RecipeNet: Image to Recipe/Nutritional Information Generator

Dorian Raboy-McGowan
Computer Science
Stanford University
dorianrm@stanford.edu

Sabrina Lu
Mathematical & Computational Science
Stanford University
slu12@stanford.edu

Luciano Gonzalez
Computer Science
Stanford University
lucigon@stanford.edu

Abstract

Food plays an instrumental role in our everyday lives and in many ways dictates our health and experiences. This paper introduces RecipeNet, a food to recipe generator that is trained on the Recipe 1M dataset to output recipes for any given food image input. Our model used ResNet-50, ResNet-101, and DenseNet-121 convolutional neural networks in order to classify images of food and encode their features, then uses K-nearest neighbors to output recommended recipes from the Recipe 1M data set. The outputted recipes come with nutritional labels that help guide users to make the choices that are best for their nutrition goals. Our model successfully recommends relevant recipes from a given input image with DenseNet-121 having the highest average cosine loss of 0.719 between images associated with the same recipe.

1 Introduction

Food is fundamental to the human experience. It has deep ties to our health, our livelihood, our emotions, and our culture. More and more these days, people want to learn about what they are eating in order to make better nutritional decisions, but many struggle to find a place to start. Our application aims to empower people to better understand and have control over their food intake in order to help achieve their health goals. The goal of our project is to create a system that accepts an image of a meal as input and outputs closely related recipes as well as nutritional information. Given an input image, RecipeNet returns several related recipes to give options for our user to pick from, and each of these recipes comes with nutritional labels such as "under 500 calories," "vegetarian," and "protein heavy" in order to guide people to make the best nutritional decisions for themselves. Our platform could help people attain a more sophisticated understanding of health and diet, and enable them to express themselves through new recipes and cooking techniques.

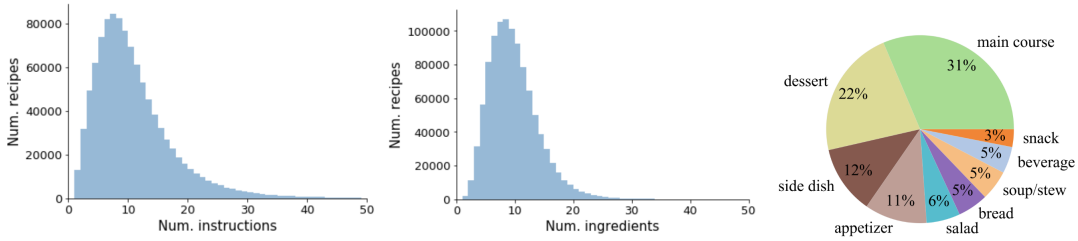
2 Related Work

In developing our project, we found it extremely helpful to investigate projects aimed at accomplishing similar tasks in order to guide us in the right direction and gain a better sense of what came before us. We found that in the past decade, there has been significant progress in the collection of food-recipe datasets. In the early stages, most works followed a classical recognition pipeline, focusing on feature combination and specialized datasets (mostly Asian cuisine). However, in 2014, Bossard et al. took a step forward and introduced the holistic Food-101 visual classification dataset with 101k images divided equally among 101 classes, setting a baseline accuracy of 50.8% [1]. Two years later in 2016, Chen and Ngo presented another large dataset with 65k recipes and 110k images, but it only covered Chinese cuisine [2]. In 2017, Salvador et al. released the Recipe1M+ dataset, which was a new large-scale, structured corpus of over one million cooking recipes and 13 million food images [3]. To date, this is the largest publicly available collection of food and recipe data, and allows for the ability to train high capacity models on aligned, multi-modal data. Therefore, this is the dataset we chose to use for our project.

There have also been a variety of approaches used to tackle the food recognition issue. In 2010, Yang et al. proposed to learn spatial relationships between ingredients using pairwise features [4]. However, this was bound only to work for standardized meals. Along with the Food-101 dataset, Bossard et al. introduced a novel method to mine discriminative parts using Random Forests. Random Forests was used to discriminatively cluster superpixels into groups (leaves) and then used the leaf statistics to create features. The researchers applied a distinctiveness measure to the leaves to merge similar leaves, then used a support vector machine (SVM) to classify the food images into categories. A few years later, Herranz et al. proposed an extended multi-modal framework that explores how visual content, context, and external knowledge can be integrated into food-oriented applications. [5]. Around the same time, Min et al. presented a multi-attribute theme modeling approach that incorporates attributes such as cuisine style, course type, flavors or ingredient types [6]. The model learns a common space between different food attributes and their corresponding food images. Finally, there have some interesting papers published specifically in the realm of extracting recipes from images. Chen et al. found that the manner of cutting and cooking ingredients play substantial roles in the food's appearance, and therefore attempted to predict ingredient, cutting, and cooking attributes given a food image [7]. On the other hand, Chang et al. looked at the possibility of several different preparations for one dish by clustering recipes based on their distance [8].

