Deep Learning Based Food Recognition

Qian Yu
Stanford University
qiany@stanford.edu

Dongyuan Mao
Stanford University
dmao@stanford.edu

Jingfan Wang
Stanford University
jingfan@stanford.edu

Abstract

Food safety and health is increasingly attracting attentions. An effective computer vision method to recognize the food category can efficiently help evaluate the food nutrition. We proposed a CNN-based food recognition method on the food recognition problem: the transfer learning and the fine-tuning on the whole architecture based on the Inception-ResNet and Inception V3 model. Our algorithm is performed on the Food-101 dataset and obtained impressive recognition results: Inception-ResNet converges much faster and achieves top-1 accuracy of 72.55% and top-5 accuracy of 91.31%. Our future work includes optimizing the network architecture and yielding a much higher leaning result. What's more, we will try to implement the recognition algorithm on the mobile devices and make it available in our practical daily lives.

1. Introduction

Food is the cornerstone of people's life. Nowadays more and more people care about the dietary intake since unhealthy diet leads to numerous diseases, like obesity and diabetes. Accurately labelling food items is significantly essential to help us keep fit and live a healthy life. However, currently referring to nutrition experts [1] or Amazon Mechanical Turk [2] is the only way to recognize the food category. In this project, we propose a deep learning based food image recognition algorithm to improve the accuracy of dietary assessment and analyze each of the network architecture. Further analysis was conducted on topics like which model results in the best accuracy, how the loss curve looks like, and how the optimizers like RMSprop or Adam optimizes the model.

2. Related Work

For food recognition, previous work mostly used traditional image processing techniques with hand-engineered features. These methods include relative spatial relationships of local features, feature fusion, manifold ranking-based approach and co-occurrence statistics between food items [3-5]. These methods either have poor adaptation to large scale, low recognition rate or are computational expensive. Recently, various machine learning methods are used for accurate recognition. Bossard et al. reported that classification accuracy on the Food-101 test set of 50.76% by mining discriminative components using Random Forests [6]. Basically, the random forest is applied to cluster the superpixels of the training dataset. Then discriminative clusters of superpixels are to train the component models. Bossard et al. also performed other advanced classification techniques, including Bag-of-Words Histogram (BOW) [7], Improved Fisher Vectors (IFV) [8], Random Forest Classification (RF) [9], Randomized Clustering Forests (RCF) [10], and Mid-Level Discriminative Superpixels (MLDS) [11]. Advanced deep learning methods, like Convolutional Neural Networks (CNN), were also used for food recognition. Bossard et al. made use of AlexNet [12] to achieve top-1 classification accuracy of 56.40%. Meyers et al. applied GoogLeNet Inception V1 and got the top-1 classification accuracy of 79% [13]. In this quarter, we had the conversation with authors of this paper. Surprisingly they said that they did use VGG instead of GoogLeNet Inception V1, which was mentioned in their paper. We thought this paper is not convincing. Liu, C. et al. applied inception model based CNN approach to two real-world food image data sets (UEC-256 and Food-101) and achieved impressive results [14].

3. Proposed Approach

3.1. Datasets

Deep learning-based algorithms require large dataset. The UPMC-FOOD-101 and ETHZ-FOOD-101 datasets are twin datasets [15,16]. Each one has the same class labels but different image files. UEC-FOOD-256 is a dataset of Japanese dishes [17]. Totally, the number of training samples is approximately 235000. In this project, we perform on the dataset of ETHZ-FOOD-101. There are also some online food image recourses like BigOven which has over 350000 samples. But unfortunately, this large database doesn't offer free large query API. In this project, we perform on the dataset of ETHZ-FOOD-101.

Table 1. Food Datasets

Dataset	# of dish classes	# of images per class	Data type
UPMC-FOOD-101	101	790-956	Text & image
ETHZ-FOOD-101	**101**	**1000**	**Image**
UEC-FOOD-256	256	150	Image

3.2. Model

CNN become increasingly powerful in large scale image recognition after Krizhevsky et al. won the first prize in ILSVRC 2012 with the introduction of AlexNet [12]. AlexNet has 60 million parameters and 650,000 neurons, consists of five convolutional layers. Those layers are followed by max-pooling layers, and three globally-connected layers with a final 1000-way softmax layer [12].

