Thesis for the Degree of Master of Engineering

FoodReco: A Mobile App for Food and Nutrition Assessments using An Enhanced Densely Connected Convolutional Network

by

UWIZEYE Delphine

Department of IT Convergence and Application Engineering

The Graduate School

Pukyong National University

August 2021

FoodReco: A Mobile App for Food and Nutrition Assessments using An Enhanced Densely Connected Convolutional Network.

FoodReco: 강화된 DenseNet을 이용한 식품 및 영양 평가를 위한 모바일 앱

Advisor: Prof. Ha-Joo Song

by

UWIZEYE Delphine

A thesis submitted in partial fulfillment of the requirements

for the degree of

Master of Engineering

in Department of IT Convergence and Application Engineering,

The Graduate School,

Pukyong National University

August 2021

FoodReco: A Mobile App for Food and Nutrition Assessments using An Enhanced Densely Connected Convolutional Network.

A dissertation

by

UWIZEYE Delphine

Approved by:

Prof. Oh-Heum Kwon

(Chairman)

Prof. Young-Bong Kim

(Member)

Prof. Ha-Joo Song

(Member)

August 27, 2021

Table of Contents

List of Tables

List of figures

FoodReco: 강화된 DenseNet 을 이용한 식품 및 영양 평가를 위한 모바일 앱

UWIZEYE Delphine

부 경 대 학 교 대 학 원　　IT융합응용공학과

요 약

건강한 식단은 질병에 걸리지 않고 건강한 삶을 유지하기위한 매우 중요한 요소이다. 따라서 매일 먹는 식단을 분석하여 고른 영양 섭취량을 유지하는 것이 필요하다. 하지만 수작업을 통해 섭취한 음식과 그것을 구성하는 영양성분을 분석하고 관리하는 것은 매우 어려운 일이다. 이에 본 논문에서는 섭취한 식품과 그것의 영양성분을 자동으로 파악하고 분석하는 모바일 애플리케이션(App)을 제안한다.

제안하는 모바일 애플리케이션은 스마트폰의 카메라에 촬영된 사진에 대해 강화된 DenseNet 을 사용한 자동인식과정을 거쳐 식품의 종류를 알아낸다. 그리고 온라인 음식 성분 정보서비스인 EDAMAM API 를 사용하여 식품에 대한 영양정보를 조회하여 사용자에게 제공한다. 따라서 사용자는 자신의 영양 상황을 간단하게 파악하고 관리할 수 있다.

1. Introduction

Nutrition assessment is becoming an essential topic in researchers, as it is the key to maintaining healthy life through eating healthy food, managing body weight, and fight against some diseases like diabetes, obesity, etc. In nutrition assessment, it is crucial to recognize the food first for analyzing it later.

It is vitally important to provide the easiest way for humans to always being aware of the food they took with its nutrients to keep managing and monitoring their food intake. Recently it is estimated that more than 5 billion people have mobile devices, and over half of these connections are smartphones[1], which means mobile devices are the most usable tools by humans. Therefore, developing a mobile application for food and nutrition assessments is an unsurpassed solution for humans.

In recent years, deep learning has been recognized as the best technique over other machine learning techniques in many fields. Computer vision is one of the most fields that deep learning is in use. The exceptional performance and easiness in the training of deep learning techniques are the most reasons that make it more popular. One of the well-known use cases is image recognition (image classification), which flourished in other applications like food image recognition[2]. As a particular food may be prepared in countless different ways (similar foods may look different), and on the other hand, different foods may look similar, it is a challenging task to recognize some types of food. However, using deep

learning techniques for food recognition is the easiest and effective means than other techniques due to its efficacity in learning high-level features of an image that can differentiate it from others.

Deep learning techniques such as convolutional neural networks (CNN) require a huge number of images for training, while most available datasets for food recognition do not have a such needed number of images, and the collection of those data is complex and expensive. Therefore, the use of transfer learning approach[3] is the best way to get better results on insufficient dataset.

Mining helpful information about the food like calories, nutrients…, is a demanding task. Due to the availability of some trustable food databases, it became easy to extract the nutrition information of the food from those databases.

In this project, we propose an android-based mobile application called FoodReco for food recognition and nutrition assessment that users can access anywhere and everywhere.

We used an enhanced DenseNet model (a deep learning model) for food recognition and then integrated it into an android device. For nutrition assessment, we connected the App to the Edamam Nutrition Analysis API[4] for getting all the nutrition information about the recognized food. With the use of our mobile application, users will get the needed food information and that will help them to improve their health conditions.

The rest of the thesis is organized as follows: Section 2 explains some important topics related to the project and reviews some past related works;

section 3 describes the working principles of our app; section 4 presents the performance and the experimental results of the proposed system; finally, in section 5 we conclude and provide direction to the future work.

2. Literature Review

2.1. Deep learning

Deep learning is a machine learning technique inspired by our brain's functionality. It uses a neural network to discover the interdependencies in between the input's variables and the observed outputs.