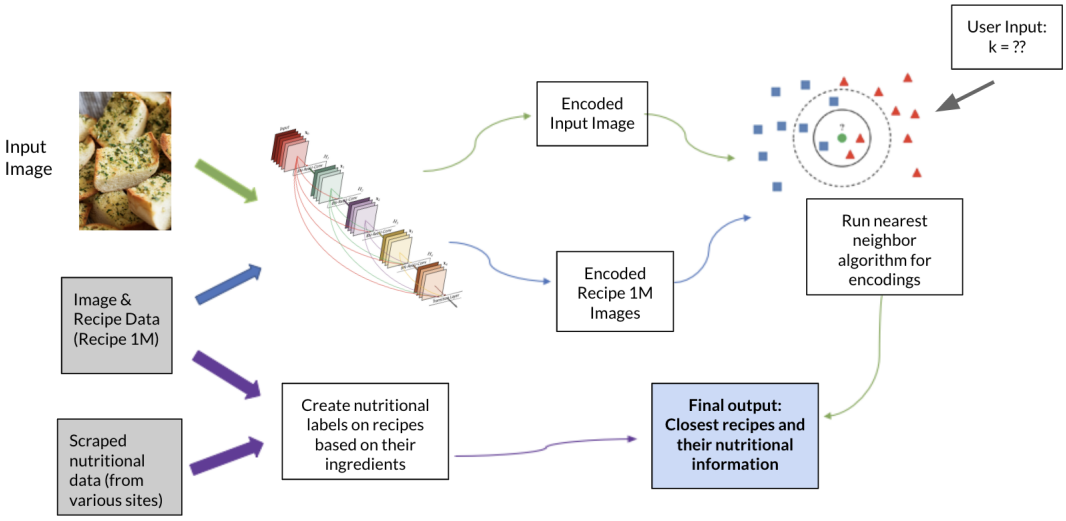
3 Dataset

Because the Recipe1M+ dataset contained 1M recipes with 13M associated images, we decided that we would take a subset of the dataset (the Recipe1M dataset) instead due to computational and time constraints. Our Recipe1M dataset includes $402,760$ recipes with $887,536$ associated images, which was the perfect size for our project. Below are some statistics on the Recipe1M+ dataset, which gives a general idea of our Recipe1M subset.



Besides the training images, the Recipe1M dataset was split into two dataframes. The first dataframe contains the id, ingredients, title, instructions, and original url location of all 1M recipes in the Recipe1M+ set. The second dataframes contains the ids of only the recipes in our Recipe1M set as well as a list of their associated image ids. We preprocessed the second dataset such that all the individual image ids were extracted and we instead had a dataframe that matched the image id to their respective recipes. There was a also a third separate dataset created by the authors of Recipe1M that contained nutritional information on a subset of about 51k recipes. This dataset, combined with data scraped from various websites on vegetarian/vegan ingredients, was used to preprocess the recipes to create the nutritional labels. We also preprocessed the images such that they were resized to have the shape (256, 256, 3) before we ran them through any of the CNN models.

4 Methodology



To build our system, we needed some way to determine how similar two images were, so that we could match an input image to the most similar image in our dataset. Direct pixel value comparisons are a poor method of evaluating similarity, mainly because we are concerned with the similarity in image content, not colors and object positions. From what we learned in class, we knew that later layers in a CNN could be used as an image encoding. Since each activation in later layers corresponds to some higher-level or abstract feature of the image, we knew we could use these encodings as a way to capture the content of images while ignoring pixel-value differences caused by lighting, object position, discoloration, etc.

With the goal of producing image encodings for every image in our dataset, we ran the Recipe 1M training data through a variety of convolutional neural networks, ResNet-50, ResNet-101, and DenseNet-121, each trained on the ImageNet dataset.

With our entire dataset encoded through the various CNNs, next came processing the user input. The first step is to generate an encoding of the input image by passing the input image through the same CNN used to encode the dataset. Once we have this encoding, we search linearly through the encoded dataset and calculate the k nearest neighbors to the input image from among the encoded dataset. The k is determined by the user depending on how many recipes they would like to see. Our system supports running nearest neighbors with either cosine similarity or euclidean distance as the distance metric between two encodings. Once the nearest neighbors are found, the recipes related to these images are displayed along with nutritional information about these recipes.

