Nutrient Analysis: How to Tell if Your Food is Healthy by Looking at It?

Roshan Ram (Mentor)
Carnegie Mellon University
Pittsburgh, PA 15213
rram@andrew.cmu.edu

Sijie Luo
Carnegie Mellon University Silicon Valley
Mountain View, CA 94035
sijieluo@andrew.cmu.edu

Yiren Zhou
Carnegie Mellon University Silicon Valley
Mountain View, CA 94035
yirenzho@andrew.cmu.edu

Yusheng Hu
Carnegie Mellon University Silicon Valley
Mountain View, CA 94035
yushengh@andrew.cmu.edu

Zihan Wang
Carnegie Mellon University Silicon Valley
Mountain View, CA 94035
zihanwa2@andrew.cmu.edu

Abstract

Food and its accompanying nutrients play critical roles in our health. Understanding food nutrients have been a difficult task in computer vision, especially with multi-stage solutions where the quality of individual stages is highly decoupled from the rest [15]. Our project aims to create an end-to-end solution using neural networks to perform accurate nutrient estimation for food with pictures of the food. Our general approach is to implement a multitask-learning neural network and train it using the Nutrition5k dataset [14], which contains depth images and highly accurate nutritional content annotations. We then attempt to incorporate other architectures as the backbone for feature extraction, along with data augmentation, to see if it outperforms the previous. We hope that the underlying research can spawn smarter applications in the field, such as more intelligent food recommendation systems deployed on more universally distributed smart devices such as smartphones.

1 Introduction

Human health depends strongly on healthy diets, which concern the type, compositions, nutrients and processing of the food. Hence, people seek to determine food attributes in a rapid and accurate manner in daily life. Consumer-grade applications such as MyFitnessPal have been adequate in that it allows users to easily log the macro-nutrients of their meals if they are familiar with the ingredients. However, the manual process still creates a barrier to entry for users who are not keen on culinary science. We attempt to explore an automatic and accurate method to estimate food nutrients with only images of the food, thus hoping to extend the possibility of spawning smarter applications for the consumers.

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

2 Related Work

Current research about calorie estimation and nutrient analysis is mostly involved with ingredient classification and volume estimation, where a single dish is classified into a set of known ingredients or items and the volume of each item is also being estimated.

Once food volume and food category have been retrieved, nutritional density datasets can be used to calculate total mass, calorie, and macro-nutrients. Traditionally, vision-based nutrient analysis and calorie estimation are done in a multi-stage fashion as it consists of multiple tasks including food image analysis, volume estimation, and nutrient derivation. There have been existing work for these tasks. For example, NutriNet [8] was a deep convolution neural network for food and drink recognition from images; Research based on semantic segmentation [11] has also been explored to perform segmentation of food images [11], which can then be used to estimate the calories of the food. However, multi-stage methods have a major limitation: each stage needs to be defined and optimized individually, thus error accumulation from stages can easily happen [15].

In contrast, end-to-end architectures can reduce error propagation while maintaining the relevancy of information internally without separating it into multiple stages. Compared to multi-stage architecture with a pipeline of multiple stages, the end-to-end architecture focuses on using a single neural network model to extract features from the input. Only the original inputs and the final outputs are required to be specified, while the information learned by the neural network is internally relevant [15]. Ruede et al.[10] adopted pretrained ResNet and DenseNet architectures backbone models to perform calorie regression.

Our project attempts to hopefully move the end-to-end approach forward by training several neural networks using a comprehensive and generic dataset, Nutrition5k [14]. Moreover, we aim to improve upon the baseline architecture proposed in the original paper associated with the dataset.

3 Contributions

3.1 Baseline Model

The baseline model described in [14] consists of two parts: 1) a backbone as the image feature extractor and 2) a multi-task [9] learning network. The backbone is based on an InceptionV2 [13] model pretrained using JFT300M [12], which receives images of resolution 256x256 as input. Before feeding into the network, the images were downsized and center cropped in an effort to keep the most relevant dish area. The backbone network works as an extractor to obtain the most salient features in the dish image, upon which the regression is performed afterwards to predict the nutrition values. The baseline architecture models the regression of three nutritional categories as a multi-task learning framework, where each regression task is trained with a separate multi-task head using the features extracted by the common backbone.