After that, there are several symbolic milestones in the history of CNN development, which are ZFNet [18] by Zeiler and Fergus, VGGNet [19] by Simonyan et al., GoogLeNet (Inception-v1) [20] by Szegedy et al and ResNet [21] by He et al.

GoogLeNet or Inception V1 was the winner of ILSVRC 2014. It largely reduced the ImageNet top-5 error from 16.4% which obtained by AlexNet to 6.7% [22]. The Inception deep convolutional architecture was introduced, with the advantages of less parameters (4M, compared to AlexNet with 60M) [20]. Average Pooling instead of Fully Connected layers at the top of the ConvNet was applied to eliminate unnecessary parameters. Later, there are several more advanced versions to Inception V1. Batch normalization was introduced in Inception V2 [23] by Loffe et al. Later the architecture was improved by additional factorization ideas in the third iteration which will be referred to as Inception V3 [24]. Inception V4 has a more uniform simplified architecture and more inception modules than Inception V3 [25]. Szegedy et al. designed Inception-ResNet to make full use of residual connections introduced by He et al. in [21] and the latest revised version of the Inception architecture. Training with residual connections accelerates the training of Inception networks by utilizing additive merging of signals [25].

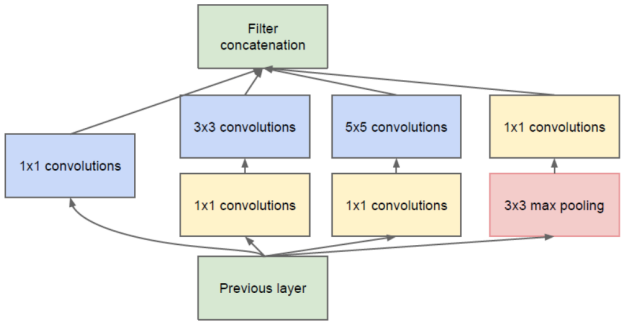

Figure 1. Inception Structure

3.3. Image Preprocessing

Tensorflow provides the API for loading and preprocessing raw images from the user. However, in our project, we still need to preprocess the input images. This is because the environmental background varies quite a lot in different food pictures. Those environmental factors are the color temperatures, luminance and so on. To have similar background environment, we utilize two methods which are Grey World method and Histogram equalization.

The white balance is processed by the Grey World Method which assumes the average of the RGB values are all similar to the one grey value. The Grey World algorithm is as followed, where α, β, γ are the scaling factors in the RGB color channels.

$$(\alpha R, \beta G, \gamma B) \rightarrow \left(\frac{\alpha R}{\frac{\alpha}{n}\sum_i R}, \frac{\alpha G}{\frac{\beta}{n}\sum_i G}, \frac{\alpha B}{\frac{\gamma}{n}\sum_i B}\right)$$

Then, we applied Histogram Equalization algorithm to increase the contrast and luminance. The image preprocessing result is shown as below. The first one is the image of a baby rib. The middle one is the image after the white balance and the right one is the one after both white balance and histogram equalization.

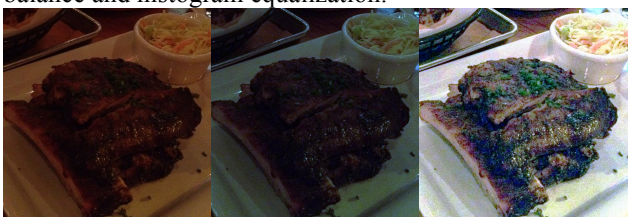

Figure 2. Image processing: (left) raw image, (middle) perform Grey World method, (right) perform Histogram Equalization on the middle image.

Image preprocessing effectively handles the problem when the pictures were taken in different environment background which speed up the learning pace and slightly improve the output accuracy. We will explore this more in the result section.