In order to get nutritional information and labels, we scraped information on ingredients that violate vegetarian, vegan, and pescetarian diets from various site and compiled them into lists. We then looked into the ingredients of each recipe and compared them to the respective lists, labeling each recipe "vegetarian," "vegan," or "pescetarian" if the ingredients did not overlap. In addition, the Recipe 1M authors had created a dataset for $51,235$ of the recipes extracting nutritional information such as fat, sodium, and protein content for the ingredients within the recipes. We used this information to create labels such as "$<$ 500 Calories," "Protein Rich," "Protein Heavy," and "Low Sodium" for these recipes and appended them to the existing labels. With these added labels, our users would be able to make conscious decisions that adhere to specific diets or lifestyles.

5 Model/Methods

For our project we have experimented with a number of different models and setups. The first step of our projcet was to create encodings of our dataset images, and we opted to create encodings using ResNet-50, ResNet-101, and DenseNet-121 CNN models. We specifically used these models because of their ability to handle the issues which arise in deep neural networks as the number of layers grow. The ResNet models solve the problem of vanishing gradients and slow learning in deeper networks by including residual blocks within its

network architectures. Residual blocks add the previous layers activations to the currents layer's value before activation:

$$a^{[l]} = \mathbf{g}^{[l]}\left((W^{[l]}a^{[l-1]} + b^{[l]}) + a^{[l-2]}\right)$$

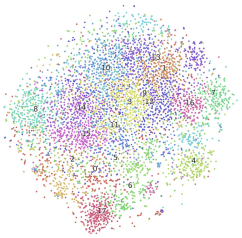
Where $\mathbf{g}^{[l]}$ is layer l's activation function. The DenseNet models use dense blocks within its network architectures in order to solve the slow learning problem. Dense blocks, rather than add the previous layers activations to the current, appends them:

$$a^{[l]} = \mathbf{g}^{[l]}\left([(W^{[l]}a^{[l-1]} + b^{[l]}), a^{[l-2]}]\right)$$

DenseNets are built on the findings from recent work that show that convolutional networks can be substantially deeper, more accurate, and efficient to train if they contain shorter connections between layers close to the input and those close to the output. DenseNets therefore connect every layer directly with every other layer, with each layer taking in the feature-maps of all proceeeding layers as its input. This encourages feature reuse and substantially reduces the number of parameters, all of which makes the model more compact and less prone to overfitting. DenseNets are therefore meant to improve flow of information and gradients throughout the network, which we wanted to investigate.[9]

Both residual and dense blocks allow layers to learn identity mapping between layers, which guarantees that adding layers can't make the network worse. This allows us to make much deeper layers without decreased performance; in the worst case scenario the larger network works the same as a smaller network and at best we get improvements from the additional layers.

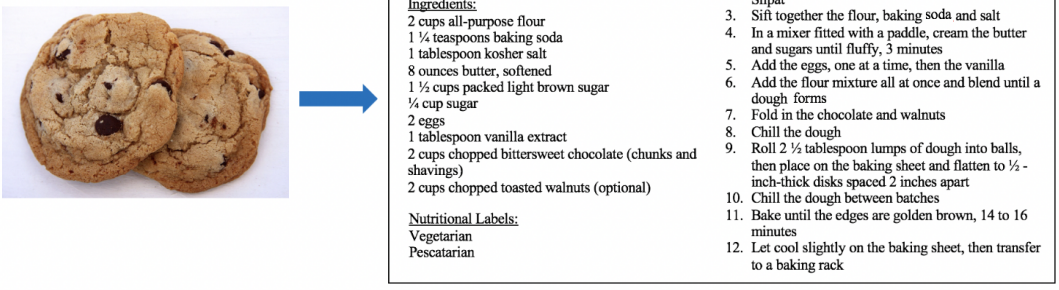
Each of the models we used come pretrained on the ImageNet dataset, trained on the dataset's 1000 classes. For our specific use case, we replace the final, fully-connected layers from these networks with a single avg-pool layer. This is because we do not want a softmax output for our network; instead we want an encoding of the original image with a much lower dimensionality than our original image. For each of these models, our avg-pool output is a 2048 dimensional encoding of the original image: perfect for our task.



Above is a visualization of 1% of our encodings, showing that there are many groupings that can be found within the encodings due to the similarity of certain images.

6 Evaluation/Results

Below are some of the top recipes outputted in running our encodings through ResNet-50 and DenseNet-121 respectively for the input images.

