NU-InNet: Thai Food Image Recognition Using Convolutional Neural Networks on Smartphone

Chakkrit Termritthikun, Paisarn Muneesawang and Surachet Kanprachar
Department of Electrical and Computer Engineering, Faculty of Engineering,
Naresuan University, Phitsanulok, Thailand.
surachetka@nu.ac.th

***Abstract*—Currently, Convolutional Neural Networks (CNN) have been widely used in many applications. Image recognition is one of the applications utilizing CNN. For most of the research in this field, CNN is used mainly to increase the effectiveness of the recognition. However, the processing time and the amount of the parameters (or model size) are not taken into account as the main factors. In this paper, the image recognition for Thai food using a smartphone is studied. The processing time and the model size are reduced so that they can be properly used with smartphones. A new network called NU-InNet (Naresuan University Inception Network) that adopts the concept of Inception module used in GoogLeNet is proposed in the paper. It is applied and tested with Thai food database called THFOOD-50, which contains 50 kinds of famousThai food. It is found that NU-InNet can reduce the processing time and the model size by the factors of 2 and 10, respectively, comparing to those obtained from GoogLeNet while maintaining the recognition precision to the same level as GoogLeNet. This significant reduction in the processing time and the model size using the proposed network can certainly satisfy users for Thai-food recognition application in a smartphone.**

***Index Terms*—Deep Learning; Food Recognition; Convolutional Neural Networks; Smartphone; Thai Food; Dataset; Inception.**

I. Introduction

Food image recognition is one of the crucial applications used these days. It allows smartphone users to know the name of the food. This is quite important for travelers who travel to foreign countries. It also helps them to be able to order food properly and know the information about the food, for example the amount of calories, possible allergies, and so on. Currently, image recognition application in a smartphone mainly requires a computer [1,2] since the recognition process requires a considerably amount of resources to serve the used database. If the size of the database is large, the limited resource in a smartphone cannot keep up processing the data. Thus, the smartphone must send the data to be processed at the third-party computer. The effectiveness of the recognition in this case then depends on the performance of the computer and the speed of the internet connection.

Research on food image recognition, however, has been mainly focused on the correctness [2–6] of the food name for the given food image. Many techniques are applied, for example using image segmentation [7, 8] to separate the food from the background image. This technique will increase the effectiveness of the food identification. However, it is not appropriate to be used in a smartphone application since more image processing is needed. To overcome such problem, one possible approach called Convolutional Neural Network (CNN) can be adopted. At present, CNN has been used widely with image recognition. In the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) for the year 2012, AlexNet [9] won the competition by using the resized RGB (Red, Green, and Blue) images as the input for CNN to learn and produce the output probabilities of the categorized classes. Further, CNN has been adopted by the winning competitors of ILSVRC, that are GoogLeNet [10] and ResNet [11] in 2014 and 2015, respectively.

CNN has been continuously studied and developed so that the recognition of the effectiveness of CNN is higher than that from the conventional techniques used in computer vision. CNN can help extracting the features including colors, textures, and shapes. Moreover, image classification can also be done by CNN. In a research [12], CNN has been applied to food image recognition by adopting AlexNet [9] and tuning the output in the fully connected layer from 4,096 to 6,144. With a large-scale food database of 2,000 categories, the pre-training technique was done with the tuned AlexNet, resulting in the extracted features with 6,144 vectors per image. These extracted features were tested with food images from the databases UEC-FOOD100 and UEC-FOOD256. It was found that the correctness of the food image recognition was improved. However, with AlexNet [9,12], the storage required is quite large; that is, at least 240 MB, which is not suitable to be used with a smartphone. SqueezeNet [13] has been developed to reduce the storage size, while keeping the effectiveness of the recognition. The storage size was lessened by the factor of 50 that is, 4.8 MB.

To further improve the effectiveness of recognition, GoogLeNet [10] was proposed by Google. It was developed under the concept of Inception module that allows CNN to analyze an image with 1×1, 3×3, and 5×5 filters. Having done these, the obtained effectiveness of recognition was better than that of AlexNet. Additionally, the dimension reduction can be achieved by adding a 1×1 convolutional layer prior to the 3×3 or 5×5 convolutional layers. The required storage was reduced by the factor of 4.8, in comparison to that of AlexNet, in which only 51.1 MB was needed.