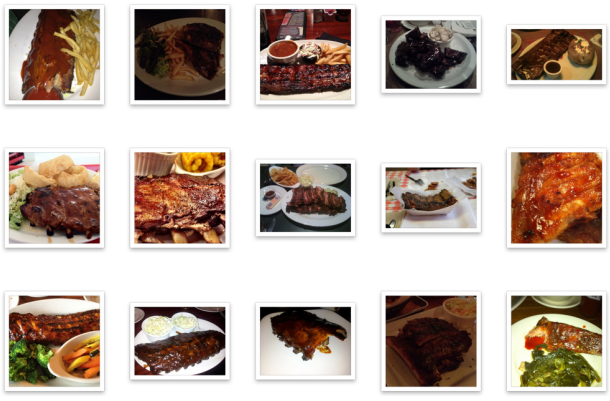

Figure 3: Images before preprocessing

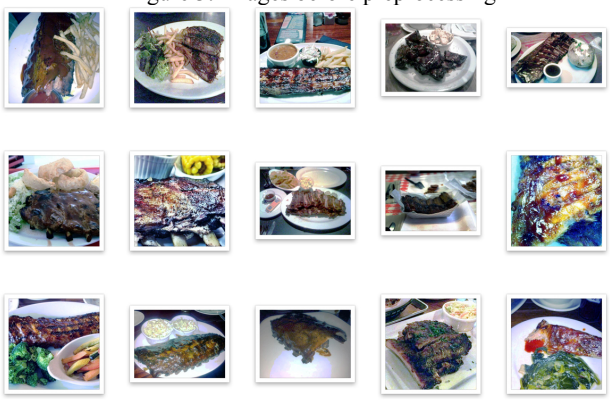

Figure 4: Images after preprocessing

3.4. Methodology

We utilized the Amazon AWS GPU resources to run the whole program. The AWS g2 instance has NVIDIA GRID K340 with 1536 CUDA cores and 4GB memory size which enables to run a complicated network structure. The framework for deep learning is the latest slim version of Tensorflow. We applied the transfer learning method to our model which is by using the pre-trained model as a checkpoint and continue to train the neural network. The reason that we can do so is because the ImageNet has very large amount of dataset and is trained quite well. We can make full use of the pre-trained model and get the feature based on the ImageNet dataset.

Firstly, use the pre-trained model of Inception V3 and Inception-ResNet. We load a checkpoint from the pre-trained model, which is a file that stores all the tensors after weeks of training for the ImageNet datasets.

Secondly, train the last layer of the network and recognize classes of food images. As we know that the last layer for the ImageNet classifier is 2048*1001. However, in our problem, we only have 101 food classes. Therefore, in this case, the dimension for the last layer is 2048*101. We replace the final 1001-way softmax with a 101-way softmax. Then we retrain the last layer to get a model based on the pre-trained model. However, we later find that the pure transfer learning can only obtain the testing accuracy around 40% which is far from the result we expected. To fix this problem, we try to train the whole layer instead of only training the last layer and obtain a much higher accuracy. We will discuss this more in the result section.

Thirdly, evaluate the result. At the first stage, we split the datasets into training and evaluation parts. The training examples take 80% of the whole datasets and the rest are considered as testing datasets. By running the evaluation on the last layer of the network, we obtain the training and testing error. We also evaluate the testing accuracy on the full layer model as discussed above.

Finally, fine tuning the network framework. We need to give the model initial parameters, and setup optimization process. For example, weight decay prevents overfitting. By adjusting the weight decay, we can balance between variance and bias. Also, optimizers like Adam and RMSprop need different initial parameters, such as learning rate and learning rate decay. To achieve the minimum loss, these parameters need be set carefully.

The training process is mainly divided into 2 processes: last layer with 0.01 initial learning rate and 101 batch size, full network with 0.001 initial learning rate and 7 batch size. Within each process, we compare the different results between raw image input and processed image input. Also, we compare different network structures: Inception V3 and Inception-ResNet.

4. Result

The results analysis will cover the following topics: optimizer selection, network analysis, model comparisons.

