# AN IMAGE CAPTIONING APPROACH TO FOOD IDENTIFICATION

**Adam Sun**
Stanford University
adsun@stanford.edu

**Maggie Sun**
Stanford University
mgsun@stanford.edu

**Amy Guan**
Stanford University
amyguan@stanford.edu

## 1 Introduction

With an endless canon of global cuisines, the nature of human recipe identification is often relegated to a locality's cooking techniques and ingredients. It is a nearly unfeasible task for a human to become familiar with a plethora of dishes from around the world, let alone the endemic ingredients, spices, and techniques that go into them. The data processing capabilities of artificial intelligence and image recognition, however, have the potential to amass a large basis of knowledge surrounding cuisine.

Food image identification is a task that can be incredibly useful across a variety of applications. Being able to produce the name of a dish given an image unshutters implicit information such as calorie information or ingredients. However, with an essentially unlimited amount of dishes in the world from a variety of global cultures, training a model with individual categories for each dish is quite unrealistic.

Since a dish is composed of distinct ingredients, we propose to frame the food image identification problem as an image captioning problem instead of a classification problem. In other words, the model will take an image of a dish as input and generate a dish name as output. Instead of explicitly defining categories for foods and limiting the set of possible foods generated, our approach uses an encoder-decoder architecture that is able to use features extracted from an image to generate a relevant and realistic caption for the image. This allows the model to not only learn captions for dishes it was trained on, but also identify dishes that it has not seen before by identifying distinct ingredients and cooking methods present in the image.


Figure 1: An example of a potential input to our model. The model would ideally see a picture of a dish and output "Beef and Broccoli". Despite not seeing examples of this particular dish in the training phase, we hope our model is able to recognize beef and broccoli separately, understanding the relationship between the two and combining the two terms in the output.

## 2 Literature Review

There exist a few other methods to identify images of food. These largely fall into two separate categories: food classification, and recipe generation.

The Food-101 dataset consists of 101,000 images of different foods from 101 different categories. Examples of such categories include "baby back ribs", "eggs benedict", and "lobster bisque" [3]. Models trained on this dataset are trained to classify among these 101 categories. In [3], a random forest was used to mine discriminative components, while in [4], a neural embedding network was combined with a convolutional neural network. In [6], a deep CNN-based model is used to classify examples of food. In [7], a new dataset known as Food2K with 2,000 different categories of dishes is proposed.

Our approach is similar to [4] and [6] in that both of our approaches use CNNs in order to extract features from images. However, while these approaches can be reliable, they are limited to a set number of categories. Their performance is largely dependent on the set of categories in the dataset. This can limit the performance of the model. On the other hand, our approach is able to generate new captions for each image.

[8] uses the Recipe1M dataset to determine a recipe from an image of food. The task for these models involves generating recipes from images. This approach is complementary to ours in that it also uses an encoder-decoder architecture with a CNN encoder and a transformer decoder that generates a sequence of text.

## 3 Dataset

Our dataset consists of a combination of pre-existing datasets and our own webscraped dataset. Because the project lies in the realm of image-captioning, we searched for datasets containing descriptive names as opposed to over-simplified generalizations. For example, we looked for data that would categorize "butternut squash soup" as opposed to just "soup". To do this, we focused on collections that name based on the food (as opposed to the cooking process). We decided to use a dataset from Epicurious from Kaggle [1] for its cleanliness of dish names.

To generate a more comprehensive dataset, we also scraped data following these same principles. We built a web crawler for an existing recipe site, which collected dish names and the associated images from simplyrecipes.com.

Combining the two datasets results in a training set of 13,131 images and a validation set of 3,284 images. Each data point consists of a feature the image of the food, and the label (the ground truth name of the dish).

To test the model, we feed it images of dishes it has never seen before. To create the test dataset, we searched "recipes" on Pinterest, selected 24 images of interest, and used the recipe title or description as the ground truth label.

For our oracle, we asked a human to blindly annotate our test set.

We preprocessed the captions to standardize across all foods. Specifically, we removed capitalization, punctuation, and stopwords (such as 'An'). Additionally, since our dataset was sourced from recipe sites, we accounted for some common recipe-specific naming conventions (such as "The Best") by removing extraneous phrases not applicable to general food captioning.

We also generated some metrics on the dataset. The most frequently used words include prepositions and conjunctions such as "with" or "and," as well as food items such as "chicken" and cooking methods such as "roasted." The most frequently used bigrams include many bigrams of the form "$X$ with" (where $X$ is a food item) such as "salad with" and "chicken with" alongside common two-word food items such as "sweet potato."

