ECE 324 Project Proposal - REScipe

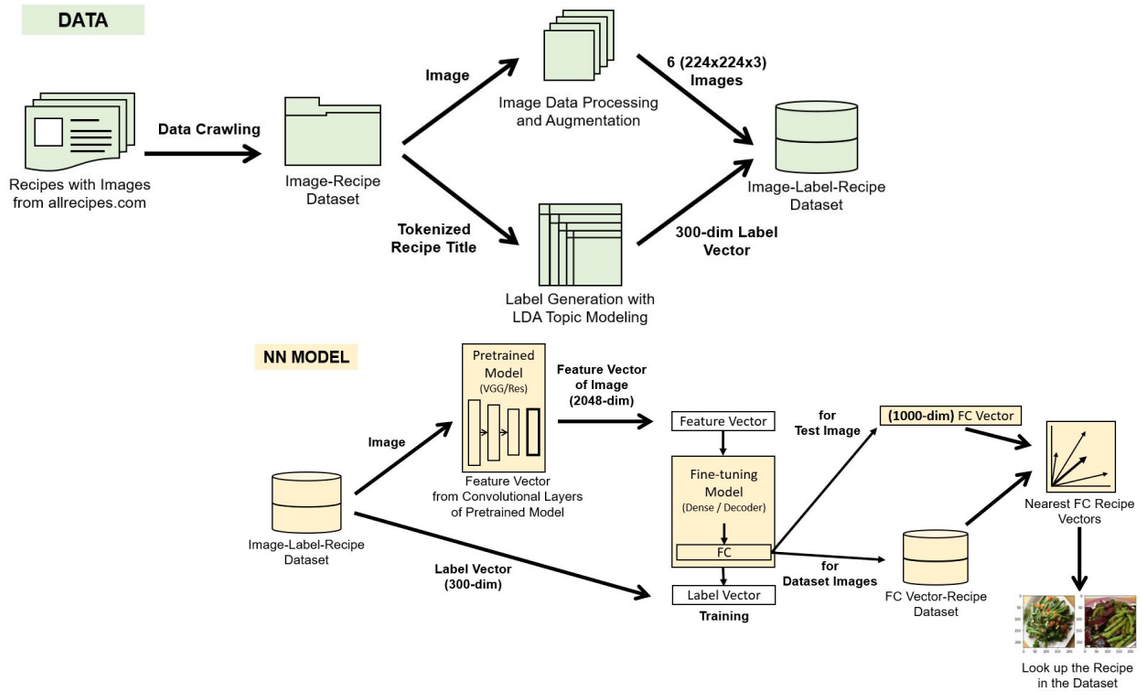
Shawn Zhang (1005154921), Jiakai Chen (1004800539), Steven Zhong (1004946437)

Word Count and Penalty: 1991 words (0% penalty)

1. Introduction

The goal of REScipe is to output a possible recipe given a food image. While browsing the internet or social media, we commonly see photos of appetizing dishes that may not come with a name or a recipe. Our project recommends a recipe to make the dish by finding a recipe image, with features similar to the food image, in a large database. Our project aims to motivate people to gain more experience with home cooking. Machine learning is required as images of dishes are highly variant, and a neural network can effectively extract a dish's features for this task.

2. Project Figure



3. Background & Related Work

One of the related papers would be "Classifying food images represented as Bag of Textons", where the researchers attempt to classify fast food from the Pittsburgh Fast-Food Image Dataset (PFID) into categories such as hamburger, taco, crispy chicken thighs and more using a Bag of Textons approach [1]. The images are stored as a vector of occurrence counts of local image features/textures (hence "bag of textons") and used SVMs to classify the 61 classes with a 67% average classification accuracy.

Another related work would be the not yet released DeepChef, found on Github under Murgio/Food-Recipe-CNN [2]. This work utilizes deep convolutional neural networks with Keras for the classification of 230 recipe categories from a German Recipe database. It uses approximate nearest neighbours to determine similar recipe images and uses a CNN to classify an input image as one of the recipe categories.