Deep learning has been preferred than other machine learning techniques because of two main reasons:

- Its easiness in training: when using other machine learning techniques, input features are extracted manually while using deep learning, those features are extracted automatically[3].
- Its performance results: deep learning has been shown the best performance results over other machine learning techniques in many fields. Like in computer vision, deep learning algorithms are 41% more accurate than machine learning algorithms in image classification, 27% more accurate in facial recognition and 25% in voice recognition[5].

It is also preferred for newer areas of applications due to an increase in the computational power of processors in particular GPUs, its better performance on a huge amount of data, and its ability to learn high-level features from data[6].

Deep learning models are executed in two steps: feed-forward and backpropagation. In feed-forward step, it takes an input and generates an output for making prediction. In backpropagation step, it calculates the error between the predicted output and target output to measure the accuracy and then use it to update the weight for improving the accuracy(performance)[2].

Its architecture forms a hierarchical and powerful feature representation which makes it suitable for analyzing and extracting useful information from large amount of data and data collected from different sources[6].

Deep learning has many architectures like deep neural networks or artificial neural networks, deep belief networks, convolution neural networks (CNN) and recurrent neural networks. CNN is currently considered as one of the most popular machine intelligence models for big data analysis in various research areas[2].

2.2. Transfer learning

Training a deep convolutional neural network require a huge number of training data for achieving a high accuracy; most of the time to collect such amount of data is very expensive[7] and most of available dataset are insufficient. Transfer learning is an important tool in machine learning to solve the problem of insufficient training data. It transfers the knowledge,

or the features learned on a large dataset that has a similar domain to the insufficient training dataset[3]. It is taking a model that is already trained (pretrained model) with a large dataset and use the features learned from that dataset to train a new dataset. It is also known as fine-tuning a model.

In short, deep learning uses a cascade of multiple layers of nonlinear processing units for feature extraction and transformation. The lower layers close to the data input learn simple features (the most general features of an image), while higher layers learn more complex features derived from lower layer features[6]; therefore, it is reasonable to think that a network trained with a large dataset like ImageNet could be reused to achieve a specialization in a slightly different task.

In transfer learning, when you want to fine-tune a model, you remove the original classifier of a pretrained model and build your own, and then for other layers, you may:

- Freeze all the convolution base (using its weights without any change) or
- Train some layers (higher layers) and leave others frozen (lower layers).

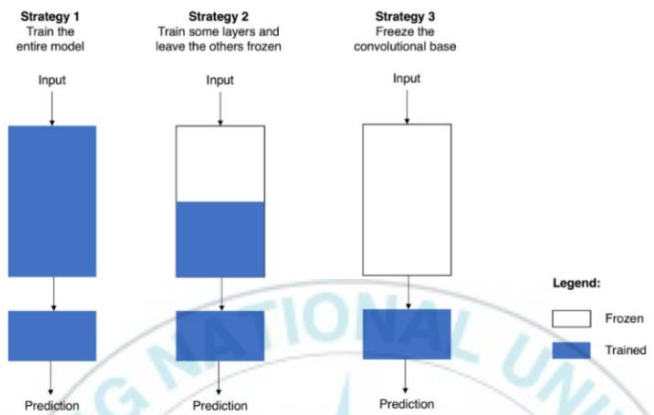


Figure 1. Training strategies of Deep learning models[8]

The above figure shows the summary of three training strategies of a deep learning model, where we may train it from scratch, fine-tune the pretrained model by either training some layers or freezing the entire convolutional base.

2.3. DenseNet

A Densely connected convolutional network, mostly called DenseNet, is a pre-trained model that connects each layer to every other layer in a feed-forward way. It is a CNN model that extends the residual learning

framework introduced by Resnet. Like Resnet[9], DenseNet has been proposed to solve the vanishing-gradient problem of the model as the model goes deeper. It also strengthens feature propagation, encourages feature reuse, and significantly reduces the number of parameters[10].

Traditional feed-forward architectures can be viewed as algorithms with a state, which is passed on from layer to layer. Each layer reads the state from its preceding layer and writes to the subsequent layer. It changes the state but also passes on information that needs to be preserved[10]. ResNets make this information preservation explicit through additive identity transformations. Even though Resnet made some improvement compared to traditional convolutional network, the number of parameters of ResNets is significantly larger because each layer has its own weights[10]. DenseNet requires considerably fewer number of parameters than Resnet to achieve similar accuracy.

There are 2 main differences between DenseNet and ResNet. Resnet uses only one feature-map of the preceding layer, while DenseNet uses the feature-maps of all preceding layers; Resnet also combine features through summation, while DenseNet concatenates all features from preceding layers. Concatenating feature maps learned by different layers increases variation in the input of subsequent layers and improves efficiency[10]; that is the reason DenseNet is more efficient.

DenseNet is mainly composed of dense blocks, transition layers (convolution layer and pooling layer), and a classification layer. It has

many architectures based on the numbers of layers they have: densenet-121, densenet-169, densenet-201 and densenet-264, the number on each name describes the number of layers that architecture has. Each DenseNet is composed of four dense blocks which are the most important part of a DenseNet that help to accomplish all the advantages of a DenseNet. Figure 2 illustrate different architectures of DenseNet.