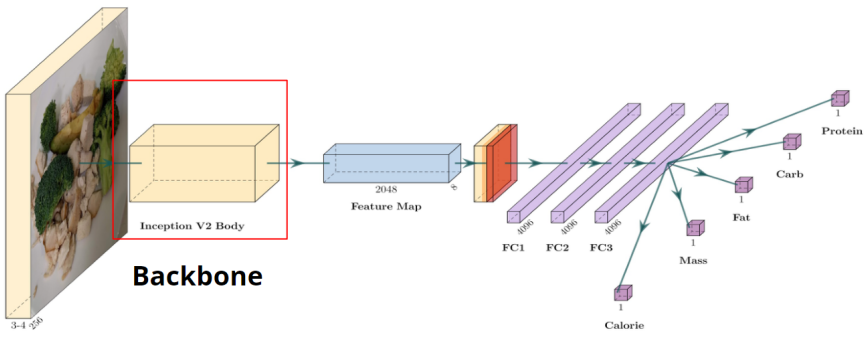


Figure 1: Baseline Model Architecture[14]

3.2 Method

We assumed that the backbone model does the most heavy-lifting work for predicting the macronutrient value. Therefore, we proposed to explore different backbone models to see if they could improve the performance of an end-to-end structure. For this, we replace the InceptionV2 backbone model of the baseline model and adopt multiple variant of the prominent Resnet[3], MobileNetV3[4], ViT[1], and ConvNeXT[7]. All of the models we selected are pre-trained with 1000 output classes. For fitting the end-to-end architecture, the backbone model is connected with one 1000×4096 fully connected layer and following two 4096×4096 fully connected layers. We used a compound mean-absolute-error (MAE) as the loss function (detailed in 4.3). We speculated that the optimizer and learning rate might affect the performance of InceptionV2 and our adopted models. Apart from RMSprop used in the baseline model, we also applied SGD and Adam with various learning rates and weight decay. In addition to the Nutrition5k dataset, we adopt various data augmentation techniques to examine the effect of augmented data on the prediction results (detailed in 4.2). Please see our GitHub repository for the implementation details: `https://github.com/yirzhou/11785-Nutrient-Estimation`.

4 Experiments

4.1 Datasets

In our project, we will use Nutrition5k dataset [14] to train our model, which is a dataset of visual and nutritional data for 5k realistic plates of food captured from Google cafeterias using a custom scanning rig. It includes 5,006 plates of food, each containing overhead RGB-D images, a fine-grained list of ingredients, per-ingredient mass, total dish mass and calories, fat, protein, and carbohydrate macro-nutrient masses.

The Nutrition5k dataset $D = \{I_i, Y_i\}_{i=1}^N$ consists of N instances of image I_i and supervision label Y_i pairs. Each supervision label $Y_i = (y_i^w, Y_i^m, y_i^{cal})$ includes three types: total weight label y_i^w, macronutrient labels Y_i^m, and a calorie label y_i^{cal}. All of the supervision labels are calculated using a function of the weight (in grams) of each ingredient K_l where l is the index going over all ingredients of dish in the image I_i as follows:

- y_i^w is the total weight of each dish
- Y_i^m is a vector of weights of each macronutrient. Let $M = \{carb, fat, protein\}$, then $\forall j \in M$, each entry in Y_i^m corresponds to $y_{ij}^m = F_{macro}(K_l, j)$ where F_{macro} is a function calculating the amount of each macronutrient.
- $y_i^{cal} = F_{cal}(K_l)$ where F_{cal} is a function calculating the total number of calories from per ingredient weight.

4.2 Data Augmentation

In order to improve the model performance, we explored several ways to achieve data augmentation. We applied the following transformations to the input image: *RandomRotate(10)*, *RandomHorizontalFlip* and *RandomPerspective*.

