A Food Photography App with Image Recognition for Thai Food

Ukrit Tiankaew*, Peerapon Chunpongthong† and Vacharapat Mettanant‡

Department of Computer Engineering,
Faculty of Engineering at Sriracha,
Kasetsart University Sriracha Campus, Chonburi, Thailand.

Emails: *ukrit.t@ku.th, †peerapon.ch@ku.th, ‡vacharapat@eng.src.ku.ac.th

***Abstract*—In this paper, we present a food photography application for smart phones, which can recognise 13 types of Thai food from photos. With this feature, the application can easily help users calculate their calories and make some suggestion, just by keep taking a photo of food they are eating. Our application uses React Native for the front-end, and Python-Flask for the back-end. For image recognition, we design a deep convolutional neural network to learn from our dataset. Moreover, we compare the result of our model with another model adapted from the famous one of Karen Simonyan and Andrew Zisserman called VGG19. We use transfer learning from the pre-trained VGG19, implementing with Keras and Tensorflow. Our result shows that the transfer learning model is better. It give us approximately 82% test accuracy or 18% top-1 error rate. Using top-3 and top-5 scores, The model reports 2.6% top-3 error rate and 1.3% top-5 error rate, which works well in our application.**

***Index Terms*—Food Photography, Deep Learning, Image Recognition**

I. Introduction

Healthcare is one of the most important issues around the world at present, It comes from many health problems that still grow continuously. For example, it is shown that 32.2% of Thai people have their weight above standard, which ranks the second highest in ASEAN [1]. One reason is, we found that people who were born between 1982 and 2002 spend 77% of their expenses on food [1].

There are many ways to improve health. One of them is to control the calories, that is, to receive calories in the appropriate amount. It is known that a man should take approximately 2500 kilocalories daily, while a woman should take only approximately 2000 kilocalories [2].

Nowadays, there are many mobile applications that help their users control calories. The main feature of these applications is getting information about what the users eat each day, and suggest how much to eat in the future. One interesting application is Calorie Mama AI [3]. It uses deep learning techniques to automatically count calories by taking food photos. It also supports planning meals for good health, trained using information from personal trainers. However, we found that the food recognition feature in Calorie Mama AI is not familiar with Thai local food. It makes a lot of wrong prediction, and does not have information on some specific kinds of Thai local food.

There are some calorie counter applications that are build for Thai people, Calorie Diary for instance [4]. Even if it does not come with food photo recognition, Calorie Diary has a lot of information about Thai food which all Thai people find in everyday life. Furthermore, it supports barcode scanning to receive and record food easily found in Thailand.

Our contribution in this paper is to combine the food photo recognition in Calorie Mama AI within a calorie counter application that is build for Thai people like Calorie Diary. We develop a new calorie counter application with information about Thai food. This new application has the feature of Thai food photo recognition, with a deep learning model that is trained using Thai food photos.

To build a good classifier, we have done in two different ways. First, we design a deep convolutional neural network (CNN) and train it with a training dataset of Thai food. In the other way, we modify the famous network of Karen Simonyan and Andrew Zisserman called VGG19 [5]. Then we use transfer learning to train the modified network with our dataset.

The next section reviews some previous results about CNNs and image classification. We describe the architecture of our network in Section III. Section IV describes the details on transfer learning from VGG19. Then we describe some technical details and the results in Section V. In Section VI, we show our calorie counter application using the CNN as its classifier. Finally, we conclude this paper in Section VII.

II. Previous Results on CNNs

CNNs have been used in image recognition since the 1980s [6]. Recently, CNNs have very successful results in visual problems, achieve superhuman performance on some of them. To measure the performance of a CNN architecture, there are visual task competitions that prepare datasets for training and testing. One popular challenge is the ILSVRC ImageNet [7]. One of its tasks is to classify images into 1,000 classes. The training set contains 1.2 million images, while the validation and test data consist of 150,000 images. In this section we look at some CNN architectures that have a good performance on ImageNet, to understand basic structures of good CNNs.

978-1-5386-7805-3/18/$31.00 ©2018 IEEE

The first network we study, called AlexNet, was proposed by Krizhevsky, Sutskever, and Hinton [8]. The architecture of AlexNet is close to the classical architecture of LeNet-5 which is the first model that introduced convolutional layers and pooling layers [9]. Instead of using average pooling layers as in LeNet-5, AlexNet inserts a max pooling layer after a few convolutional layers. The architecture ends with three fully connected layers. This CNN architecture won the ILSVRC ImageNet in 2012 with 17% top-5 error rate.