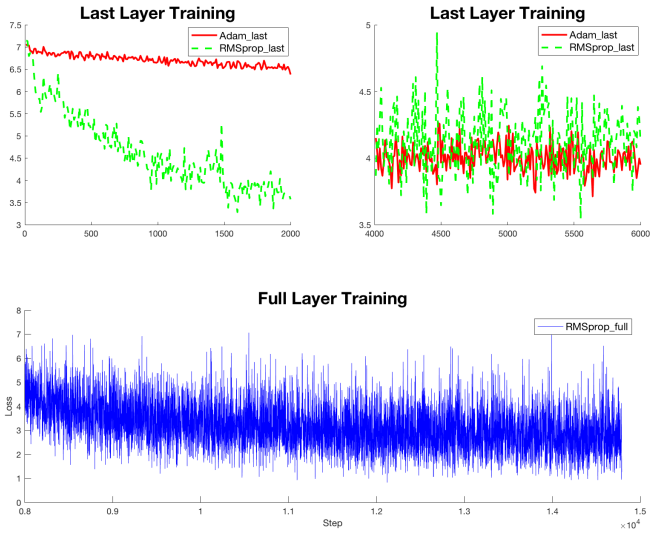

Figure 5. Loss curve of each iteration step with different optimizers

Figure 6. Results of Inception V3: Accuracy of each epoch

Figure 7. Results of Inception-ResNet: Accuracy of each epoch

4.1. Optimizer selections

Choice of optimizers greatly affect on loss descending process. This is the first thing we need to decide to achieve the global minimum. We compare two mostly used optimizers to see their performance. The first one is RMSprop, which is proposed to solve the premature convergence of Adagrad [26]. For each step, RMSprop divides the learning rate by average momentum of squared gradients and a constant, which prevents premature diminishing of learning rate. The second one is Adam, which reduces the loss fluctuation of RMSprop by introducing additional average momentum of gradients average [26]. It replaces the gradients with gradients average in the update of theta, which makes its loss descend more smoothly than that of RMSprop in stochastic gradient descent. Fig.5 compares these built-in optimizers under the last layer training and full network training. It can be seen that RMSprop descends faster at the beginning, but fluctuates a lot. While Adam descends slower but steadier. Therefore, we choose RMSprop at the beginning for its quicker convergence and Adam at the end for its stability. Also, the fluctuation in full network training is much greater than that in the last layer training. This is because the batch size of full network training is much smaller due to limited GPU memory.

4.2. Network analysis

Choice of network models determines final prediction accuracy after convergence of retraining. Here, we need to pick models that can be smoothly adapted to food recognition. Fig. 6&7 compares two best models on ImageNet – Inception V3 and Inception-ResNet.

Firstly, we only retrain the last layer (softmax layer) of the model. This means that the features extracted from ImageNet can be generalized to food recognition. The figures show that Inception-ResNet preforms better after convergence of the last layer retraining, meaning that the features extracted by Inception-ResNet is more general than those by Inception V3. Also, the comparison between raw image input and processed image input shows that preprocessing has no improvement on recognition, because the former layers of those models have already learned to preprocess image.

Secondly, we retrain the model on all layers. The Figures show that retraining on full network gains a large improvement on retraining only on the softmax layer. This suggests that the features extracted from ImageNet are not suitable for food recognition. Also, this time, preprocessed input has roughly 3% accuracy advantage to raw input. This is because certain neurons that are used for preprocessing are freed to extract additional features of food, which results in a better prediction. Additionally, on full network training, Inception V3 performs roughly the same as Inception-ResNet. This suggests that these two models have the same learning ability, but features of Inception-ResNet are more extensible. Lastly, overfitting occurs only when accuracy rise above 60%, and we got final overfitting of 3%.