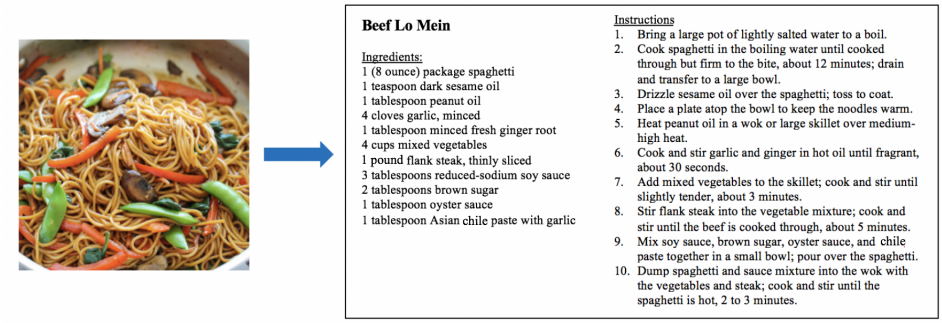


Flat-and-Chewy Chocolate-Chip Cookies

Ingredients:
2 cups all-purpose flour
1 ¼ teaspoons baking soda
1 tablespoon kosher salt
8 ounces butter, softened
1 ½ cups packed light brown sugar
¼ cup sugar
2 eggs
1 tablespoon vanilla extract
2 cups chopped bittersweet chocolate (chunks and shavings)
2 cups chopped toasted walnuts (optional)

Nutritional Labels:
Vegetarian
Pescatarian

Instructions
1. Preheat the oven to 325 degrees.
2. Line two baking sheets with parchment paper or Silpat
3. Sift together the flour, baking soda and salt
4. In a mixer fitted with a paddle, cream the butter and sugars until fluffy, 3 minutes
5. Add the eggs, one at a time, then the vanilla
6. Add the flour mixture all at once and blend until a dough forms
7. Fold in the chocolate and walnuts
8. Chill the dough
9. Roll 2 ½ tablespoon lumps of dough into balls, then place on the baking sheet and flatten to ½-inch-thick disks spaced 2 inches apart
10. Chill the dough between batches
11. Bake until the edges are golden brown, 14 to 16 minutes
12. Let cool slightly on the baking sheet, then transfer to a baking rack



Beef Lo Mein

Ingredients:
1 (8 ounce) package spaghetti
1 teaspoon dark sesame oil
1 tablespoon peanut oil
4 cloves garlic, minced
1 tablespoon minced fresh ginger root
4 cups mixed vegetables
1 pound flank steak, thinly sliced
3 tablespoons reduced-sodium soy sauce
2 tablespoons brown sugar
1 tablespoon oyster sauce
1 tablespoon Asian chile paste with garlic

Instructions
1. Bring a large pot of lightly salted water to a boil.
2. Cook spaghetti in the boiling water until cooked through but firm to the bite, about 12 minutes; drain and transfer to a large bowl.
3. Drizzle sesame oil over the spaghetti; toss to coat.
4. Place a plate atop the bowl to keep the noodles warm.
5. Heat peanut oil in a wok or large skillet over medium-high heat.
6. Cook and stir garlic and ginger in hot oil until fragrant, about 30 seconds.
7. Add mixed vegetables to the skillet; cook and stir until slightly tender, about 3 minutes.
8. Stir flank steak into the vegetable mixture; cook and stir until the beef is cooked through, about 5 minutes.
9. Mix soy sauce, brown sugar, oyster sauce, and chile paste together in a small bowl; pour over the spaghetti.
10. Dump spaghetti and sauce mixture into the wok with the vegetables and steak; cook and stir until the spaghetti is hot, 2 to 3 minutes.

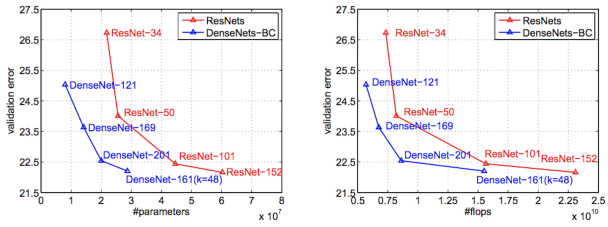
For our evaluation metric, we randomly took 10,000 recipes with more than 1 corresponding image to them.