Layers	Output Size	DenseNet-121	DenseNet-169	DenseNet-201	DenseNet-264
Convolution	112 × 112	7 × 7 conv, stride 2			
Pooling	56 × 56	3 × 3 max pool, stride 2			
Dense Block (1)	56 × 56	[1 × 1 conv; 3 × 3 conv] × 6	[1 × 1 conv; 3 × 3 conv] × 6	[1 × 1 conv; 3 × 3 conv] × 6	[1 × 1 conv; 3 × 3 conv] × 6
Transition Layer (1)	56 × 56	1 × 1 conv			
	28 × 28	2 × 2 average pool, stride 2			
Dense Block (2)	28 × 28	[1 × 1 conv; 3 × 3 conv] × 12	[1 × 1 conv; 3 × 3 conv] × 12	[1 × 1 conv; 3 × 3 conv] × 12	[1 × 1 conv; 3 × 3 conv] × 12
Transition Layer (2)	28 × 28	1 × 1 conv			
	14 × 14	2 × 2 average pool, stride 2			
Dense Block (3)	14 × 14	[1 × 1 conv; 3 × 3 conv] × 24	[1 × 1 conv; 3 × 3 conv] × 32	[1 × 1 conv; 3 × 3 conv] × 48	[1 × 1 conv; 3 × 3 conv] × 64
Transition Layer (3)	14 × 14	1 × 1 conv			
	7 × 7	2 × 2 average pool, stride 2			
Dense Block (4)	7 × 7	[1 × 1 conv; 3 × 3 conv] × 16	[1 × 1 conv; 3 × 3 conv] × 32	[1 × 1 conv; 3 × 3 conv] × 32	[1 × 1 conv; 3 × 3 conv] × 48
Classification Layer	1 × 1	7 × 7 global average pool			
		1000D fully-connected, softmax			

Figure 2. Different architectures of DenseNet[10]

Each ‘conv’ layer denotes a consecutive sequence of three operations: batch normalization (BN), rectified linear unit (RELU) and a 3x3 or 1x1 convolution(conv).

Dense block is designed in a way that each layer is directly connected to all subsequent layers and takes all the feature maps of all the preceding layers as input. Figure 3 shows the architecture of a dense block with 4 layers.

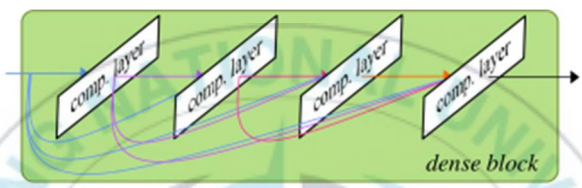


Figure 3. Dense block architecture with 4 layers[11]

That architecture of a dense block improves the flow of information between layers and avoid the vanishing-gradient problem that may be caused by using a very deep model.

Between every dense block, there is a transition layer used to reduce the number of feature maps before entering the next dense block.

DenseNet uses ReLU (rectified linear activation unit) that allows the neural network to skip over other layers of the model that may contain non-linearities. It is a non-linear activation function that acts as a linear function and allows the learning of complex data relationship.

2.4. Deep learning models for food recognition

During this section, we review some past works related to food recognition using deep learning techniques. Transfer learning has been used in most of the works reviewed due to the insufficient of dataset problem.

In [12], they have used a deep convolutional neural network (DCNN) for recognizing food, and they achieved an accuracy of 78.77% and 67.57% for UEC-FOOD100 and UEC-FOOD256, respectively. In[13], DeepFood (CNN-based model) was used for food recognition with bounding boxes (using cropped images) and without bounding boxes (using original images). With bounding boxes, they attained an accuracy of 63.8% and 77.2% for UEC-FOOD256 and UEC-FOOD100, respectively. Without bounding boxes, they attained an accuracy of 54.7% and 57.0% for UEC-FOOD256 and UEC-FOOD100, respectively. WISeR (Wide-Slice Residual Networks) technique in[14] achieved 90.27%, 89.58%, and 83.15% of accuracy on Food-101, UEC-FOOD100, and UEC-FOOD256, respectively using cropped images; and the results for using original images with ground truth were 79.46% and 72.71% on UEC-FOOD100, and UEC-FOOD256, respectively. In [15], DenseFood (based on DenseNet) showed 81.23% accuracy for the VIREO-172 dataset. In[16], researchers propose a concatenation of deep features from AlexNet and VGG16 models for food recognition. Their results showed an accuracy of 99.00%, 88.08% and 62.44% for FOOD-5K, FOOD-11 and FOOD-101

datasets, respectively. They have also used ResNet50 model for FOOD-101 and obtained the accuracy of 79.86%.

2.5. Nutrition-based mobile applications

Food recognition model is more useful when it is deployed in a mobile device for being accessible easily. The following reviews are based on food recognition mobile applications.

FoodTracker[17]:is a mobile application built based on food recognition and nutrition analysis. It uses a DCNN (deep convolutional neural network) based on Mobilenet for recognizing food. After that, they connected their app with Nutritionix API for getting the food item nutrition facts on the user interface of an app.