4.3. Model comparisons

Table 2. Accuracy comparison (on Food-101 dataset)

Method	Top-1 Accuracy	Top-5 Accuracy
Bag-of-Words Histogram [6]	28.51%	NA
Randomized Clustering Forests [6]	28.46%	NA
Random Forest Classification [6]	32.72%	NA
Improved Fisher Vectors [6]	38.88%	NA

Method	Top-1 Accuracy	Top-5 Accuracy
Discriminative Components with Random Forests [6]	50.76%	NA
Mid-Level Discriminative Superpixels [6]	42.63%	NA
AlexNet [6]	56.40%	NA
GoogLeNet (10,000 iterations) [14]	70.2%	91.0%
GoogLeNet (300,000 iterations) [14]	77.4%	93.7%
Inception V3 (last layer training)	35.32%	62.97%
Inception-ResNet (last layer training)	42.69%	72.78%
Inception V3 (full layer training)	70.60%	90.91%
Inception-ResNet (full layer training)	72.55%	91.31%

From Table 2, Inception V3 and Inception-ResNet training on full layers outperforms other methods except GoogLeNet. Compard with other CNN architecture, AlexNet is the first architecture, which makes CNN popular. AlexNet is a 7-layer model, consisting 5 convolutional layers and 2 fully-connected layers. It was the winner of ImageNet ILSVRC challenge in 2012 with top-5 error of 16.4%, while Inception V1 reduced the top-5 error to 6.7%.

The main reason why our methods perform better is because the inception module (Figure 1) increases the representation power of neural network. The input is fed into an additional 1*1 convolutional layer instead of feeding into 3*3 and 5*5 convolutional layer to reduce the input dimension. Moreover, after the 3*3 max-pooling layer, the output is fed into an additional 1*1 convolutional layer, resulting in deeper depth and less dimensions. Multiple modules are used to form the GoogLeNet, making the network hierarchical step by step. Overall, the feature mapping contains much more information than before.

From Table 2, after 300,000 iterations, the deep learning method developed by Liu, C. et al. has better top-1 accuracy and top-5 accuracy than us. Actually, Liu, C. et al. also applied transfer learning and had very similar architecture of GoogLeNet with us. Then main reason why their results are better is that they have more iterations. Liu, C. et al. also reported that after 10,000 iteration steps, the top-1 accuracy is 70.2% and top-5 accuracy is 91.0%, which means our results are competitive since we only iterate 12,000 steps due to the limit of the computation resources. If we perform more iterations, our methods may have similar performance.

Overall transfer learning method results in good accuracy for food recognition. The ConvNet pre-trained on ImageNet is a good feature extractor for a new dataset. We also implement the fine-tuning on the weights of the pre-trained network by continuing the backpropagation. There are two important factors. One is the size of new dataset. Food 101 consists of 101*1000 images, which is large enough for CNN. The other one is that ImageNet-like in terms of the content of images and the classes. The images in Food-101 have great similarity to ImageNet.

As we can see from Table 2, Inception and ResNet model both output very nice recognition accuracy. This is because both models have very deep network structures. When the parameters are all fit on the training set, the network can make very accurate generalization on the testing set. However, since ResNet has the residual structure which adds the forward propagation in the network, the ResNet can accelerate the learning speed and result in much less information loss in the network iterations.

5. Conclusion and Future Work

In conclusion, fine-tuning on all the layers significantly increase the recognition accuracy with updated weights. In addition, Inception-ResNet outputs better recognition accuracy than the Inception-V3 because of the residual structure.

Here are something that we can explore more on this food recognition problem. The time we choose to evaluate the training and testing accuracy is after processing the whole training dataset. However, it may not be the time when we reach the lowest loss, instead, it is a lower bound of the evaluating accuracy. Therefore, the result we got when using the Inception-ResNet is possibly not the best result we could get from the current model. To get a better result, we can set up a loss threshold for doing the evaluation. We expect a higher training and testing accuracy but we will probably encounter an overfitting problem. The second thing we can do is to do slight modification to the ResNet and train the whole network on a powerful GPU instead of using the pre-trained model based on the ImageNet challenge. However, this may take quite long time to do because Food-101 is a large dataset and require weeks of training time. But we may get a better result since we can find the best network structure and parameters that fits the problem well. In the previous work, Liu, C. et al. has done the bounding box which did image segmentation first and then do the classification. This would effectively increase the testing accuracy since we can eliminate other non-related objects and keep the main part of the food. This requires manual cropping or automatic segmentation which makes the model even more complicated and will give a much better result. Extracting features in the last layer is a very common way to visualize the main feature component in each class and will give us very straightforward sense of why the CNN can understand each dish.