In 2013, Zeiler and Fergus proposed a way to improve the model architecture from AlexNet. Their architecture outperforms AlexNet and won the ILSVRC ImageNet in 2013 [10].

In 2014, Simonyan and Zisserman showed that increasing the depth of a CNN by adding more convolutional layers can improve the performance [5]. One of their best model architecture has 19 weighted layers, called VGG19. In the same year, Szegedy et al. proposed an architecture called GooLeNet, introducing inception modules [11]. Both VGG network and GooLeNet have less that 7% top-5 error rates on the ImageNet.

In 2015, He, Zhang, and Ren presented a residual network that makes training CNNs become possible in a very deep network [12]. Their ResNet with 152 layers achieved 3.57% top-5 error rate and won the ILSVRC ImageNet in 2015.

While new techniques are still being developed continuously, we think that standard architectures like AlexNet and VGG19 are easy to design and should work well for specific tasks. Therefore, we choose VGG19 to be our base architecture. The VGG19 architecture is shown in Table I where convolutional layer parameters are denoted as conv(receptive field size)-(number of channels).

TABLE I
THE ARCHITECTURE OF VGG19.

Layer number	Configuration
	input (224×224 RGB images)
1	conv3-64
2	conv3-64
	maxpool
3	conv3-128
4	conv3-128
	maxpool
5	conv3-256
6	conv3-256
7	conv3-256
8	conv3-256
	maxpool
9	conv3-512
10	conv3-512
11	conv3-512
12	conv3-512
	maxpool
13	conv3-512
14	conv3-512
15	conv3-512
16	conv3-512
	maxpool
17	fully connected 4096
18	fully connected 4096
19	fully connected 1000
	softmax

III. CNN FOR THAI FOOD RECOGNITION

In this section, we first describe our classification problem and the process of getting the dataset. Then we describe the details of our customized CNNs and their results.

A. Problem and dataset

In this paper, we consider 13 classes of different kinds of Thai local food, consist of

1) Fried pork/chicken with garlic and pepper on rice,
2) Stir-fried pork/chicken with Thai red curry paste on rice,
3) Plain noodles,
4) Noodle soup,
5) Stir-fried pork/chicken with Thai basil on rice,
6) Thai-style omelet with rice,
7) Fried rice,
8) Chicken rice,
9) Thai-style fried chicken with rice,
10) Barbecued red roast pork with rice,
11) Crispy pork belly with rice,
12) Stir-fried chinese broccoli with crispy pork belly on rice,
13) Stir-fried noodle with black soy sauce.

Hence, our image recognition problem is to classify an image into one of these 13 classes.

To build a good image recognition model using CNNs, we need a dataset. Our dataset is obtained by developing a script to access Google Images using the food name of each class as a keyword, and load the images suggested by Google.

The image set we get from Google Images is noisy. Therefore, we manually clean the dataset by hands at last. We finally get 7,632 total images of all classes as shown in Table II. Each image is resized to be 160×160 pixels. The images are separated to 80% for training and 20% for testing.

B. The first CNN architecture

To create a CNN model architecture, we follow the architecture of the famous networks we studied. If we consider those networks, the architecture of them share some design templates. They consist of blocks of convolutional layers and a maxpool layer. After a number of such blocks, they have some fully connected layers at the end.

With this template of model architecture, we create a set of simple model architectures, train them with our training set, and measure their accuracy on the test set. Our highest test accuracy rate comes from a model architecture that starts with three blocks of a convolutional layer with a maxpool layer. Then the blocks are followed by two fully connected layers of 512 nodes and 13 nodes, separated by a dropout layer to prevent overfitting. We use ReLU activation function

TABLE II
THE DATASET

Class ID	Number of images
1	420
2	185
3	588
4	477
5	410
6	1048
7	1896
8	697
9	327
10	469
11	327
12	452
13	336