Nutritrack[18] : is an android-based food recognition app for nutrition awareness. It uses Clarifai API in android environment which utilizes machine learning algorithm in detecting various food that is widely available in the market. After detecting the food, the data is managed by Nutritionix API to return the nutrition estimation of the recognized food. Nutritrack uses cloud-based databases which are hosted by the android-based API to support the implementation of the project development.

MyNutriCart [19]: is a mobile application(app) developed to guide individuals for making smart and healthy choices when purchasing food in grocery stores. It provides a healthy grocery list based on the daily

nutritional recommendations of the individuals that constitute the participant's household. This list took into consideration a pre-defined budget, which was maximized by connecting to supermarkets' discounts. The specific objectives of the app are to improve food selection when purchasing foods in the grocery stores based on a pre-defined budget, to improve dietary patterns based on the Dietary Guidelines for Americans, and to improve weight status.

OncoFood [20]: is a mobile phone app aimed to record and optimize the dietary behavior in oncologic patients. The app records the daily intake of the patients and compare them with the nutrients targets to help patients to reach their nutritional goals. They created two groups of patients, one with app, others without an app and both with nutrition counseling and nutritional therapy to measure the efficacity of the app. They have found that patients who track their daily dietary habits using the app are more likely to reach their nutritional goals than others without the app.

3. Design and implementation of FoodReco

3.1. Enhanced DenseNet

We have built a deep learning model (enhanced DenseNet) by enhancing a densely connected convolutional network (DenseNet) for food image recognition.

As we focused on building a model with highest efficiency for helping People to manage their food intakes, we have built a model based on DenseNet due to its efficiency compared to other CNN models.

As DenseNet has been trained on ImageNet dataset, which is different from our dataset, we chose to retrain again some upper layers for helping the model to adapt on our dataset.

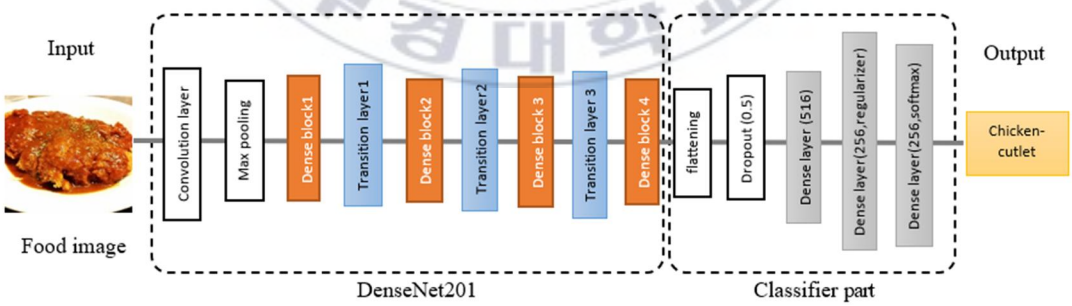
We have used Densenet-201(with 201 layers) as a base model where we froze 120 first layers and trained the remained upper layers to extract the key features of an image during training. We also modified the classifier part of DenseNet201 by creating our classifier to improve the performance of our model.

When training a dense neural network, it tends to memorize the training data and fails for new data; that is called overfitting. To avoid that, we have used dropout which is the most technique used in deep learning to avoid overfitting.

Because we used insufficient dataset, we have used a technique called data augmentation which increases the diversity of the training set by applying different types of transformation on training dataset. Not only It increases the amount of the training set to prevent overfitting, but also it increases the capacity of a model to correctly recognize an image, no matter the angle or the position an image has been took.

For the classifier part, we used three dense layers to improve the classification performance of the model, as many dense layers help in decreasing errors during classification. We also added regularizers for improving the classification results on the new data. Figure 4 shows the architecture of Enhanced DenseNet model we proposed.

Figure 4. Enhanced DenseNet model structure



3.2. FoodReco (an android App)

After training and testing the model, we have deployed it into an android application for helping the users to access it easily and anywhere.

For deploying a deep learning model, we have used TensorFlow lite as an intermediary between a model and android studio features.

3.2.1. TensorFlow Lite

For training a deep learning or simply a machine learning model, some platforms must be used. Among those platforms there is one called TensorFlow[21]. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that help users to easily build and deploy machine learning models' applications. Models used TensorFlow are called TensorFlow models. Keras[22] is a high-level library written in Python that can be used to train a machine learning model. As it is a high-level library, it always needs a backend for running. Among the possible Keras backends, there is TensorFlow which is even a default backend for Keras. When a Keras model has used TensorFlow backend is also a TensorFlow model.

As TensorFlow and Android are both developed by Google, TensorFlow models can be integrated into android software environment easily by adding TensorFlow lite library as a dependency[23]. TensorFlow Lite[24] is a set of tools used to run TensorFlow models in mobile devices or simply IoT devices. It has 2 main components:

- TensorFlow lite interpreter that runs the model in the device.
- TensorFlow lite converter that converts the model in an efficient form for use by interpreter.