4.3 Metrics

Our goal is to predict the precise nutritional content of a dish, including the number of calories it contains, the total mass, and the respective mass for each macronutrient (fat, carbohydrates and protein), from a single RGB image I_i available in the dataset. We train a regression model to predict the values, and we measure the regression accuracy using the mean absolute error (MAE) defined as:

$$MAE = \frac{1}{N} \sum_{i=1}^{N} |\hat{y}_i - y_i|$$

$$l_{multi} = l_{calorie} + l_{mass} + l_{fat} + l_{carb} + l_{protein}$$

where $\hat{y}_i$ is the predicted value for a given test image I_i for each nutritional content metric, and y_i is the corresponding ground-truth value. For the metric units, macronutrients and mass are measured in

grams, while calories values use kilocalories units. Same as [14], apart from MAE values, we use the percentage of MAE with respect to the ground truth value as an indicator of how the prediction deviates from the ground truth. Specifically, we are optimizing the overall loss l_{multi} defined as:

$$
\begin{aligned}
l_{multi}(D \mid W) = & \frac{1}{N} \sum_{i=1}^{N} [l_m \left(I_i, Y_i^m \mid W\right) \\
& + l_c \left(I_i, y_i^{cal} \mid W\right) \\
& + l_w \left(I_i, y_i^w \mid W\right)] \\
l_m \left(I, Y^m \mid W\right) = & \frac{1}{|M|} \sum_{j \in M} \left|\hat{y}_j^m - y_j^m\right| \\
l_c \left(I, y^{cal} \mid W\right) = & \left|\hat{y}^{cal} - y^{cal}\right| \\
l_w \left(I, y^w \mid W\right) = & \left|\hat{y}^w - y^w\right|
\end{aligned}
$$

where l_{multi} is weighted over all three nutrition components loss: calories MAE loss l_c, total mass MAE loss l_m and macronutrients MAE loss l_m.

4.4 Backbones

4.4.1 Resnets

Residual neural networks (ResNet)[3] utilize *skip connections*, or *shortcuts* to jump over some layers, which can avoid the problem of vanishing gradients and mitigate the degradation problem. ResNet is easier to optimize and can gain accuracy from considerably increased depth.

In this project, we implemented pretrained *ResNet-50*, *ResNet-101* and *ResNet-152* respectively as our network's backbone, the architectures are shown in Figure 1.

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Figure 2: ResNet Architectures[3]

4.4.2 MobileNetV3

MobileNetV3[4] is a generation of MobileNets which is tuned to mobile phone CPUs through a combination of hardware-aware network architecture search (NAS) complemented by the NetAdapt algorithm and then subsequently improved through novel architecture advances.

MobileNetV3 includes the use of the mobile-friendly squeeze-and-excitation block, which replaces the classical sigmoid function with a piecewise linear approximation. Combining h-swish plus mobile-friendly squeeze-and-excitation with a modified version of the inverted bottleneck structure introduced in MobileNetV2 yielded a new building block for MobileNetV3.

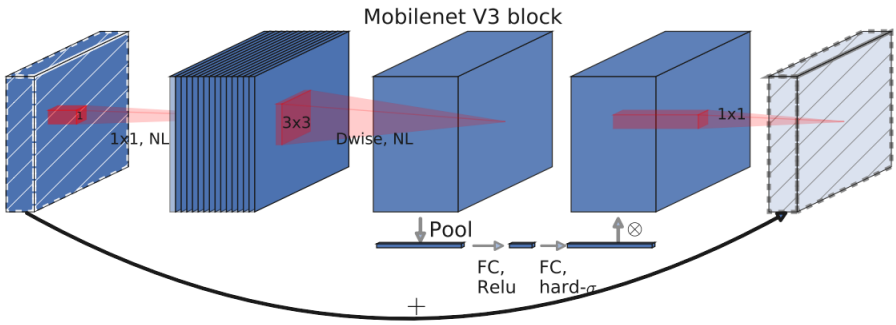


Figure 3: MobileNetV3 block[4]

4.4.3 ViT

Inspired by the Transformer scaling successes in NLP, Vision Transformer (ViT)[1] which applies a standard Transformer directly to images by splitting an image into patches and providing the sequence of linear embeddings of these patches as an input to a Transformer.