Model	ResNet-50	ResNet-101	DenseNet-121
Cosine Loss	0.598	0.588	0.719
Average Euclidean Distance	33.52	33.47	17.97

To evaluate our model, we tested the relative accuracy/similarity of the encoding of images to the other images attached to the same recipe. Average cosine similarity and average euclidean distance, where values of 1 and 0 respectively are perfect matches of two images, were two metrics we used to quantify this. By using these metrics to measure the similarity of an image to others associated with the same recipe, we can therefore quantify how accurate our found encodings are for an image and thus measure the accuracy of the resulting recipes our model would output. The values found above indicate that using more advanced versions of classification architectures improves our model's ability by providing it with more accurate encodings. Specifically, DenseNet-121 performed significantly better than ResNet-50 and ResNet-101. Overall, these values give confidence in that the recipes recommended are relevant to a given input image. In addition, when setting $k = 3$, we found that in most cases the first 2 recipes were relatively accurate, but the third recipe was often a little off base. As such, there is room for improvement for higher values of k.

7 Conclusion/Future Work

The limitations of time and computing power are ultimately the main constraints in our project. Without such restrictions, future steps could be taken towards implementing models and algorithms that take advantage of these absent limitations in return for a higher accuracy in image classification and closer recipe recommendations. Our model relied on utilizing pre-trained image classification models such as ResNet-50 and DenseNet 121, to handle the initial classification of images. Illustrated below are graphs that plot the relative validation errors for subsequent versions of the DenseNet and ResNet models. Integrating the models with the lowest validation errors would be the obvious next step to guaranteeing improvements to our existing model. Implementing these versions would expand the range of images our model can accurately classify.



[10]

In addition, we could train on the entire Recipe1M+ dataset instead of the Recipe 1M subset, which would give us a larger variety of recipes to choose from and improve our performance for higher values of k. Finally, we could expand our project such the user could opt to input a recipe and get a corresponding image. This would likely required the specific instructions and ingredients within the recipes to be encoded, and work alongside our image encodings.

8 Contributions

- Topic research and data set selection - Sabrina, Luciano, Dorian
- Research on related works - Sabrina
- Project proposal write-up - Sabrina, Luciano, Dorian
- Transfer learning on ResNet-50/ResNet-101/DenseNet121 (encoding extraction) - Luciano
- Nearest neighbors - Luciano, Dorian
- Recipe extraction and nutritional information labeling - Sabrina
- Project milestone write-up - Sabrina, Luciano, Dorian
- Project poster - Sabrina, Dorian
- Project final write-up - Sabrina, Luciano, Dorian
- Project video - Sabrina, Luciano, Dorian

Github code: https://github.com/slu1212/cs230

9 Works Cited

[1] L. Bossard, M. Guillaumin, and L. Van Gool. Food-101– mining discriminative components with random forests. In European Conference on Computer Vision, pages 446–461. Springer, 2014.
[2] C.-w. N. Jing-jing Chen, "Deep-based ingredient recognition for cooking recipe retrival," ACM Multimedia, 2016.
[3] Salvador, Amaia, Hynes, Nicholas, Aytar, Yusuf, Marin, Javier, Ofli, Ferda, Weber, Ingmar, and Toralba, Antonio. Learning cross-model embbeddings for cooking recipes and food images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
[4] Yang, S.L., Chen, M., Pomerleau, D., Sukthankar, R.: Food recognition using statistics of pairwise local features. In: CVPR (2010)
[5] L. Herranz, W. Min, and S. Jiang, "Food recognition and recipe analysis: integrating visual content, context and external knowledge," CoRR, vol. abs/1801.07239, 2018. [Online]. Available: http://arxiv.org/abs/1801.07239
[6] W. Min, S. Jiang, S. Wang, J. Sang, and S. Mei, "A delicious recipe analysis framework for exploring multi-modal recipes with various attributes," in Proceedings of the 2017 ACM on Multimedia Conference, ser. MM '17. New York, NY, USA: ACM, 2017, pp. 402–410. [Online]. Available: http://doi.acm.org/10.1145/3123266.3123272
[7] J.-j. Chen, C.-W. Ngo, and T.-S. Chua, "Cross-modal recipe retrieval with rich food attributes," in Proceedings of the 2017 ACM on Multimedia Conference, ser. MM '17. New York, NY, USA: ACM, 2017, pp. 1771–1779. [Online]. Available: http://doi.acm.org/10.1145/3123266.3123428
[8] M. Chang, L. V. Guillain, H. Jung, V. M. Hare, J. Kim, and M. Agrawala, "Recipescape: An interactive tool for analyzing cooking instructions at scale," in Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, ser. CHI '18. New York, NY, USA: ACM, 2018, pp. 451:1–451:12. [Online]. Available: http://doi.acm.org/10.1145/3173574.3174025
[9] Manish Chablani, "DenseNet", Towards Data Science. https://towardsdatascience.com/densenet-2810936aeebb
[10] Gao, Hao. "The Efficiency of Densenet." Medium. Last modified August 15, 2017. https://medium.com/@smallfishbigsea/densenet-2b0889854a92.