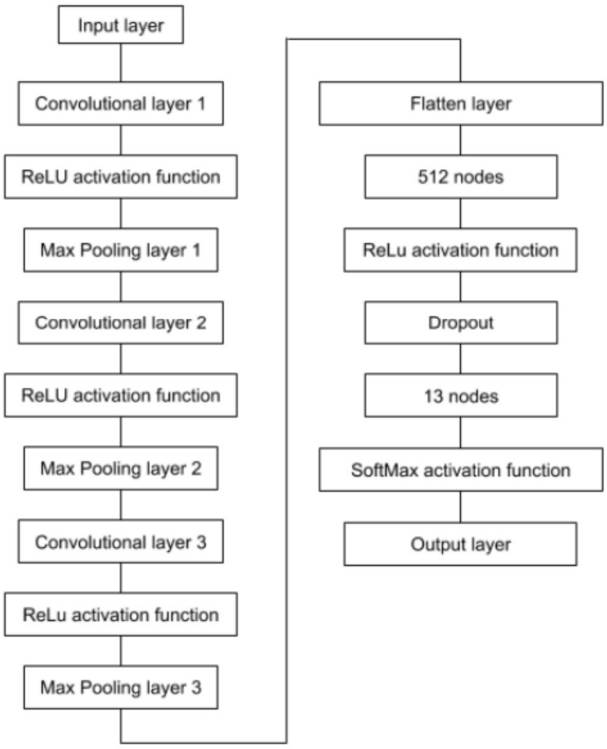

Fig. 1. The first CNN architecture.

for all the weighted layers except the output layer, which uses the softmax function. This model architecture is shown in Figure 1.

IV. TRANSFER LEARNING USING VGG19

Besides the model architecture described in the last section, we modify some parts of VGG19's structures and train this model using transfer learning.

Deep CNNs have been studied for a long time for better understanding. Recently we know that after training, the first layers of CNNs try to extract some low-level features from

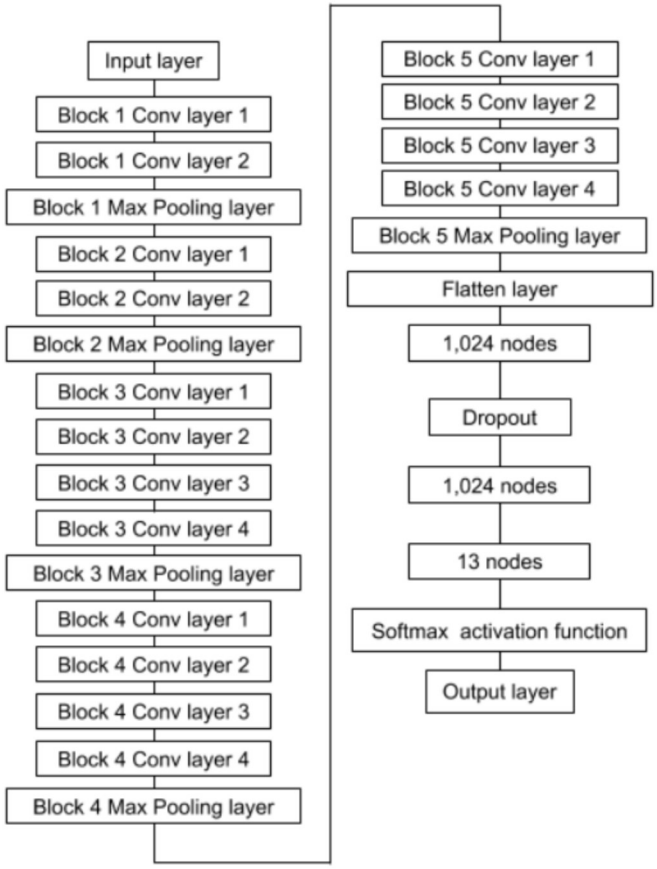

Fig. 2. A new CNN model modified from VGG19.

images, such as a vertical edge or a plain texture in white. The following layers can use these informations to extract more complex features and so on. For this reason, using a CNN model with its pre-trained parameters can make progress much faster. There are some techniques that can be done in transfer learning. For example, we can freeze the values of parameters in some of the first layers, and use our training data to tune the rest parameters in the network. The reason is, most image recognition task share the same low-level features.

A new CNN model architecture we use in this section is modified from VGG19. We remove three fully connected layers from VGG19, and replace them with a fully connected layer with 1024 nodes, followed by a dropout layer, another fully connected layer with 1024 nodes, and the output layer with 13 nodes using softmax as its activation function, respectively. This modified VGG19's structure is shown in Figure 2. We also use ReLU as the activation function of all layers except the output layer.