Considering the Food Image Recognition in a smartphone, it can seen that not just the correctness of the recognition, but the processing time and the required storage (or parameters) are important as well. In this paper, all these three important factors are taken into account. A new network called NU-InNet (Naresuan University Inception Network) is proposed. The concept of the inception module

adopted by GoogLeNet was utilized and further improved. The aim is to reduce the processing time and the parameters in comparison to those obtained from GoogLeNet, while maintaining the recognition accuracy to be at the same level.

The organization of this paper is done as follows. The related technology is given in Section II. The proposed network in this paper is explained in Section III. In Section IV, the obtained results are presented and discussed. Finally, the conclusion is given in Section V

II. Related Technology

Considering the current computer technology, it is seen that the data can be processed much faster utilizing GPU (Graphics Processing Unit) to function with CPU (Central Processing Unit). This then allows the deep learning technique, which is widely used in object detection and image recognition, to perform much better in terms of the resulting accuracy in detection and recognition. Convolutional Neural Network (CNN), one of the variants of deep learning technique, is adopted in this paper to be used for Thai food image recognition. In this section, the detail about CNN and related ones will be given.

A. Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) is a type of feed-forward artificial neural network (ANN). It contains different layers, including the input layer, convolutional layer, pooling layer, and fully-connected layer. These layers are stacked on top of each other according to the CNN architecture in order to do the recognition tasks. These layers are explained briefly as follows.

Input layer is the layer that contains images of the training data and testing data. These images are in the format of RGB and the image size depends on the model used in the network. For example, an image of 256 × 256 pixels, the data contained in such image equal [256×256×3], where the number 3 refers to the 3 channels of RGB.

Convolutional layer is the important layer of CNN where the dot product between the filter and the particular volume of the input data is determined according to the filter size. The product starts from the position (0, 0) of the input data and moves one pixel (stride 1) at a time from the left to the right and from the top to the bottom of the image. The obtained output from these products is the activation map. For example, with a data of [224×224×3] and 96 3×3 filters, moving the filter 2 pixels (stride 2) at a time, the size of the achieved activation map will be [111×111×96].

Pooling layer is the layer put after a convolutional layer in order to reduce the representation parameters of the network, resulting in a less computational process for the following steps. The function to be used in this layer can be one of the non-linear functions, for example max pooling, average pooling, L2 pooling, and so on. Among these functions, max pooling is currently the most common pooling to be used since it has shown to give a better performance. For the max pooling, the maximum value of the considered elements in the filtering area will be selected. The parameters of the whole network will be reduced depending on the size of the filter and the striding step. For example, with the input data sized [111×111×96], using a 3×3 filter with striding step of 2, the resulting data size will be reduced to [55×55×96].

Fully-connected layer is the layer put at the end of the network. All activations in the preceding layers are connected to this layer. The layer reduces the size of the data to be one-dimension data.

CNN has been developed to recognize the data in the deep level by adding more hidden layers to the network. An image can be recognized in three dimensions: the width, the height, and the depth. The network will divide the image into parts and analyze each to extract the important features, for example, colors, shapes, textures, and so on. These features can certainly be used to classify the image.

B. Deep Learning Framework [14]

In order to develop CNN, the framework to be used must be specifically designed. The designed framework can be chosen depending on the computer language or the operation system that users are working with. Considering this research, it is focused on the application to be used with an Android smartphone; hence, one possible choice is the framework called Caffe, which is widely used, developed with C++. Normally, to develop an application for an Android device, Java is commonly used. Hence, to work with Caffe framework in order to develop an Android application, certainly C/C++ has to be adopted, Native Development Kit (NDK) has to be used.

III. The Proposed Network

In this section, the dataset, the proposed network architecture, and the implementation of such network in Thai food image recognition in a smartphone will be described.

A. THFOOD-50 Dataset

The dataset of 50 kinds of Thai popular food images were collected as shown in Figure 1. These images were collected from search engines, namely the Google, Bing, and Flickr. In each kind of Thai food, there are approximately 200 to 700 images. These images are divided into 2 groups; that is, 90% of the images are in the training group and 10% of the images are in the testing group. The images are resized to have the size of 256×256 pixels suiting the CNN model to be used.

Figure 1: Examples of Thai food images in the THFOOD-50 dataset

B. Proposed Network Architecture

Inception module used in GoogLeNet10 is adopted in the proposed network. Two versions of the network are

proposed. The architectures of these inception modules are shown in Figure 2.