4. Data Source, Processing and Labeling

Data collection was done by web scraping allrecipes.com. We used a web scraping library for cooking websites to create our raw dataset, which consists of the recipe title, image source, ingredients, recipe, and others, by iterating through a large range of recipe URLs. We removed ones with the default "no image" image and collected 46904 recipes.

	A	B	C	D	E	F	G	H
1	Index	Title	Image Source	Total Time	Serving Size	Ingredients	Recipe	URL Ext.
2	0	Whole Wheat Pita Bre	https://imagesvc.me	150	12 servings	['1 cup warm wate	In a bowl mix th	70006
3	1	Baked Mac and Chees	https://imagesvc.me	30	1 serving	['3 tablespoons ur	Preheat an ove	70012
4	2	Blonde Bobbie	https://imagesvc.me	2	1 serving	['Â½ cup Ice cubes	Place ice in a sh	70014
5	3	Egg Butter	https://imagesvc.me	45	1 cup	['4 eggs', 'Â½ cup l	Place the eggs i	70019
6	4	Beer Pizza	https://imagesvc.me	55	2 pizzas	['1 tablespoon oli	Preheat oven to	70032
7	5	Microwave Lemon Cu	https://imagesvc.me	16	2 cups	['1 cup white suga	In a microwave	70036
8	6	Apple Chutney	https://imagesvc.me	45	5 cups	['15 tart apples - p	In a saucepan,	70037
9	7	Frank's Spicy Alabama	https://imagesvc.me	140	8 servings	['2 pounds ground	Crumble the gro	70038
10	8	Simple and Tasty Chin	https://imagesvc.me	115	6 buns	['1 tablespoon act	Sprinkle the yea	70048
11	9	Italian Sausage Deligh	https://imagesvc.me	60	6 servings	['6 (3.5 ounce) link	Bring a large po	70051
12	10	Cousin David's Slow C	https://imagesvc.me	605	4 servings	['1 (3 pound) beef	Place the beef l	70096

“Raw Data sourced from allrecipes.com”

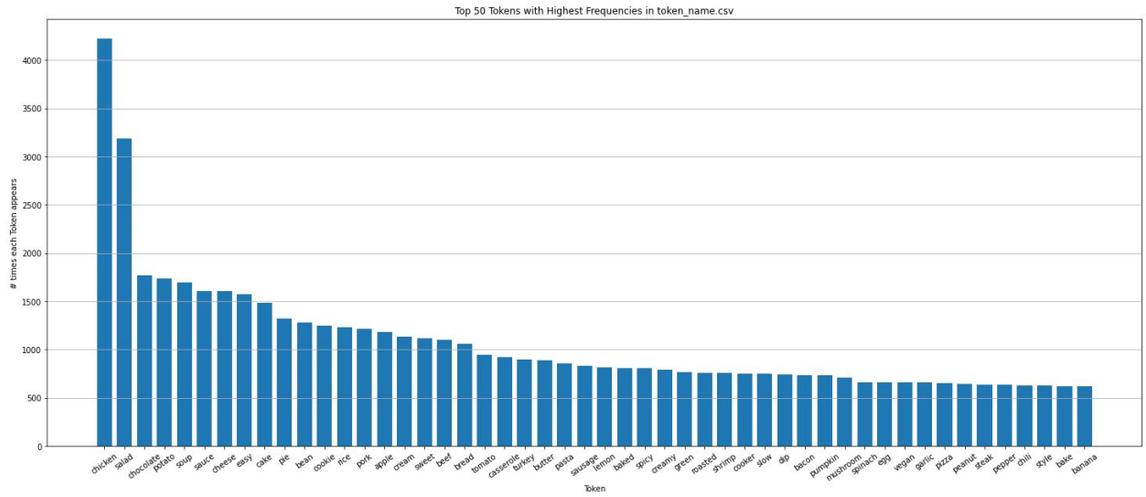
We processed the images to reduce memory usage and to increase the amount of training data. We resized all images to the size of 224x244x3(RGB) and generated 5 augmented images with cropping and colour jitter for each.