Once the trained model with its weights was saved, we have used TensorFlow lite converter for converting the model in a format that TensorFlow lite interpreter is able to understand(.tflite) and then we put the model with food names(labels) in android studio to build an application. By recognizing the food, the App returns the top three predicted food names of a given food image with its confidence.

3.2.2. Nutrition analysis

After recognition of the food, further analysis of the food is needed for getting the necessary information about it. We have used Edamam Nutrition Analysis API for extracting all the necessary information about the recognized food.

Edamam API have been made available and provided by Edamam LLC (“Edamam”) through Edamam ’s web site to provide users the opportunity to discover nutritional information for recipes and other food. It covers all key use cases related to recipe and food text natural language processing and nutrition analysis. It uses NLP (Natural Language Processing) for extraction of food entities from unstructured text. We have used Food text analysis part as we want the API to return the nutritional analysis of the food text we entered as input. For submitting food text, a user must include

the quantity of the food to get nutrition information, otherwise the API will return only a food match. Through the API, we get health label information like kidney friendliness, alcohol-free, egg-free, etc. The API also gives information about diet labels regarding high-fiber and low-fat, and total nutrients amount: calories, cholesterol, carbs, and so on. The recognized food name with the highest confidence is sent directly as a food text input for the API to return the nutrition information about it. The predicted name of the food image with its nutrition information will displayed on our App interfaces. Figure below briefly illustrate the design of our mobile App.

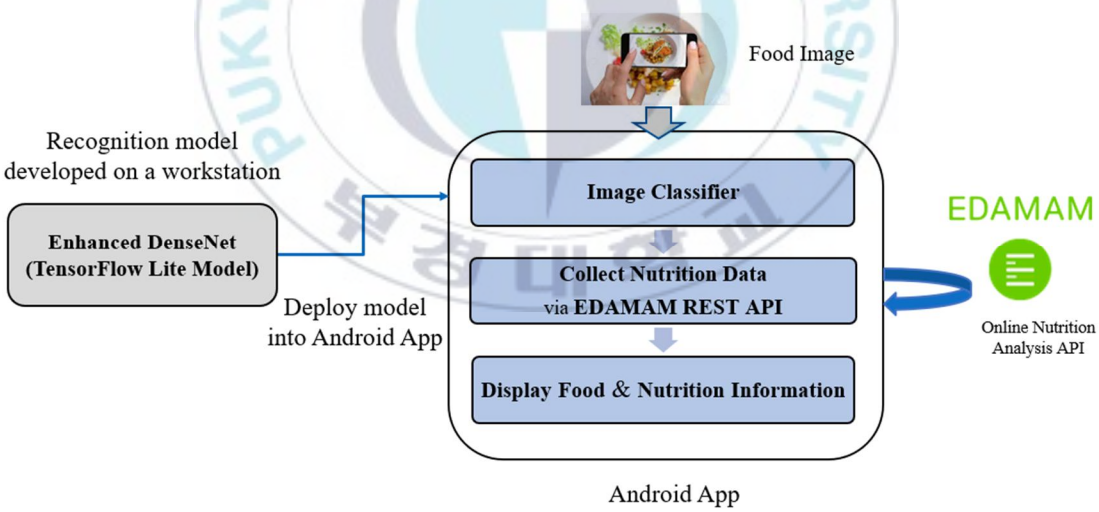


Figure 5. Design of FoodReco App

4. Experimental results

The experiment was run on a desktop of AMD Ryzen 7 3700X 8-Core Processor 3.59 GHz, 32GB RAM. We have used Python language to train our model on google colaboratory using our local host with GPU. We used JAVA language for development of our App. It is an object-oriented programming language and the primary language used for Android application development.

4.1. Dataset

We have used UEC-FOOD256[25] dataset, which is freely available online. It is composed by 256 different types of food, each type contains at least 110 images, and each image has a bounding box indicating the position of the food item in the photo. The dataset is composed of 31,395 colored food images. The most categories in that dataset are from Japan, they may not be familiar with other people than Japanese or simply Asian people. The nature of that dataset is complex as each image has many types of food. For better results, it needs to be cropped so that the model learns only the features of a true food image type. Most of categories look also very similar which make it more complex as they cannot be recognized simply by eye.

Deep learning model requires a huge number of images for efficient performance when training it from scratch; unfortunately, this dataset contains insufficient images per type. For getting good results on that dataset, better is to use pre-trained model and then fine-tune it on the dataset. We adopted DenseNet model as a pre-trained model for the purpose.

4.2. Performance evaluation

In our experiments, we have used UEC-FOOD256 dataset in two ways: with the background (uncropped images) and without any background (cropped images). For model validation, we split the dataset into two sets (training set and validation set by 60% and 40% correspondingly).

We have used fixed Image size of (300,300,3) in all experiments, and for data augmentation we used width-shift-range=0.3, zoom-range=0.3, height-shift-range=0.2 and rotation-range=30. For regularization, we used 0.5 value of dropout parameter and 0.01parameter's value for L2 regularization.

Training and validating our model were based on Keras API with a TensorFlow backend. We have used overall accuracy as a measure of the performance of the proposed methods. We used the following metrics for accuracy comparison:

- Top-1 accuracy that shows the performance of the model by considering only the first prediction class of an image having a high probability.
- Top-5 accuracy which shows the first five prediction classes of an image having high probabilities. We have used 40 epochs in all experiments.