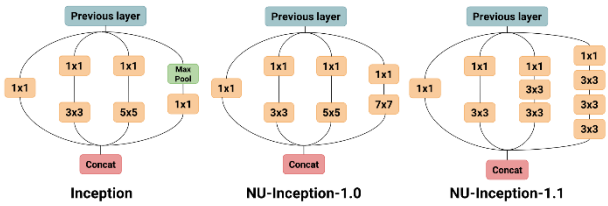

Figure 2: Module architecture of Inception (left), NU-Inception-1.0 (middle), and NU-Inception-1.1 (right)

Table 1
Detail of layers in NU-InNet 1.0

Type	Patch Size/Stride	Output Size
Input Image	-	224×224×3
Convolution	7×7/2	109×109×96
Max Pool	3×3/2	54×54×96
Convolution	1×1/1	54×54×96
Convolution	5×5/2	25×25×96
Max Pool	3×3/2	12×12×96
1× NU-Inception 1.0	As in Figure 2	12×12×256
Average Pool	-	1×1×256

Table 2
Detail of layers in NU-InNet 1.1

Type	Patch Size/Stride	Output Size
Input Image	-	224×224×3
Convolution	3×3/2	111×111×32
Convolution	3×3/1	109×109×32
Convolution	3×3/1	109×109×64
Max Pool	3×3/2	54×54×64
Convolution	1×1/1	54×54×64
Convolution	5×5/2	25×25×96
Max Pool	3×3/2	12×12×96
1× NU-Inception 1.1	As in Figure 2	12×12×256
Average Pool	-	1×1×256

a. NU-InNet 1.0

The inception module (as shown in Figure 2 (left)) is modified by changing its 3×3 max pooling layer and 1×1 convolutional layer to be 1×1 and 7×7 convolutional layers, respectively, as shown in Fig.2 (middle). The filter weights are set by "Xavier" with a constant filter bias of 0.2. And, for down sampling the size of the activation map after convolutional layers, the average pooling layer with a stride of 2 is used. The detail of this proposed NU-InNet 1.0 network is shown in Table 1 and Figure 3. There are totally 16 layers used in this network as shown in Figure 3 (middle). The details of these layers are given in Table 1.

b. NU-InNet 1.1

For this proposed network, the NU-Inception 1.0 module is modified by changing [15,16] any 5×5 convolutional layer to be 2 3×3 convolutional layers and changing any 7×7 convolutional layer to be 3 3×3 convolutional layers. These changes can be viewed in Figure 2 (right). By doing this, the processing time in the module will be lessened. Similarly, the detail of this proposed NU-InNet 1.1 network is shown in Table 2 and Figure 3. There is a total of 21 layers used in this network, as shown in Figure 3 (right). The detail of these layers is given in Table 2.

C. Implementation

The speed of data processing and the size of model are very crucial in designing a neural network, especially for its usage with a smartphone. In the proposed networks, one of the benchmark is the recognition accuracy that has to be at least not poorer than that from GoogLeNet, while the processing time and model size have to be lower. To obtain these properties, the number of modules in the proposed networks is set to be one module; while in GoogLeNet, nine modules were used as seen in Figure 3. The use of less number of modules decrease the processing time and model size; however, the recognition efficiency is also lessened. To improve the recognition efficiency, Batch Normalization [16] has to be put at the end of each convolutional layer as performed in ResNet [11]. By doing this, the training accuracy can be improved.

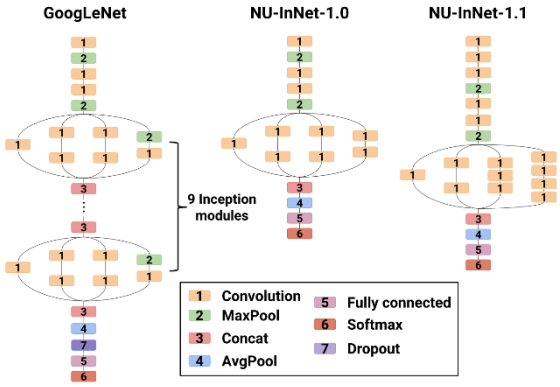

Figure 3: GoogLeNet, NU-InNet 1.0, and NU-InNet 1.1 architectures

IV. Results and Discussion

The proposed networks were used in Thai food image recognition. For the training process, a HPC (High-Performance Computer) with specifications Inter(R) Xeon(R) E5-2683 v3 @2.00-GHz 56-Core CPU, 64-GB RAM, and NVIDIA Tesla K80 GPU were used with the operating system Ubuntu Server 14.04.5 and Caffe14. For the testing process, a smartphone with specifications Intel(R) Atom(TM) Z3580 @2.33-GHz 4-Core CPU and 4-GB RAM was used with Android 6.0.1 operating system.