From the analysis of the most frequently used words and bigrams, the dataset has clear trends of being skewed towards Western ingredients and dishes, as well as dishes with compound naming. Additionally, certain common foods, such as beef are underrepresented. This may be due to certain foods being disproportionately popular for cooking and thus more commonly represented in recipe datasets. For example, chicken is an accessible and relatively easy item to cook and thus more likely to appear in recipes.

## 4 Baseline

As our baseline, we test the transfer learning capabilities of a state-of-the-art model trained on a more general dataset. We use a pretrained Vision Transformer (ViT) encoder combined with a GPT-2 decoder with Hugging Face's Vision-to-

[1] https://www.kaggle.com/datasets/pes12017000148/food-ingredients-and-recipe-dataset-with-images

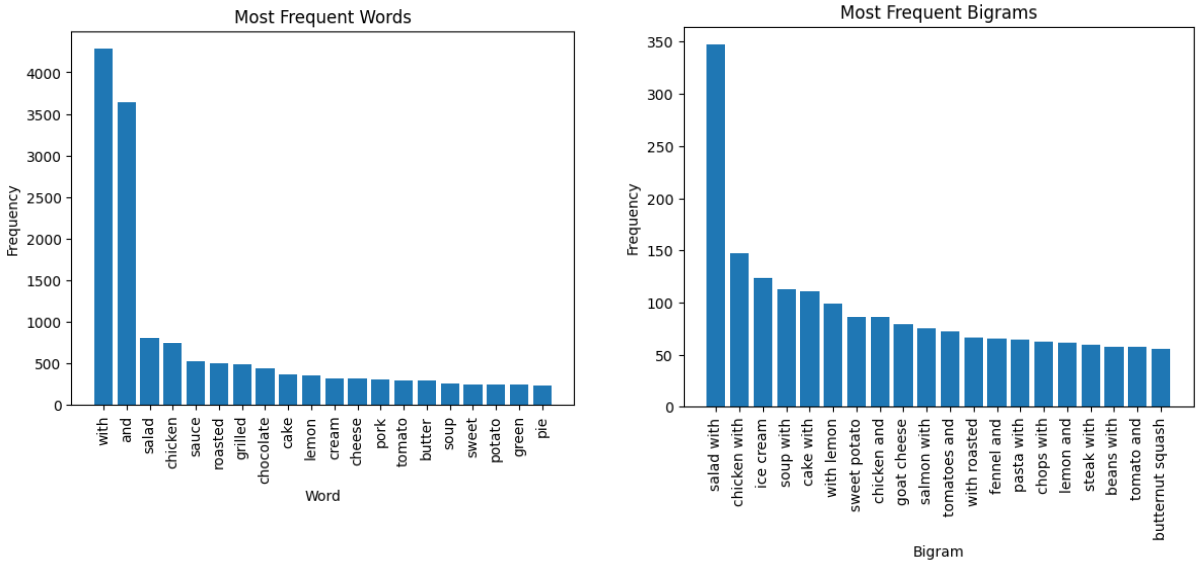


Figure 2: Top 20 most frequently appearing words and bigrams within the captions of the dataset.

Text Encoder-Decoder Framework. The model has been trained on the COCO-2017 image captioning dataset for 6900 steps with batch size 256. It can be found at https://huggingface.co/spaces/flax-community/image-captioning[9].

The COCO-2017 [5] dataset is a large-scale captioning dataset consisting of 330K images with 5 captions per image. The wide majority of images in this dataset do not involve food. As a result, most of the captions generated by a model trained on this dataset are too general, frequently being "a plate of food". Therefore, we foresee the model's captions for our test set to be way too general for our purposes, thereby acting as our baseline.

## 5 Main Approach

For our main approach, we combine a convolutional neural network encoder with a transformer decoder. The convolutional neural network we use is EfficientNetB0 pretrained on ImageNet, and the decoder consists of a 2-layer transformer consisting of causal self attention layers, cross attention layers, and feed forward layers. We use code from https://www.tensorflow.org/tutorials/text/image_captioning to construct our model [1].

During training, we tokenize the captions into sequences of integers, with each integer representing the corresponding index in the vocabulary, whose size we set at 5000. The convolutional neural network takes a 224x224x3 RGB image of food as input, and transforms the images of food into feature maps. We freeze the entire CNN during training, prompting it to serve only as a feature extractor. Then, we use the feature maps as input to a transformer, which generates an output caption for the image.

In accordance with [1], we experiment with three different values of the temperature parameter $t - 0, 0.5, 1$ for determining the generated words. A temperature of 0 indicates greedy decoding, choosing the most likely next word at each step. A temperature of 1 samples according to the logits, and a temperature of 0.5 samples based on twice the logits. We use the encoder model as a feature extractor, freezing it during the training process, only training the transformer decoder. We use masked loss and masked accuracy in order to mask out the padding, since our model outputs sequential data [9].