For building our app we have used android studio version 4.1.1 and 6.5 Gradle version. We used TensorFlow lite android support library for making easier to integrate our TensorFlow model in android device. It also provides high-level APIs that help transform raw input data into the form required by the model, and interpret the model's output, reducing the amount of boilerplate code required. we have used sound cloud API library for cropping image activity. For networking Edamam API with an android app, we have used Volley library which is easy and very fast. It also has many other advantages like providing debugging and tracing tools, automatic scheduling of network requests, prioritizing requests, … We have also added some permissions like the one for using internet for accessing the Edamam API, one for using a camera of a device, one for saving the picture we captured during classification, ...

4.3. Results

We performed many experiments, but we saved the models that achieved the most interesting results for comparison and getting some knowledge

from them. We used our proposed model with freezing all layers of densenet201 and with training some layers and freeze others (our proposed model). As a result, Table 1 shows that training some upper layers improved the performance as the model can extract all the important features from an image during training.

Table 1. Performance comparison of freezing all layers of Densenet201 model and freezing only 120 layers.

Proposed Method	Training accuracy (top 1)	Validation accuracy (top1)
all layers frozen	0.8607	0.6035
only 120 layers frozen	0.9223	0.6530

We have also trained our proposed model on dataset with uncropped images (many food types appeared in one image). The results in Table 2 showed that even though the images are complex to learn key features, our model got the best results.

Table 2. Performance comparison of Resnet152 model and the proposed model for uncropped images.

Method	Top1 acc(train)	Top1 acc (Val)	Top5 acc(train)	Top5 acc (Val)
Resnet152	0.6984	0.4751	0.9258	0.7441
Proposed	0.8830	0.5693	0.9895	0.8060

Then after, we used dataset with cropped images (in each image, it appears only one food type); This experiment was the most exciting one as is the one that gave us an idea on how our method is effective compared to others. That option is the one we used to build our mobile App.

Table 3. Performance comparison of Resnet152 model and the proposed model for cropped images.

Method	Top1 acc (train)	Top1 acc (Val)	Top5 acc (train)	Top5 acc (Val)
Resnet152	0.8657	0.5681	0.9817	0.8226
Proposed	0.9223	0.6530	0.9987	0.8665

The following graph shows the performance of our proposed model on training and validation sets of cropped images as we increase the number of epochs. It shows that the accuracy increases as the number of epochs increases.

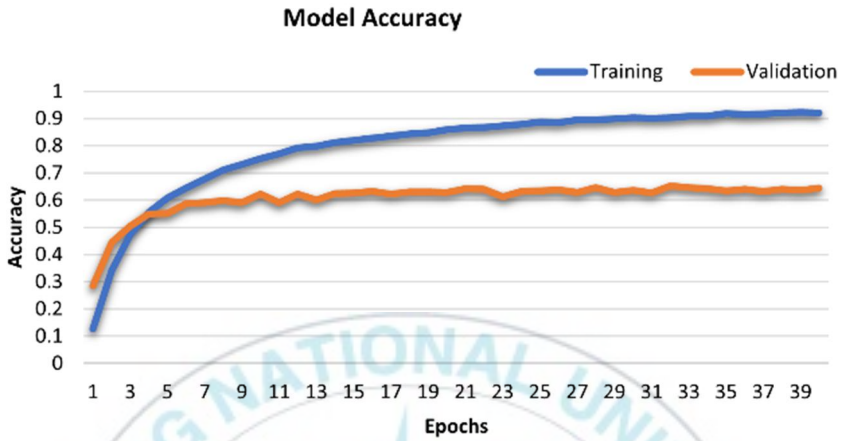
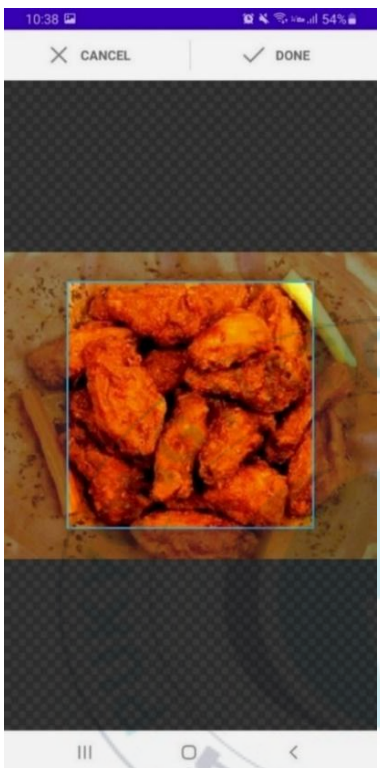


Figure 6. Performance of our model versus epochs number

After saving the model, we used it to build an android App for food recognition. We also used Edamam API to collect the nutrition information of the recognized food. The following figures are the screenshots of the FoodReco mobile application:

Figure 7. Launching the App **Figure 8. Browsing a food image**

After launching the application, the user will be asked to take a photo either by a camera or from gallery. We have used two options for making our application more flexible depending on the user's wish. Figure 8 shows the user's gallery after choosing gallery as a source of a picture.