A. Comparisons between NU-InNets and Winning Models from ILSVRC

For the training process, AlexNet, GoogLeNet, SqueezeNet, NU-InNet 1.0, and NU-InNet 1.1 were trained from scratch. The database was divided into 2 parts; that is, 90% for training and 10% for testing. To get reliable testing results, 10-fold-cross-validation was used. The following hyper-parameters were used: adaptive gradient solver, mini-batch size of 64, learning rate of 0.01, weight decay of 0.0005, and epoch size of 100.

Three aspects of performance were studied. The first one was the accuracy obtained from the average of 10-fold-cross-validation. Top-1 and Top-5 accuracies were reported. The second one was the processing time required by each model. Batch size was set to be one in the data layer in order to allow CNN to process one iteration per image. 500 images were randomly selected and the average forward-backward time required per image was determined. The last one was the portability which contains two parameters; that

is, the total number of parameters and the storage required to store the trained model.

Table 3
Performance of NU-InNet 1.0, NU-InNet 1.1, and the winning models from ILSVRC

Model	Average Accuracy Top1 (%)	Average Accuracy Top5 (%)	Average Forward-Backward (ms/image)	Para-meters ($\times 10^6$)	Model size (MB)
AlexNet [9]	58.1	86.4	13.90	58.48	217
SqueezeNet [13]	58.2	87.4	24.53	0.75	2.86
GoogLNet [10]	68.4	91.7	40.13	10.45	39.9
NU-InNet 1.0	69.8	92.3	18.16	0.88	3.37
NU-InNet 1.1	68.7	92.3	36.52	0.89	3.42

The performance of the proposed models and the winning models from ILSVRC is shown in Table 3. Considering Top-1 and Top-5 accuracies, it is seen that the proposed NU-InNet 1.0 and 1.1 are better than the winning models from ILSVRC. For example, for Top-1 accuracy, the best accuracy is from NU-InNet 1.0, that is 69.8% accuracy. However, considering the average forward-backward time, it was found that the smallest one is from AlexNet (that is, 13.90 ms/image) since the architecture of AlexNet is less complicated. For the number of parameters and model size, it is seen that SqueezeNet requires the smallest number of these two factors, that are 0.75×106 and 2.86 MB, respectively.

Comparing NU-InNets with AlexNet in terms of the average forward-backward time, NU-InNet 1.0 requires slightly longer time to process, that is, 4.26 ms/image longer. While NU-InNet 1.1 requires the average time of approximately 2 times larger than that from Nu-InNet 1.0 since the number of layers in NU-InNet 1.1 is larger than the number of layers in NU-InNet 1.0. Additionally, considering NU-InNets with SqueezeNet in terms of the required number of parameters and the model size, it is seen that both NU-InNets require small numbers of parameters and model size similar to SqueezeNet.

From the previous discussion, it has been shown that the proposed NU-InNet 1.0 and 1.1 can deliver the accuracy with the same range as that from GoogLeNet. Moreover, as shown in Figyure 4, the obtained average of the forward-backward time, the number of parameters, and the model size of the proposed models are smaller, especially for the number of parameters and the model size. These impressive performances allow the proposed models to be appropriately utilized in a smartphone at which these factors are very limited.

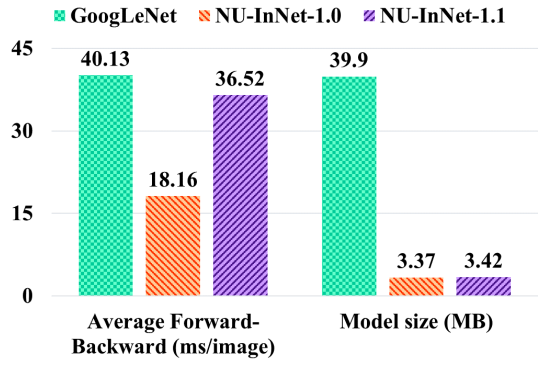

Figure 4: Comparisons between GoogLeNet, NU-InNet 1.0, and NU-InNet 1.1

B. Thai Food Image Recognition Application for Android

After the training process, the proposed models and GoogLeNet were applied to use with an Android smartphone in order to test for the required execution time. The tested image size is 1080×1080 pixels. The execution time for each model is shown in Table 4.