We train for 15 epochs on our dataset with Adam optimizer and a learning rate of 1e-4 and a batch size of 64.

## 6 Evaluation Metric

To evaluate the performance of each approach, we used an MPNet model that was finetuned on 1 billion sentence pairs found on Hugging Face [2]. Using this model, we encoded each caption into a feature vector representation of the caption. We then use the cosine similarity metric to measure the similarity of each approach (baseline, oracle, and our approaches) to the ground truth.

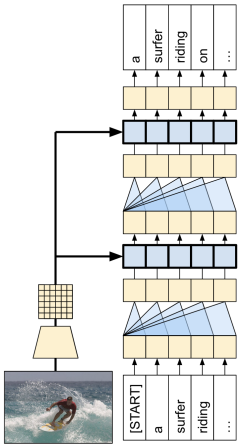


Figure 3: The model architecture. The image encoder transforms the input image into a feature map that is then used by the transformer to generate an output caption. [9]

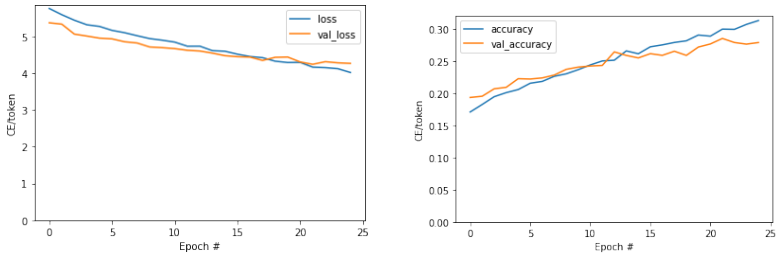


Figure 4: Training and validation loss and training and validation accuracy over epochs. The model begins to gradually overfit at around epoch 25.

The cosine similarity of two feature vectors $A$ and $B$ is defined as

$CosineSimilarity(A, B) = cos(\theta) = \frac{A \cdot B}{||A||||B||}$.

While we initially experimented with different text similarity metrics, such as the Dice Coefficient, we found that these metrics only had the capability to measure whether exact words are shared between two strings. This metric is too strict for our purposes, since we desire captions generated to be similar in meaning, not necessarily in exact words.

## 7 Results and Analysis

| | Baseline | Oracle | Ours (t = 0) | Ours (t = 0.5) | Ours (t = 1) |
|---|---|---|---|---|---|
| Mean | 0.417 | 0.799 | 0.452 | 0.447 | 0.381 |
| STD | 0.169 | 0.192 | 0.170 | 0.176 | 0.126 |

Figure 5: Mean and STD of cosine similarity across each method.

We can see from the distributions of the results that our model at $t = 0$ and $t = 0.5$ is an improvement from the baseline.

In Figure 6, we see that the cosine similarity for the baseline is centered around 0.5, which suggests that the baseline generally achieves a moderate similarity with the ground truth caption. Meanwhile, the cosine similarity of the oracle is centered around 0.9, with a few instances of low cosine similarity. This suggests that the oracle generally generates captions that are very close to the ground truth, although with a risk of outliers on the left tail, similar to having a left-skewed underlying distribution.

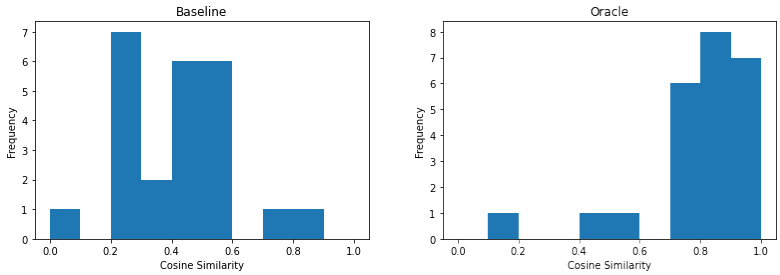


Figure 6: Distribution of cosine similarities for baseline and oracle across our test set. The baseline is centered at around 0.5, while the oracle is mostly centered around a high cosine similarity.