“Resized Original Images”

We processed the recipe titles to generate lower-dimensional labels for training. We used the “en_core_web_sm” spaCy model to tokenize and lemmatize the titles and to remove stop words. To compensate for spaCy’s poor plural-to-singular conversion, we used the “inflect” library. We cleaned the dataset by removing titles containing no common token or consisting of a single uncommon word. The filtered dataset contains 46659 recipes with a generally balanced distribution of tokens.

name	token_name
Whole Wheat Pita Bread	[wheat, pita, bread]
Baked Mac and Cheese for One	[baked, mac, cheese]
Blonde Bobbie	[blonde, bobbie]
Egg Butter	[egg, butter]
Beer Pizza	[beer, pizza]
...	...

“Titles to Tokens”



“50 Most Common Tokens”

We used the filtered dataset to train a Latent Dirichlet Allocation (LDA) model for label generation. First, we used Gensim to convert titles to “TF-IDF” representations. Then, we set up the training environment for the LDA model and tuned its hyperparameters by inspecting topics and evaluating term-related recipes of common tokens.

```
Sample Topics of gensim LDA model using TF-IDF:
(0, '0.617*"smoke" + 0.210*"high" + 0.043*"fiber"')
(1, '0.954*"salmon" + 0.000*"missouri" + 0.000*"manjar"')
(2, '0.324*"marinade" + 0.213*"barbeque" + 0.152*"african"')
(3, '0.291*"thigh" + 0.251*"fennel" + 0.222*"dark"')
(4, '0.490*"cheesecake" + 0.232*"alfredo" + 0.139*"blue"')
(5, '0.540*"egg" + 0.321*"pesto" + 0.077*"steamed"')
```

```
>>>>> 10  top recipes related to  salad <<<<<<
['gelatin', 'salad']     0.5646806673752455
['orange', 'carrot', 'gelatin', 'salad']         0.4907170399521171
['patriotic', 'gelatin', 'salad']        0.46548112207803805
['pastel', 'gelatin', 'salad']   0.4581125640275216
['whip', 'carrot', 'salad']      0.4556416336893485
['fizzy', 'gelatin', 'salad']    0.4556160624645823
```

“Topics & Term-related Recipes”

After finishing training, we used the LDA model to produce a 300-element label for each recipe title and filtered out ones with poor probability distributions after normalization. At last, the dataset has 45151 labelled recipes.

recipe_name	...	label
[wheat, pita, bread]	...	[0.001109877913429523, 0.001109877913429523, 0...
[baked, mac, cheese]	...	[0.0016639162998775608, 0.0016639162998775608,...
[blonde, bobbie]	...	[0.0016638935108153079, 0.0016638935108153079,...
[egg, butter]	...	[0.0033222591362126247, 0.0033222591362126247,...
[beer, pizza]	...	[0.0033222591362126247, 0.0033222591362126247,...
...	...	...

"Tokenized Titles with Labels"

Finally, the training set consists of 200000 images(original+4 augmented). The validation set consists of 40000 images(augmented). The testing set consists of 6600 completely new images. The train/val/test split is roughly 81/16/3. We choose to split the dataset by samples rather than by percentage so as to save our data files more comprehensively in integer batches.

5. Architecture

ResNet-34 with Fully-connected Layers (demonstrates overfitting)

Pre-trained Model	ResNet-34 *without fc1 and pool_time*(frozen)
Fine-tuned Model	BatchNorm1d(4608), Linear(4608,1024), BatchNorm1d(1024), LeakyReLU(), Dropout(0.15), Linear(1024,1000), BatchNorm1d(1000), LeakyReLU(), Dropout(0.1), Linear(1000,300), LeakyReLU()
Range of Learning Rate	[0.001]
Range of Batch Size	[512]
Epochs	120

ResNet-50 with Fully-connected Layers

Pre-trained Model	ResNet-50 *without fc1*(frozen)
Fine-tuned Model	BatchNorm1d(2048), Linear(2048,1024), BatchNorm1d(1024), LeakyReLU(), Linear(1024,1000), BatchNorm1d(1000), LeakyReLU(), Linear(1000,300), LeakyReLU()
Range of Learning Rate	[0.001(when overfitting), 0.0003]
Range of Batch Size	[64(when overfitting), 128, 256]
Epochs	120