To train this model, we initialize all parameters to be the pre-trained parameters from VGG19 using the ImageNet dataset. Afterwards, we freeze all parameters in the first five layers to be non-trainable. Then we train the rest parameters using the training images in our dataset.

TABLE III
THE ACCURACY OF THE MODIFIED VGG19 FOR THAI FOOD RECOGNITION.

Score	Error rate
top-1	18%
top-3	2.6%
top-5	1.3%

V. IMPLEMENTATION AND RESULTS

This section describes technical tools we use, along with the result of our models.

A. Technical implementation

In the process of getting and preparing data, we use Python and Selenium to write a script that search for images of each class on Google and save all the suggested images for us. After removing noise by hands, we encrypt the image set of each class to be an HDF5 (.h5) data file. We publish all files of this dataset available online for research and educational use [13].

To build CNN models, we use Keras and TensorFlow on Python. With some configuration, we can use the GPU to train our model with Keras and TensorFlow. In the training process, Keras has many data augmentation features to help us expand the size of our dataset. The data augmentation operations we choose are zoom, rotate, width shift, height shift, and horizontal flip.

We train all our models on Intel(R) Core(TM) i5-6200U CPU @ 2.3GHz. (4 CPUs) with 12 GB of RAM and NVDIA GeForce 920MX GPU. For the modified VGG19 model, an epoch of training uses approximately 12-15 minutes.

B. Experimental results

For the first 5-layers CNN, we set the filter size of each convolutional layers to be 32, 64, and 64, respectively. The stride values are 5, 3, and 2. Then we train the model using minibatch size of 32 samples with Adam optimization algorithm proposed by Kingma and Ba [14]. The learning rate is 0.001 and the number of epoch is 20. Our trained model has approximately 63.3% of test accuracy, or 36.7% top-1 error rate. The performance of this model does not satisfy our desired goal.

For the modified VGG19 model, with 5 layers frozen, the accuracy of the training set and the test set are shown in Figure 3. The model achieves 82% test accuracy. In other words, this model has 18% top-1 error rate, which is more better than the previous model. When we test this model with top-3 and top-5 error rates, the model reports 2.6% top-3 error rate and 1.3% top-5 error rate, which make our application work very well. We conclude the modified VGG19 results for this classification problem again in Table III.

VI. THE MOBILE APPLICATION

In this section, we describe technical details of our application. This application is called Calpal. We developed this application using React Native for the front-end, so we can deliver the application cross-platform. For the back-end, we use Python with Flask framework to setup the web server, and use Firebase to store all the data. The trained CNN is located on the server.

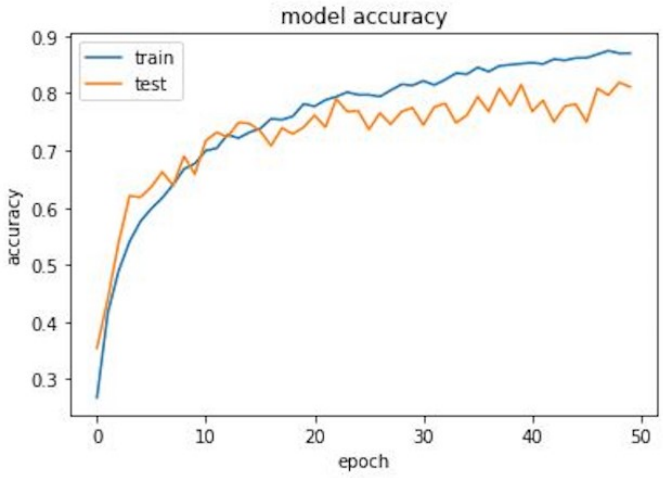

Fig. 3. The accuracy of the modified VGG19 on the training set and the test set.

Fig. 4. The screen of Calpal when the user takes a photo.

When users take a photo of food they are going to eat using Calpal, the application sends this photo to the server. Then the server asks the CNN to predict top-3 possible classes of food in the photo. The server sends this information back to the application to show as a prediction list of food on the screen. The user can choose the right class easily, or add another type of food if nothing matches. Figure 4 shows the screen of Calpal when the user takes a photo of food. The prediction list is shown in Figure 5.