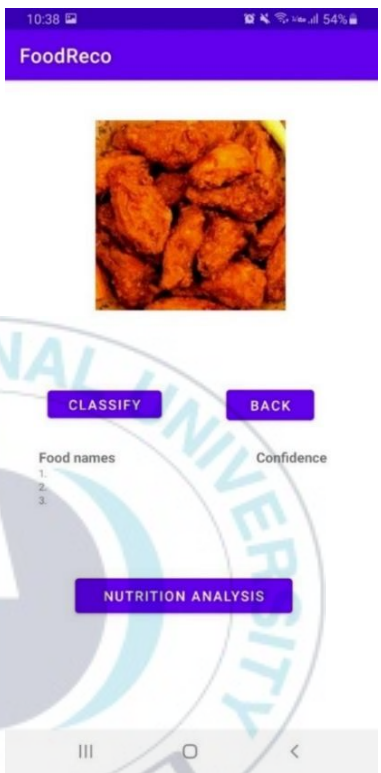


Figure 9. Cropping image

Figure 10. Classify cropped image.

After selecting a food photo, a user is asked to crop it for removing any background if it exists. Figure 9 shows a selected image being cropped. A cropped picture will be shown on the previous interface as shown on figure 10,and a user may classify the image or go back to retake a photo in a case it is needed.

Figure 11. Showing predicted names of an image

After classify the picture, the app will return the top three predicted food names of it with its confidence as shown on Figure 11. Then, if a user wish to know the nutrition information of the recognized food, he/she will continue with ‘nutrition analysis’.

Figure 12. Copy the selected name Figure 13. Health information

When a user selected to get nutrition information of the food, the food name with the highest confidence will be copied directly to the following interface as shown on figure 12.we used an edit text option here to allow any change that a user may want to make on the input text like changing the food name whenever necessary or changing the amount of the food. On that interface, a user may select any type of information he/she wants to get; either diet info, health info or nutrients.The nutrition information will be displayed on the interface as shown on figures: 13 for health information, 14 for diet information and 15 for nutrients information.

Figure 14. Diet information

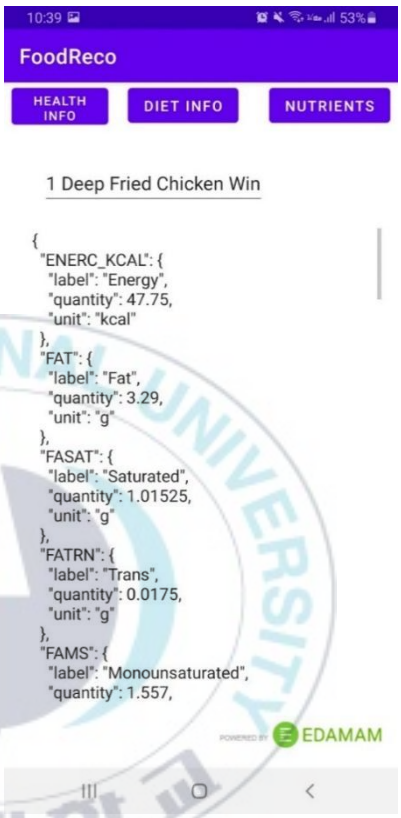


Figure 15. Nutrients information.

Getting nutrition information from nutrition analysis API will be possible only if a user's device has internet, otherwise there will be an error, as the API is accessible online.

5. Conclusion

This study aimed to develop a successful mobile app that can help people to keep monitoring and controlling their dietary behavior using food images. We built the app using an Enhanced DenseNet model and Edamam nutrition analysis API. The application implements food recognition and nutrition analysis tasks shortly and efficiently with more reliable and accurate outcomes. It is more flexible with high capacity as it gives many options to choose to the user wherever needed.

An enhanced DenseNet model used in the app showed the best performance compared to the existing approaches, where it can recognize a new image at an accuracy of 65.3% which is the best performance on that complex dataset we used. Edamam API also showed that it delivers all the necessary and well-arranged information about the food.

As a result, the proposed mobile app shows that it is a convenient tool for people to maintain their lives healthy. In the future, we wish to expand the functionality of our App by creating a database inside to record all food intakes of the user with its nutrition facts so that a user may get his/her historical food intake easily wherever needed. We also wish to add weight goal setter and tracker to help users to set and reach their weight goals for ameliorating their health conditions.

Appendix

A 1 Notations

Notation	Description
API	Application Programming Interface
App	Application
DenseNet	Densely Connected Convolutional Network
CNN	Convolutional Neural Network
Resnet	Residual Network
Acc	Accuracy
Train	Training
Val	Validation
GPUs	Graphics Processing Units

References

[1] R. Paramastri *et al.*, "The use of mobile applicatiosn to improve nutrition behaviour: A systematic review," *Comput. Methods Programs Biomed.*, vol. 192, p. 105459, 2020, doi: 10.1016/j.cmpb.2020.105459.