Table 4
Execution time required by GoogLeNet, NU-InNet 1.0, and NU-InNet 1.1

Model	Execution Time (ms)
GoogLeNet	906
NU-InNet 1.0	351
NU-InNet 1.1	691

From Table 4, it is seen that the use of NU-InNets significantly reduces the execution time in comparison to that from GoogLeNet. A reduction of 555 and 215 ms/image can be obtained from NU-InNet 1.0 and NU-InNet 1.1, respectively. The faster execution time in NU-InNets can then result in a fast responding time in the Thai Food Image Recognition application. Certainly, the use of these proposed models in the application improves the satisfaction of smartphone's users.

V. Conclusion

In this paper, NU-InNet 1.0 and 1.1 have been proposed. The inception concept in GoogLeNet was adopted and modified in the proposed models. A new Convolutional Neural Network (CNN) was designed. The main objective of the proposed models was to reduce the processing time and the model size, while keeping the accuracy not to be poorer than that of GoogLeNet so that the proposed models can be used in a smartphone. The proposed models have been tested with the dataset THFOOD-50 which contains images of 50 famous Thai menu. It is found that the accuracies obtained from NU-InNet 1.0 and 1.1 are slightly better than that of GoogLeNet. The required processing time is reduced by a factor of 2 for the case of NU-InNet 1.0 comparing to GoogLeNet. The model size from both proposed models is less than one-tenth of the model size required by GoogLeNet. It is clearly shown that the significant reductions in terms of processing time and model size from the proposed NU-InNet 1.0 and 1.1 can lead to a more suitable Thai Food Image Recognition application in a smartphone.

Acknowledgments

This work was supported by Naresuan University, Thailand.

References

[1] T. Maruyama, Y. Kawano, and K. Yanai, "Real-time mobile recipe recommendation system using food ingredient recognition," in *Proceedings of the 2nd ACM international workshop on Interactive multimedia on mobile and portable devices*, 2012, pp. 27-34.

[2] N. Tammachat and N. Pantuwong, "Calories analysis of food intake using image recognition," in *Information Technology and Electrical Engineering (ICITEE), 2014 6th International Conference on*, 2014, pp. 1-4.

[3] M. M. Anthimopoulos, L. Gianola, L. Scarnato, P. Diem, and S. G. Mougiakakou, "A food recognition system for diabetic patients based on an optimized bag-of-features model," *Biomedical and Health Informatics, IEEE Journal of,* vol. 18, pp. 1261-1271, 2014.

[4] Y. Kawano and K. Yanai, "Foodcam: A real-time food recognition system on a smartphone," *Multimedia Tools and Applications,* pp. 1-25, 2015.

[5] V. Bettadapura, E. Thomaz, A. Parnami, G. D. Abowd, and I. Essa, "Leveraging context to support automated food recognition in restaurants," in *2015 IEEE Winter Conference on Applications of Computer Vision*, 2015, pp. 580-587.

[6] Y. Matsuda, H. Hoashi, and K. Yanai, "Recognition of multiple-food images by detecting candidate regions," in *Multimedia and Expo (ICME), 2012 IEEE International Conference on*, 2012, pp. 25-30.

[7] D. Mery and F. Pedreschi, "Segmentation of colour food images using a robust algorithm," *Journal of Food engineering,* vol. 66, pp. 353-360, 2005.

[8] Y.-W. Chang and Y.-Y. Chen, "An improve scheme of segmenting colour food image by robust algorithm," *Proc. Algo2006,* 2006.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097-1105.

[10] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, et al., "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1-9.

[11] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *arXiv preprint arXiv:1512.03385,* 2015.

[12] K. Yanai and Y. Kawano, "Food image recognition using deep convolutional network with pre-training and fine-tuning," in *Multimedia & Expo Workshops (ICMEW), 2015 IEEE International Conference on*, 2015, pp. 1-6.

[13] F. N. Iandola, M. W. Moskewicz, K. Ashraf, S. Han, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 1MB model size," *arXiv preprint arXiv:1602.07360,* 2016.

[14] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, et al., "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22nd ACM international conference on Multimedia*, 2014, pp. 675-678.

[15] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," *arXiv preprint arXiv:1512.00567,* 2015.

[16] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167,* 2015.