ResNet-50 with Decoding Fully-connected Layers

Pre-trained Model	ResNet-50 *without fc1*(frozen)
Fine-tuned Model	BatchNorm1d(2048), Linear(2048,4096), BatchNorm1d(4096), LeakyReLU(), Dropout(0.15), Linear(4096,4096), BatchNorm1d(4096), LeakyReLU(), Dropout(0.15), Linear(4096,1000), BatchNorm1d(1000), LeakyReLU(), Dropout(0.1), Linear(1000,300), LeakyReLU()
Range of Learning Rate	[0.001(when overfitting), 0.0003]
Range of Batch Size	[64(when overfitting), 128, 256]
Epochs	120

(All use MSELoss() and Adam optimizer)

To save memory and speed up training, we preprocess images into feature vectors by storing the outputs of convolutional layers. During our training, feature vectors are input into fully-connected layers. Later, we will know this is problematic with ResNet models.

6. Baseline Model

Pre-trained Model	VGG-16 *without fc*(frozen)
Fine-tuned Model	BatchNorm1d(4096), Linear(4096,1024), BatchNorm1d(1024), LeakyReLU(), Dropout(0.15),

	Linear(1024,1000), BatchNorm1d(1000), LeakyReLU(), Dropout(0.1), Linear(1000,300), LeakyReLU()
Range of Learning Rate	[0.001]
Range of Batch Size	[128]
Epochs	120

VGG-16 is a classical CNN for image classification. It is shallower than ResNets but has more parameters. We used the hyperparameters from the fined-tuned ResNet-34 to allow a comparison of feasibility. A difference between the original two models was that VGG had 2 fully-connected layers, while ResNets had only 1.

7. Quantitative Results

LDA Model

The LDA model was evaluated based on the number of unique topics and the accuracy of sample term topics. They helped determine the label size, the training chunk size, and the number of passes. The best LDA model has 298 unique topics with a term topic accuracy of 86%(129/150 terms).

```
(295, '0.586*"hazelnut" + 0.178*"jambalaya" + 0.081*"mole"')
(296, '0.426*"milk" + 0.176*"custard" + 0.119*"fast"')
(297, '0.921*"ham" + 0.014*"coat" + 0.000*"manjar"')
(298, '0.385*"dressing" + 0.381*"mango" + 0.121*"salad"')
(299, '0.317*"mama" + 0.274*"sourdough" + 0.070*"coriander"')

# repeated topic: 2
```

"Number of Repeated Topics"

update 512	Chunk#	16	32	64	128	256	512
Topic#	/150	32	16	8	4	2	1
100			34 (147s)				56 (41s)
150							
200					84 (101s)	76 (82s)	55 (75s)
250			84 (249s)	90 (158s)			
300		66 (462s)	***129 (309s)***	***118 (189s)***	(140s)		
350			92 (309s)	***124 (223s)***			
400			***110 (341s)***	104(229s)			105 (139s)

```
>>>>> 10  top recipes related to  chocolate <<<<<<
['chocolate', 'cheesecake']      0.5837430327611446
['blue', 'cheese', 'chicken']    0.49031658048389554
['chocolate', 'cheesecake']      0.4884687351574934
['chocolate', 'peppermint', 'cheesecake']        0.48483111346700114
['savory', 'blue', 'cheese', 'cheesecake']       0.4760080560155332
['philadelphia', 'chocolate', 'cheesecake']      0.4704744078163579
['chocolate', 'turtle', 'cheesecake']    0.44817913693912764
['chocolate', 'blis', 'cheesecake']      0.4481791369382284
['chocolate', 'cheesecake', 'blondie']   0.4454370939504848
['blue', 'cheese', 'fettucine']          0.4412993696294484
```