Compared with CNNs architectures, when pre-trained on large amounts of data and transferred to multiple mid-sized or small image recognition benchmarks, ViT achieves better results while requiring substantially fewer computational resources to train.

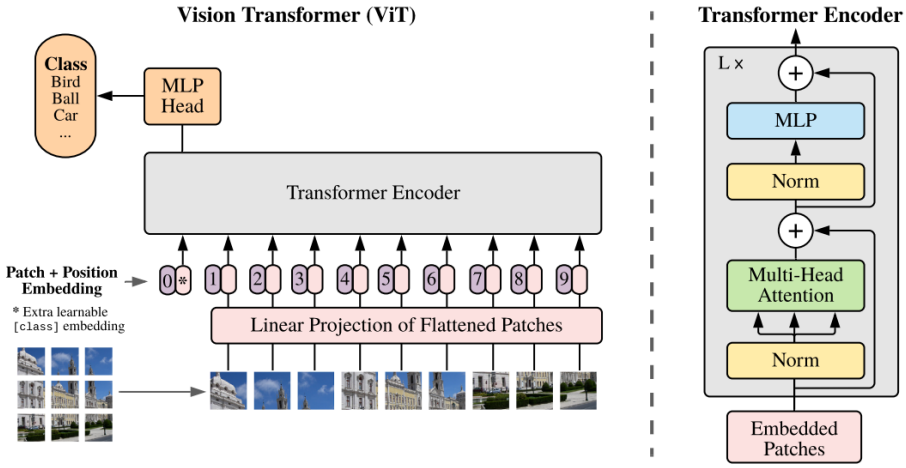
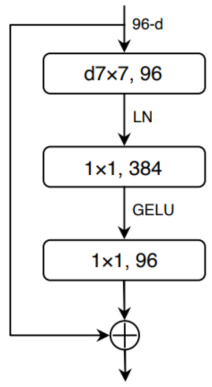


Figure 4: Vision Transformer Model[1]

4.4.4 ConvNeXt

Although ViT completely changed the fundamental approach to network architecture design in computer vision, a vanilla ViT faces difficulties when applied to general computer vision tasks such as object detection and semantic segmentation. ConvNext[7] explores the design spaces and tests the limits of what a pure ConvNet can achieve, and competes favourably with Transformers in terms of accuracy and scalability while maintaining the simplicity and efficiency of standard ConvNets.



	output size	• ConvNeXt-T
stem	56×56	4×4, 96, stride 4
res2	56×56	[d7×7, 96; 1×1, 384; 1×1, 96] × 3
res3	28×28	[d7×7, 192; 1×1, 768; 1×1, 192] × 3
res4	14×14	[d7×7, 384; 1×1, 1536; 1×1, 384] × 9
res5	7×7	[d7×7, 768; 1×1, 3072; 1×1, 768] × 3
FLOPs		4.5×10^9
# params.		28.6×10^6

Figure 5: ConvNext Block and Architecture[7]

5 Results

5.1 Baseline Implementation

We implemented the InceptionV2 backbone, two 4096-dimensional fully connected (FC) layers followed by a final third and fourth FC layers for multi-task learning, as well as the loss functions as stated in [14]. Then we evaluate the model outputs by calculating absolute and percentage mean average error from a CSV of per-dish nutrition values, which can generate results that can be directly compared to those reported in [14].

The outcomes vary. We first trained the network with hyperparameters suggested by [14], however, the result was not very close to the one reported in the paper. Then we tried multiple combinations of hyperparameters to match the baseline paper result, the results are listed in Table 1.

For all combinations we tried, we found that the model can get a lower Calorie MAE compared with which reported in the paper. As for Fat, we could get similar results as the paper. However, our results of MAE for Mass(except Adam), Carb and Protein the were higher than which illustrated in the paper.

