing efficiency and research objectives.

Ultimately, the GPU hardware limitations constrained the complexity of the models and training procedures, potentially restricting the generalizability of the results. Future research with more powerful computational resources could investigate larger and more complex models to further explore the potential of this approach.

8 Acknowledgements

The authors would like to express their sincere gratitude to Professor Ilias Tagkopoulos for his invaluable guidance and support towards the completion of this research project. As the professor for graduate-level course 289G Deep Learning, his extensive expertise in advanced deep learning has been instrumental in shaping this work. From enlightening lectures elucidating complex architectural details to insightful discussions sparking new research directions, Professor Tagkopoulos's mentorship has strengthened the methodologies and outcomes presented here.

References

[1] Jing-jing Chen, Chong-Wah Ngo, and Tat-Seng Chua. "Cross-modal recipe retrieval with rich food attributes". In: *Proceedings of the 25th ACM international conference on Multimedia*. 2017, pp. 1771–1779.

[2] Jingjing Chen and Chong-Wah Ngo. "Deep-based ingredient recognition for cooking recipe retrieval". In: *Proceedings of the 24th ACM international conference on Multimedia*. 2016, pp. 32–41.

[3] Weiwei Cheng, Eyke Hüllermeier, and Krzysztof J Dembczynski. "Bayes optimal multilabel classification via probabilistic classifier chains". In: *Proceedings of the 27th international conference on machine learning (ICML-10)*. 2010, pp. 279–286.

[4] Wenliang Dai et al. *InstructBLIP: Towards General-purpose Vision-Language Models with Instruction Tuning*. 2023. arXiv: 2305.06500 [cs.CV].

[5] Junnan Li, Hoi Steven, and Rose Donald. *Blip: Bootstrapping Language-Image Pre-Training for Unified Vision-Language Understanding and Generation*. 5 Nov. 2022. URL: blog.salesforceairesearch.com/blip-bootstrapping-language-image-pretraining/.

[6] Dhruv Mahajan et al. "Exploring the limits of weakly supervised pretraining". In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 181–196.

[7] S Hamid Rezatofighi et al. "Deepsetnet: Predicting sets with deep neural networks". In: *2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE. 2017, pp. 5257–5266.

[8] S Hamid Rezatofighi et al. "Joint learning of set cardinality and state distribution". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 32. 1. 2018.

[9] Amaia Salvador et al. "Inverse cooking: Recipe generation from food images". In: (2019), pp. 10453–10462.

[10] 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, pp. 3020–3028.

[11] Grigorios Tsoumakas and Ioannis Vlahavas. "Random k-labelsets: An ensemble method for multilabel classification". In: *European conference on machine learning*. Springer. 2007, pp. 406–417.

[12] Jiang Wang et al. "CNN-RNN: A Unified Framework for Multi-Label Image Classification". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2016.