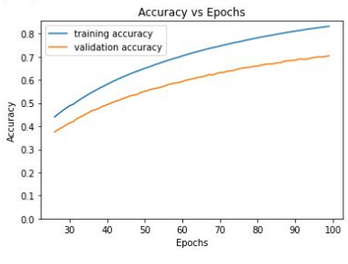
"Best Accuracy from Tuning & Example of a Term Topic"

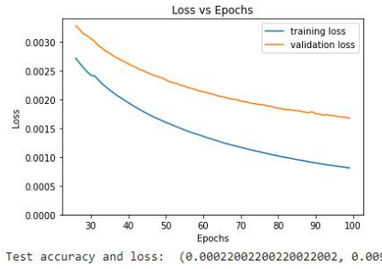
NN Model

The decision function for the NN model was the cosine similarity with a threshold of 0.9. Since labels are 300-dimensional and normalized, cosine similarity is more demonstrative of performance than the euclidean distance with a more arbitrary threshold.

The baseline VGG achieved 83% training accuracy, 70% validation accuracy, and 1% testing accuracy.

```
[100] train loss: 0.00081 train accu: 0.832235 valid loss: 0.001678 valid accu: 0.704168
```

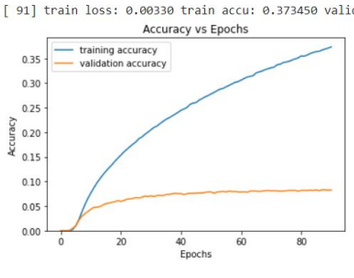


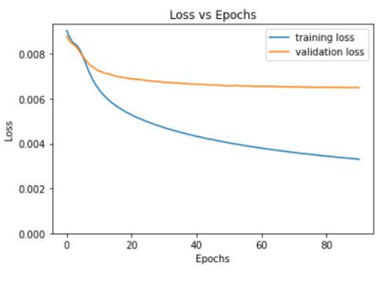



```
Test accuracy and loss:  (0.0002200220022002, 0.009595020054106992)
```

The ResNet-34 with FC layers achieved 37% on training, 8% on validation, and 0% on testing.

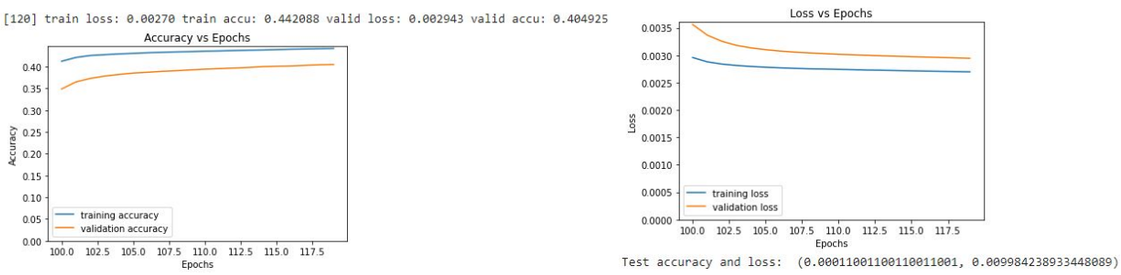
```
[ 91] train loss: 0.00330 train accu: 0.373450 valid loss: 0.006502 valid accu: 0.082675
```





The results from ResNet-34 was concerning. The model suffered from both underfitting and overfitting. To improve the performance, we switched to ResNet-50 with more complicated FC designs and train with different batch sizes. We regularly stop the training to tune the hyperparameters.

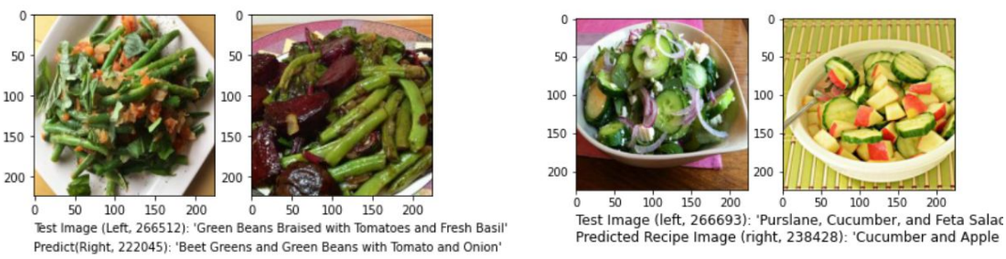
The ResNet-50 with the same design achieved 44% on training, 40% on validation, and 1% on testing.