	Optimizer	LR	Calorie	Mass	Fat	Carb	Protein
Baseline[14]	RMSprop	1e-4	70.6 / 26.1%	40.4 / 18.8%	5.0 / 34.2%	6.1 / 31.9%	5.5 / 29.5%
Attempt 1	RMSprop	1e-4	54.8 / 23.0%	66.8 / 31.6%	5.93 / 49.8%	10.56 / 58.2%	8.25 / 49.0%
Attempt 2	RMSprop	2e-4	68.4 / 28.7%	58.1 / 27.5%	6.40 / 53.7%	10.16 / 56.1%	8.63 / 51.2%
Attempt 3	SGD	1e-4	55.1 / 23.1%	64.3 / 30.4%	5.93 / 49.8%	10.37 / 57.2%	8.18 / 48.6%
Attempt 4	SGD	2e-4	92.4 / 38.7%	69.8 / 33.4%	6.83 / 57.4%	10.68 / 58.9%	9.18 / 54.5%
Attempt 5	Adam	2e-4	51.4 / 21.5%	29.1 / 13.8%	4.96 / 41.6%	9.86 / 54.4%	8.30 / 49.3%

Table 1: Baseline Implementation results

5.2 Different backbones

Table 2 shows our experiment results of switching different backbones for image feature extraction. All the models listed in this table are trained without any augmentation steps.

Backbone	Calorie	Mass	Fat	Carb	Protein
InceptionV2	54.8 / 23.0%	66.8 / 31.6%	5.93 / 49.8%	10.56 / 58.2%	8.25 / 49.0%
ResNet50	41.0 / 17.2%	20.9 / 9.9%	4.81 / 40.4%	10.08 / 55.6%	8.19 / 48.6%
ResNet101	39.8 / 16.7%	20.2 / 9.6%	4.99 / 42.0%	**9.81 / 54.1%**	8.14 / 48.3%
ResNet152	**35.0 / 14.7%**	**19.0 / 9.0%**	**4.64 / 39.1%**	9.83 / 54.2%	8.05 / 47.8%
MobileNetV3	46.0 / 19.3%	33.3 / 15.8%	4.98 / 41.8%	9.93 / 54.8%	**7.93 / 47.1%**
ViT	110.3 / 46.3%	70.8 / 33.5%	7.63 / 64.2%	10.90 / 60.1%	10.65 / 63.2%
ConvNeXt-T	62.3 / 26.2%	65.0 / 30.7%	6.30 / 52.9%	10.54 / 58.1%	8.55 / 50.8%
ConvNeXt-S	47.1 / 19.8%	75.5 / 35.7%	5.75 / 48.3%	10.77 / 59.4%	8.26 / 49.1%
ConvNeXt-base	82.8 / 34.7%	58.8 / 27.8%	6.85 / 57.5%	10.48 / 57.8%	8.95 / 53.2%

Table 2: Results for different backbones

It is observed that changing the backbone model from InceptionV2 to ResNet variants and MobileNet variants leads to significant performance gain in all five types of nutrients. All ResNet variants decreased the error by a large margin, especially for mass nutrients. Among these three variants, ResNet 152 had the best performance, with the drop of MAE percentage as 6.3%, 22.6%, 10.7%, 4.0%, 1.2% for calorie, mass, fat, carb and protein nutrients respectively. On the other hand, ViT and ConvNeXt variants lead to a decrease in performance. For ViT backbone, it increased the MAE to almost double the baseline model in calorie nutrients. For ConvNeXt variants, although each model decreased the MAE in one or two types of nutrient, generally they did not perform as well as the baseline model.

5.3 Ablation study

We also performed an ablation study on the augmentation step to evaluate how image augmentation techniques would affect our models' performance. We chose four models that had performance gain mentioned in the section above, and Table 3 shows the metric difference between without/with augmentation steps for these models.

Backbone	Δ Calorie	Δ Mass	Δ Fat	Δ Carb	Δ Protein
ResNet50	-1.26 / -6.53%	+56.53 / +26.74%	+0.75 / +6.3%	+0.88 / +4.84%	+0.01 / +0.01%
ResNet101	+7.43 / +3.11%	+4.63 / +2.19%	-0.09 / -0.76%	+0.13 / +0.74%	+0.28 / +1.71%
ResNet152	+12.39 / +5.2%	+7.41 / +3.51%	+0.23 / +1.9%	+0.17 / +0.93%	+0.17 / +1.00%
MobileNetV3	+1.89 / +0.79%	+0.79 / +0.38%	+0.11 / +0.97%	-0.08 / -0.46%	+0.10 / +0.59%