References

[1] Martin, C., Correa, J., Han, H., Allen, H., Rood, J., Champagne, C., Gunturk, B., Bray, G.: Validity of the remote food photography method (RFPM) for estimating energy and nutrient intake in near real-time. Obesity (2011)

[2] Noronha, J., Hysen, E., Zhang, H., Gajos, K.Z.: Platemate: crowdsourcing nutritional analysis from food photographs. In: ACM Symposium on UI Software and Technology (2011)

[3] S. Yang, M. Chen, D. Pomerleau, and R. Sukthankar, "Food recognition using statistics of pairwise local features," in Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, 2010.

[4] Y. Matsuda and K. Yanai, "Multiple-food recognition considering co-occurrence employing manifold ranking," in Pattern Recognition (ICPR), 2012 21st International Conference on, 2012.

[5] TADA: Technology Assisted Dietary Assessment at Purdue University, West Lafayette, Indiana, USA, available at http://www.tadaproject.org/.

[6] Bossard, L., Guillaumin, M., & Van Gool, L. (2014, September). Food-101–mining discriminative components with random forests. In European Conference on Computer Vision (pp. 446-461). Springer International Publishing.

[7] Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In: CVPR (2006)

[8] S´anchez, J., Perronnin, F., Mensink, T., Verbeek, J.: Image Classification with the Fisher Vector: Theory and Practice. IJCV (2013)

[9] Bosch, A., Zisserman, A., Munoz, X.: Image Classification using Random Forests and Ferns. In: ICCV (2007)

[10] Moosmann, F., Nowak, E., Jurie, F.: Randomized clustering forests for image classification. PAMI (2008)

[11] Singh, S., Gupta, A., Efros, A.A.: Unsupervised discovery of mid-level discriminative patches. In: ECCV (2012)

[12] Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS (2012)

[13] Meyers, A., Johnston, N., Rathod, V., Korattikara, A., Gorban, A., Silberman, N., ... & Murphy, K. P. (2015). Im2Calories: towards an automated mobile vision food diary. In Proceedings of the IEEE International Conference on Computer Vision (pp. 1233-1241).

[14] Liu, C., Cao, Y., Luo, Y., Chen, G., Vokkarane, V., & Ma, Y. (2016, May). DeepFood: Deep Learning-Based Food Image Recognition for Computer-Aided Dietary Assessment. In International Conference on Smart Homes and Health Telematics (pp. 37-48). Springer International Publishing.

[15] UPMC-FOOD-101, http://webia.lip6.fr/~wangxin/

[16] ETHZ-FOOD-101, https://www.vision.ee.ethz.ch/datasets_extra/food-101/

[17] UEC-FOOD-256, http://foodcam.mobi/dataset256.html

[18] Zeiler, M. D., & Fergus, R. (2014, September). Visualizing and understanding convolutional networks. In European Conference on Computer Vision (pp. 818-833). Springer International Publishing.

[19] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

[20] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... & Rabinovich, A. (2015). Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 1-9).

[21] He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep residual learning for image recognition. arXiv preprint arXiv:1512.03385.

[22] Convolutional Neural Networks (CNNs / ConvNets), http://cs231n.github.io/convolutional-networks/

[23] Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167.

[24] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2015). Rethinking the inception architecture for computer vision. arXiv preprint arXiv:1512.00567.

[25] Szegedy, C., Ioffe, S., & Vanhoucke, V. (2016). Inception-v4, inception-ResNet and the impact of residual connections on learning. arXiv preprint arXiv:1602.07261.

[26] An overview of gradient descent optimization algorithms, http://sebastianruder.com/optimizing-gradient-descent/