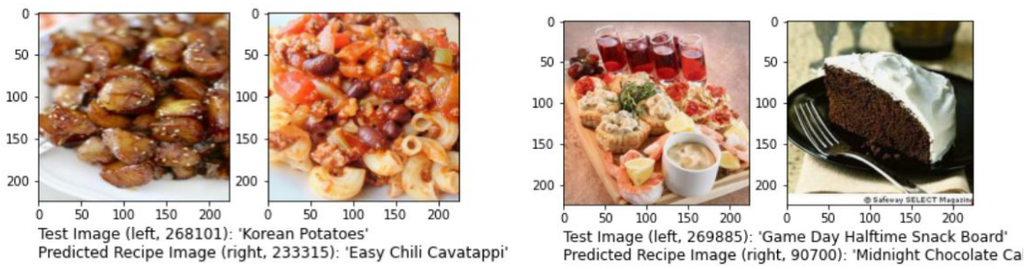
The ResNet-50 with decoding design achieved 79% on training, 67% on validation, and 1% on testing.

(the chosen model is frozen before 130 epochs; the right figure is for testing accuracy only)

8. Qualitative Results



“Good Results - FC Vector of the Image and FC Vector of Dataset Recipe are similar.”



“Poor Results - Only images are similar; or they are completely different.”

Looking at the top two images, our model performs well when our input has similar features to those mapped in our database. It generally does well on salads, soups, burgers, and vegetable dishes. However, if the input features are underrepresented in our model (game board), it maps to a non-similar recipe or midnight chocolate cake. This chocolate cake happens to account for many of the outputs. We speculate that during LDA label creation and training, the chocolate cake feature vector is far away (euclidean distance) from the rest of the recipes and would monopolize the feature space that contains the output of unfamiliar inputs. This caused our test set performance to be very poor as the test set is from a completed new distribution.

```
                                                  label    code
0     [0.0016638935108153076, 0.0016638935108153076,...  266512
1     [0.0016638935108153076, 0.0016638935108153076,...  266512
2     [0.0016638935108153076, 0.0016638935108153076,...  266512
3     [0.0016638935108153076, 0.0016638935108153076,...  266512
4     [0.0016638935108153076, 0.0016638935108153076,...  266512
...                                                 ...     ...
9085  [0.0033222591362126242, 0.0033222591362126242,...  269899
9086  [0.0033222591362126242, 0.0033222591362126242,...  269899
9087  [0.0033222591362126242, 0.0033222591362126242,...  269899
9088  [0.0033222591362126242, 0.0033222591362126242,...  269899
9089  [0.0033222591362126242, 0.0033222591362126242,...  269899

[9090 rows x 3 columns]
```

"Test Set with a Distribution of (New Image+5 Augmented)"

9. Discussion and Learnings

Our model performs well on the validation set that contains augmented images, meaning our model does generalize to some degree. We believe our model performed somewhat poorly on the test set, but foreseeable since it was out-of-data distribution. Our model generally performs well when our training titles accurately represent the recipes with few/no extraneous words, when we have many of the same types of food (e.g. burgers, salads), and their features are distinct from other food groups.
Besides the tuning of the model and architecture, its performance was also limited by a considerable number of poorly named recipes and unclear images, as well as an unbalanced recipe representation.