Table 3: Comparison of performance between without/with augmentation

As shown in the table, all four models we chose increased the MAE more or less after using the augmented images, which means the augmentation step had a negative impact on the nutrient prediction. However, this degradation is not consistent for these four models: ResNet50 did better in predicting calories, but significantly worse in mass; ResNet101 and ResNet152, on the other hand, had a larger MAE increase on calories than on mass; MobileNetV3 had the least impact from adding augmentation step. The reason why augmented images could deteriorate models performance might be that augmentation adds too much noise, preventing models to correctly extract useful features for later nutrient estimation. In terms of nutrient perspective, the MAE difference varies largely in calories and mass, yet for fat, carb and protein, the difference is much smaller. This is because calories and mass have greater magnitude of values than fat, carb and protein, which in the context of multi-task learning have less disturbance.

5.4 Real-world Inference

We tested our model on real-world food images, below are the inferred results of macronutrients and calories of a hamburger, a plate of fried rice and a plate of Kung Pao chicken.

Figure 6: One Delicious Hamburger [6]

Calorie: 680.8314, Mass: 439.7498, Fat: 36.7070, Carb: 40.7099, Protein: 42.6074

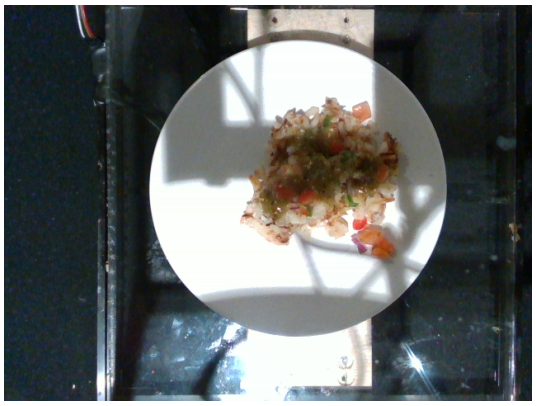


Figure 7: Fried Rice [14]

Calorie: 161.3916, Mass: 53.4847, Fat: 10.0680, Carb: 6.9713, Protein: 10.1660

Figure 8: Kung Pao Chicken [5]

Calorie: 568.4076, Mass: 400.8180, Fat: 29.7444, Carb: 35.7798, Protein: 35.4848

6 Discussion

Based on the results from Section 5, the following observations can be made.

First, deeper backbones tend to lead to lower mean-absolute-error values. This can be best demonstrated using models of the same class. Given the same data set (either with or without data augmentation), ResNet152 exhibits the lowest MAE values for all categories, while ResNet101 has the second-lowest and ResNet50 has the highest of the three. The main purpose of the backbone is to capture critical features from the given images, and while why deeper networks work better is still not concretely understood, the task of image segmentation might be a quintessential "deep" and hierarchical process [2]. By the Universal Approximation Theorem, although shallow networks should theoretically perform the same as deeper networks, deeper networks generally require fewer neurons. Moreover, models with deeper backbones converged much faster in our experience while the shallow ones tended to oscillate during earlier training loops. Hence, the depth of the backbone network has a direct impact on the performance of this task. We also found that the ViT model didn't outperform ResNets, our inference for this situation is that though the vanilla ViT can obtain great results on classification tasks, it may face difficulties when applied to the object detection and semantic segmentation.

Second, the data augmentation we applied to the vanilla data did not lead to any better performance. Apart from center-cropping, we experimented with random perspectives, random horizontal flips, and random rotations. Their rationale is to create the effect of "side-angled" images to hopefully improve the robustness of the network. From the results, the models trained using augmented data had overall worse performance than their counterparts trained using the vanilla images. One possible explanation is that the extra augmentation techniques did not add useful values to the data. Center-cropping is necessary since it allows the network to capture critical features in a canonical position for the recognition filters. Apart from it, more augmentation might lead to image distortions, which might be fine for classification tasks but unsuitable for estimation tasks. If time is permitted, we could incorporate depth images for volume estimation which could lead to better results instead of using data augmentation.