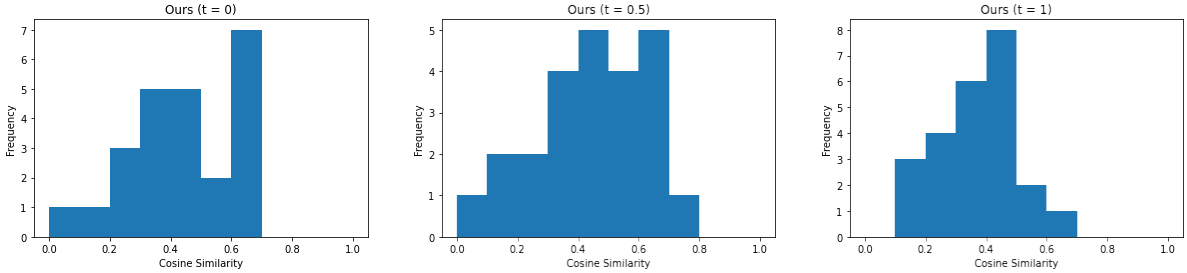


Figure 7: Distribution of cosine similarities for each method across our test set.

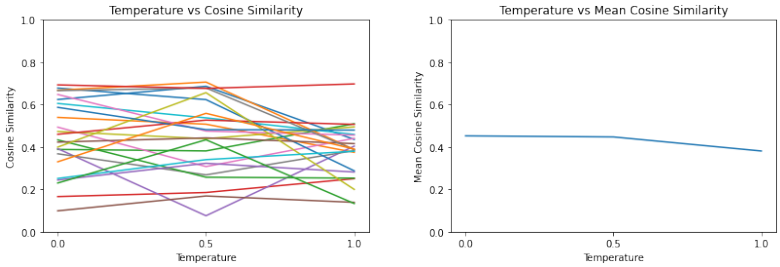


Figure 8: Graphs of temperature vs cosine similarity across all test examples. The mean cosine similarity decreases slightly as temperature increases.

In Figure 5 and Figure 7, we see that the cosine similarities for $t = 0$ and $t = 0.5$ have similar mean and standard deviation. Both of these have a higher mean cosine similarity than that of the baseline. Generally, the cosine similarities distributed similarly as well.

The cosine similarities for $t = 1$ are more concentrated between 0.3 and 0.5, making it slightly right skewed. The mean for $t = 1$ is also lower than that of baseline, $t = 0$, and $t = 0.5$. This suggests that the model with temperature parameter $t = 1$ is expected to generally underperform.

Figure 8 reveals that the mean cosine similarity decreases slightly as temperature increases. This trend reveals that it is optimal to be greedy when picking the next word when generating a caption. However, looking at each test example, it can be seen that there is a significant distribution across each example. There does not seem to be a general trend that most examples conform to as the temperature increases.

## 8 Error Analysis

The variation in performance of our model revealed by the distributions reveals that our model is not very robust to different examples and should not be relied upon for critical applications. For example, our model should not be used by people with high-risk needs, like food allergies or strict dietary restrictions. While our model does perform somewhat well on this food captioning task, it is not yet reliable to be used in the real world, where there might be more severe consequences for misidentifications.

There are specific categories of food that are more prone to misidentification: varied proteins, non-Western foods, and desserts. As depicted in Figure 9, proteins tend to be classified as chicken. The model likely captured characteristics of proteins and attributed it to chicken as opposed to other proteins. This is likely due to the underlying dataset, which had a disproportionate inclusion of chicken.

Additionally, in Figure 9, it is evident that the model is ineffective at classifying non-Western foods. Generally, while non-Western foods were included in the dataset, Western foods greatly outnumbered them. This means that the model likely intereted non-Western food descriptors as more unlikely labels or did not have enough data to properly attribute them to image characteristics.

Finally, in Figure 9, desserts were more likely to be mislabeled, although the general category is correctly attributed. Again, there was an underrepresentation of dessert as opposed to meal foods, which likely means that there was not enough representative data for the model to learn off of.