What was surprising was that many of our outputs were the recipe "Midnight Chocolate Cake". As mentioned in the qualitative results section, this could be due to its feature vector being isolated from the other clusters of feature vectors (clusters of burgers, salads), thus mapping all the images that aren't extremely similar to these clusters to the cake instead. We were also surprised at how "creative" and unrelated some of the titles were to the recipe, which made our task difficult even after filtering out words with few occurrences.

```
268550 ['blackberry', 'slushie'] probability of topics are too similar.
269264 ['dulce', 'leche', 'brownie'] probability of topics are too similar.
269877 ['attack', 'shooter'] probability of topics are too similar.
269894 ['love', 'potion'] probability of topics are too similar.
269896 ['puck', 'cheeseburger', 'slider'] probability of topics are too similar.
79797 ['stroopwafel'] probability of topics are too similar.
81149 ['suspiro'] probability of topics are too similar.
94576 ['tiropita'] probability of topics are too similar.
241660 ['salpicon'] probability of topics are too similar.
246438 ['bordeaux'] probability of topics are too similar.
Removed 574 recipes with insignificant topic probabilities.
```

"Difficulty in LDA Modeling: Unrelated Naming"

If we were to do this again, we may use ingredients instead of titles since they are more consistent across recipes. One reason that we went with titles was that we wanted our model to recognize larger food types, such as burgers and cakes. An additional option would be to use glove vectors but would come with lots of data cleaning such as minimizing the impact of spices and taking ingredient quantity into consideration.

One thing we would be sure to try is unfreezing some of the ResNet layers and retraining with our data. The outputs from the current pre-trained ResNet were too clustered to be used for further MLP training as the outputs for the images all had cosine similarities above 0.99. This means that Resnet inherently picked up the "food" class in its pre-trained residual blocks.

```
Index(['resnet_features', 'label'], dtype='object')

cos_similarity  e_distance      index (cmp with next)
[[0.99545217]]  2.0607522       0
[[0.99532866]]  2.094894        1
[[0.9956309]]   2.028304        2
[[0.9959206]]   1.9654641       3
[[0.99534404]]  2.1097574       4
[[0.99569434]]  2.0096145       5
[[0.99562407]]  2.021868        6
[[0.99579453]]  1.9653133       7
[[0.9956789]]   1.9935576       8
[[0.995701]]    2.004184        9
```

"Highly Similar Feature Vectors from ResNet Convolutional Layers"

Lastly, we could always simplify the problem to a strict classification between set categories such as cakes and pies, but the use case for such a model is much more limited.

10. Ethical Framework

We identify two main stakeholders of our project as the recipe uploader and user because they are linked to the input and the output of our project, respectively. Other stakeholders in this project include the recipe creator (which may not be the recipe uploader), and us, the project creators.

A recipe uploader is an online blogger and is the source of our recipes. What they post on the website directly influences the quality of our data pool. When they post great descriptions of their recipes and upload clear images, our model is benefited by having more well labelled, unclustered data. By gathering uploaders' data, the ethical question of consent on data collection arises. The use of our model that recommends recipes will decrease the autonomy of recipe uploaders/creators, since their data will be shared on another platform without their direct permission. However, the collection of such data can increase beneficence to the users as it allows a diverse recipe pool that we can source from.

A user will be inputting an image into our model and obtaining a recipe output. The output of our model has potential influences on their diets since they are likely to try out the recommended recipe. This creates a huge non-maleficence ethical issue from two aspects:
The use of our project may decrease non-maleficence to users by

- Recommending unhealthy food/version instead of healthy (bad for lifestyle)
- Recommending recipe that can cause an allergic reaction (may cause death)

As the creators, we understand that if our project were to be deployed in the real world, we would have to take these issues into serious consideration. Such an example would be asking for the users' allergies or food preferences (low-sodium, vegan diet) and filtering out the recommendations accordingly.

References

[1] G. M. Farinella, M. Moltisanti, S. Battiato. Classifying food images represented as Bag of Textons. *2014 IEEE International Conference on Image Processing*. https://ieeexplore-ieee-org.myaccess.library.utoronto.ca/document/7026055

[2] M. Serifovic. Murgio/Food-Recipe-CNN. *DeepChef*. https://github.com/Murgio/Food-Recipe-CNN

Permissions:

Jacky Chen

- permission to post video: yes
- permission to post final report: yes
- permission to post source code: yes

Shawn Zhang

- permission to post video: yes
- permission to post final report: yes
- permission to post source code: yes

Steven Zhong

- permission to post video: yes
- permission to post final report: yes
- permission to post source code: yes