7 Conclusion

In this report, we presented our attempt at nutrition estimation using an end-to-end approach while exploring possible improvements. We used a comprehensive food data set, Nutrition5k, and experimented with various backbones for feature extraction from the images. We showed that the depth of the backbone network plays a significant role in the performance of the model. We experimented with several data augmentation techniques that did not work as effectively.

We expect our investigation to add value to using neural networks as an end-to-end approach for nutrition estimation, with multi-task learning incorporated. With more robust backbones and more advanced data collection techniques such as depth images, there is potential for end-to-end nutrition estimation. Once more efficient backbones architectures are developed, we expect to see more and more applications deployed to mobile computing platforms that are universal and accessible to health-conscious users, lowering the barrier to getting started with a healthier lifestyle.

8 Division of Work

Sijie Luo

- Co-researched applications of neural networks in food analysis
- Co-developed code for the baseline model and multi-task learning
- Developed code for training
- Trained models with ConvNeXt as the backbone
- Supported writing of report
- Supported writing of presentation slide

Yiren Zhou

- Co-researched applications of neural networks in food analysis
- Developed code for data-set, data-loader, and performance evaluation
- Co-developed code for the baseline model and multi-task learning
- Trained models with MobileNetV3 and ResNet50 as the backbone
- Supported writing of report
- Supported writing of presentation slide

Yusheng Hu

- Co-developed code for data-set, data-loader, multi-task learning, and evaluation
- Coordinated communication between our mentor and the team
- Trained models with ConvNeXt as the backbone
- Co-researched applications of neural networks in food analysis
- Supported writing of report
- Supported writing of presentation slide

Zihan Wang

- Co-researched applications of neural networks in food analysis
- Co-developed code for data-set, data-loader, multi-task learning, and evaluation
- Trained models with ResNet50, ResNet101, ResNet152, and ViT as the backbone
- Tested the model on real-world images
- Supported writing of report
- Supported writing of presentation slide

References

[1] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. (2020). An image is worth 16x16 words: Transformers for image recognition at scale.

[2] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. `http://www.deeplearningbook.org`.

[3] He, K., Zhang, X., Ren, S., and Sun, J. (2015). Deep residual learning for image recognition. *CoRR*, abs/1512.03385.

[4] Howard, A., Sandler, M., Chu, G., Chen, L.-C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., Le, Q. V., and Adam, H. (2019). Searching for mobilenetv3.

[5] KIRBIE (2021). Kirbie's cravings.

[6] Larsen, L. (2021). Hamburger hot dogs.

[7] Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T., and Xie, S. (2022). A convnet for the 2020s.

[8] Mezgec, S. and Koroušić Seljak, B. (2017). Nutrinet: A deep learning food and drink image recognition system for dietary assessment. *Nutrients*, 9(7).

[9] Ruder, S. (2017). An overview of multi-task learning in deep neural networks.

[10] Ruede, R., Heusser, V., Frank, L., Roitberg, A., Haurilet, M., and Stiefelhagen, R. (2020). Multi-task learning for calorie prediction on a novel large-scale recipe dataset enriched with nutritional information. *CoRR*, abs/2011.01082.

[11] Sudo, K., Murasaki, K., Shimamura, J., and Taniguchi, Y. (2014). Estimating nutritional value from food images based on semantic segmentation. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication*, UbiComp '14 Adjunct, page 571–576, New York, NY, USA. Association for Computing Machinery.

[12] Sun, C., Shrivastava, A., Singh, S., and Gupta, A. (2017). Revisiting unreasonable effectiveness of data in deep learning era. *CoRR*, abs/1707.02968.

[13] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. (2015). Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567.

[14] Thames, Q., Karpur, A., Norris, W., Xia, F., Panait, L., Weyand, T., and Sim, J. (2021). Nutrition5k: Towards automatic nutritional understanding of generic food. *CoRR*, abs/2103.03375.

[15] Wang, W., Min, W., Li, T., Dong, X., Li, H., and Jiang, S. (2021). Vision-based food analysis for automatic dietary assessment. *CoRR*, abs/2108.02947.