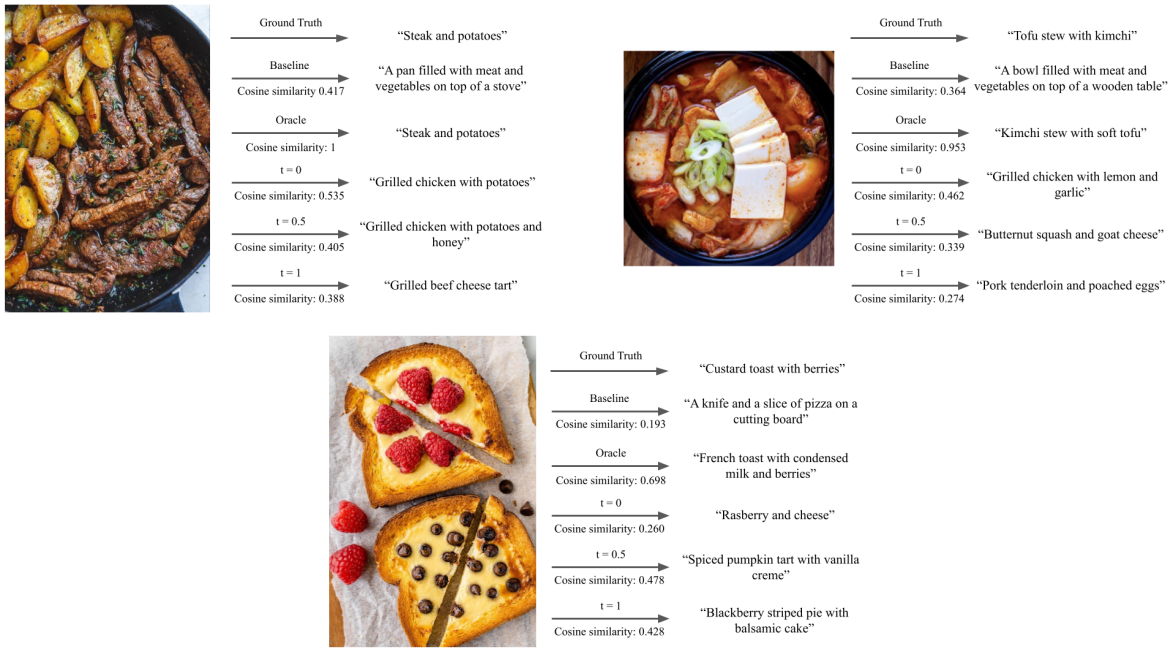


Figure 9: A comparison of ground truth labels with our baseline, our oracle, and our generated captions ($t = 0, 0.5, 1$). The baseline gives a very general description of the image, which makes sense since it is trained on a very general dataset of images. Our oracle reliably described what she saw in the image, although with slightly altered phrasing. The model misidentifies certain parts of the images, especially when it comes to non-chicken meat and miscellaneous ingredient add-ons (such as misidentifying tofu as cheese).

## 9 Future Work

Given our limited computing resources and time, we were not able to collect a more extensive dataset and were not able to experiment with more powerful models. We found that the training of the model was limited by the recipes hosted on the websites we scraped. Recipes that host American-centric recipes only elucidate information about Western ingredients and cooking techniques, and we found that our model had more difficulty naming dishes of non-Western origin.

Future work would firstly focus on collecting a more robust body of data, allowing the model to recognize and understand more ingredients and cooking techniques. Scraping websites for non-European or American dishes would be a useful step to contributing to the breadth of the model's recognition abilities.

We would also like to collect more data overall and invest more into standardizing data across sources. For example, more preprocessing could be done on eliminating recipe-centric naming conventions outside of simply excluding stopwords. Furthermore, diversifying data sources to outside of just recipe datasets would help reduce the overrepresentation of foods that are more likely to be in recipes or home-cooked meals (such as chicken). Furthermore, recipe datasets may

have certain stylistic naming conventions that may not directly correspond with general human-generated food captions, so diversifying outside of recipe datasets would also help incorporate a wider range of stylistic choices.

## 10 Resources

The data can be found at https://drive.google.com/file/d/1MZ3NFTPpyvqzCW88W-7ZB7KowiA8lc1B/view?usp=sharing.

The code can be found at https://github.com/Asun13/food-captioning.

## References

[1] Image captioning with visual attention nbsp;: nbsp; tensorflow core.

[2] Sentence-transformers/all-mpnet-base-v2 · hugging face.

[3] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In *European Conference on Computer Vision*, 2014.

[4] Kuang-Huei Lee, Xiaodong He, Lei Zhang, and Linjun Yang. Cleannet: Transfer learning for scalable image classifier training with label noise. *CoRR*, abs/1711.07131, 2017.

[5] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014.

[6] Chang Liu, Yu Cao, Yan Luo, Guanling Chen, Vinod Vokkarane, and Yunsheng Ma. Deepfood: Deep learning-based food image recognition for computer-aided dietary assessment. *CoRR*, abs/1606.05675, 2016.

[7] Weiqing Min, Zhiling Wang, Yuxin Liu, Mengjiang Luo, Liping Kang, Xiaoming Wei, Xiaolin Wei, and Shuqiang Jiang. Large scale visual food recognition. *CoRR*, abs/2103.16107, 2021.

[8] Amaia Salvador, Michal Drozdzal, Xavier Giró-i-Nieto, and Adriana Romero. Inverse cooking: Recipe generation from food images. *CoRR*, abs/1812.06164, 2018.

[9] Yih-Dar Shieh. Image captioning - a hugging face space by flax-community.