In Calpal, we also calculate appropriate calories of users depend on their personal information such as weights and

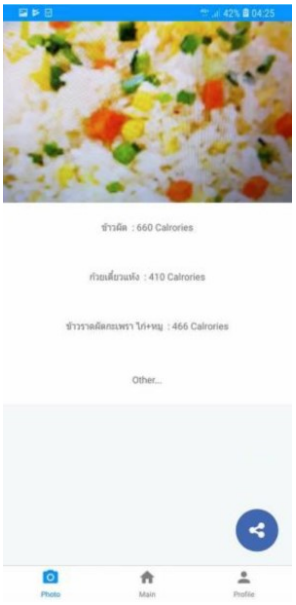

Fig. 5. The prediction list of foods in Calpal.

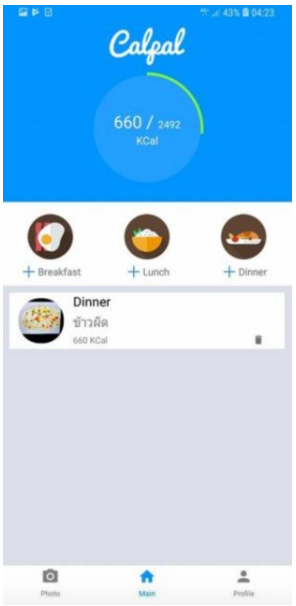

Fig. 6. The user profile screen.

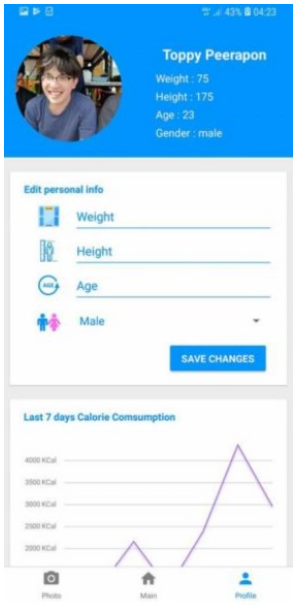

Fig. 7. The main screen.

ages. Moreover, we track the calories of users from their food photos, by looking in the calorie table. Figure 6 shows the main screen of the application and Figure 7 shows the profile screen that has historical track of a user. We has a Facebook share button to post a food photo to the user's Facebook profile wall, along with the calorie information. We wish these all features can make controlling calories more enjoyable.

VII. Conclusion

In this paper, we present a calorie counter application for smart phones called Calpal. Our application can classify 13 types of Thai food from photos. To build the classifier, we try designing a simple CNN model architecture and adapting the VGG19 model. The results show that using transfer learning in the modified VGG19 model works distinctly better. The model archives 18% top-1 error rate, 2.6% top-3 error rate, and 1.3% top-5 error rate.

While our application works well for its job, there are more than only these 13 types of Thai food we find in everyday life. One open problem is to improve the prediction model to be smarter to recognize various types of Thai food, and predict the calories more accurately.

References

[1] B. Bangkok. (2017) New thai generation health trend. [Online]. Available: http://www.bltbangkok.com/Health/

[2] Pobpad. Calories, how to calculate for good health. [Online]. Available: https://www.pobpad.com/

[3] Azumio. (2017) Calorie mama instant food recognition, a smart camera app that uses deep learning to track nutrition from food images. [Online]. Available: http://www.caloriemama.ai/

[4] Dimo. (2017) Calorie diary. [Online]. Available: http://www.mycaloriediary.com

[5] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014. [Online]. Available: http://arxiv.org/abs/1409.1556

[6] A. Gron, *Hands-On Machine Learning with Scikit-Learn and Tensor-Flow: Concepts, Tools, and Techniques to Build Intelligent Systems*, 1st ed. O'Reilly Media, Inc., 2017.

[7] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, ser. NIPS'12. USA: Curran Associates Inc., 2012, pp. 1097–1105. [Online]. Available: http://dl.acm.org/citation.cfm?id=2999134.2999257

[9] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[10] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," *CoRR*, vol. abs/1311.2901, 2013. [Online]. Available: http://arxiv.org/abs/1311.2901

[11] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," *CoRR*, vol. abs/1409.4842, 2014. [Online]. Available: http://arxiv.org/abs/1409.4842

[12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[13] V. Mettanant. (2018) Thai food photo recognition dataset. [Online]. Available: https://github.com/vacharapat/thai-food-photo-recognition-dataset

[14] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014. [Online]. Available: http://arxiv.org/abs/1412.6980