[2] L. Zhou, C. Zhang, F. Liu, Z. Qiu, and Y. He, "Application of Deep Learning in Food: A Review," *Compr. Rev. Food Sci. Food Saf.*, vol. 18, no. 6, pp. 1793–1811, 2019, doi: 10.1111/1541-4337.12492.

[3] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, "A survey on deep transfer learning," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 11141 LNCS, pp. 270–279, 2018, doi: 10.1007/978-3-030-01424-7_27.

[4] "Nutrition Analysis API." https://developer.edamam.com/edamam-nutrition-api (accessed Mar. 12, 2021).

[5] "Deep Learning Tutorial: Neural Network Basics for Beginners." https://www.guru99.com/deep-learning-tutorial.html (accessed Feb. 26, 2021).

[6] P. P. Shinde and S. Shah, “A Review of Machine Learning and Deep Learning Applications,” *Proc. - 2018 4th Int. Conf. Comput. Commun. Control Autom. ICCUBEA 2018*, 2018, doi: 10.1109/ICCUBEA.2018.8697857.

[7] K. Wang, X. Gao, Y. Zhao, X. Li, D. Dou, and C.-Z. Xu, “Pay Attention to Features, Transfer Learn Faster CNNs,” *Iclr*, pp. 1–14, 2019.

[8] “Transfer learning from pre-trained models | by Pedro Marcelino | Towards Data Science.” https://towardsdatascience.com/transfer-learning-from-pre-trained-models-f2393f124751 (accessed Jun. 21, 2021).

[9] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2016-Decem, pp. 770–778, 2016, doi: 10.1109/CVPR.2016.90.

[10] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, Nov. 2017, vol. 2017-January, pp. 2261–2269, doi: 10.1109/CVPR.2017.243.

[11] N. Takahashi and Y. Mitsufuji, “Multi-Scale multi-band

densenets for audio source separation," *IEEE Work. Appl. Signal Process. to Audio Acoust.*, vol. 2017-Octob, pp. 21–25, 2017, doi: 10.1109/WASPAA.2017.8169987.

[12] K. Yanai and Y. Kawano, "FOOD IMAGE RECOGNITION USING DEEP CONVOLUTIONAL NETWORK WITH PRE-TRAINING AND FINE-TUNING Keiji Yanai Yoshiyuki Kawano Department of Informatics , The University of Electro-Communications , Tokyo , Japan," *Int. Conf. Multimed. Expo Work. . IEEE*, pp. 1–6, 2014.

[13] C. Liu, Y. Cao, Y. Luo, G. Chen, V. Vokkarane, and Y. Ma, "DeepFood: Deep Learning-Based Food Image Recognition for Computer-Aided Dietary Assessment," *LNCS*, vol. 9677, pp. 37–48, 2016, doi: 10.1007/978-3-319-39601-9_4.

[14] N. Martinel, G. L. Foresti, and C. Micheloni, "Wide-slice residual networks for food recognition," *Proc. - 2018 IEEE Winter Conf. Appl. Comput. Vision, WACV 2018*, vol. 2018-Janua, pp. 567–576, 2018, doi: 10.1109/WACV.2018.00068.

[15] A. S. Metwalli, W. Shen, and C. Q. Wu, "Food Image Recognition Based on Densely Connected Convolutional Neural Networks," *2020 Int. Conf. Artif. Intell. Inf.*

Commun. ICAIIC 2020, pp. 027–032, 2020, doi: 10.1109/ICAIIC48513.2020.9065281.

[16] A. Sengur, Y. Akbulut, and U. Budak, “Food Image Classification with Deep Features,” *2019 Int. Conf. Artif. Intell. Data Process. Symp. IDAP 2019*, no. September, 2019, doi: 10.1109/IDAP.2019.8875946.

[17] J. Sun, K. Radecka, and Z. Zilic, “FoodTracker: A real-time food detection mobile application by deep convolutional neural networks,” *arXiv*, no. September, 2019.

[18] A. B. Ocay, J. M. Fernandez, and T. D. Palaoag, “NutriTrack: Android-based food recognition app for nutrition awareness,” *2017 3rd IEEE Int. Conf. Comput. Commun. ICCC 2017*, vol. 2018-Janua, pp. 2099–2104, 2018, doi: 10.1109/CompComm.2017.8322907.

[19] D. López *et al.*, “Development and evaluation of a nutritional smartphone application for making smart and healthy choices in grocery shopping,” *Healthc. Inform. Res.*, vol. 23, no. 1, pp. 16–24, 2017, doi: 10.4258/hir.2017.23.1.16.

[20] T. Orlemann *et al.*, “A Novel Mobile Phone App (OncoFood) to Record and Optimize the Dietary Behavior of Oncologic Patients: Pilot Study,” *JMIR Cancer*, vol. 4,

no. 2, p. e10703, 2018, doi: 10.2196/10703.

[21] M. Abadi *et al.*, “TensorFlow: A system for large-scale machine learning,” *Proc. 12th USENIX Symp. Oper. Syst. Des. Implementation, OSDI 2016*, pp. 265–283, 2016.

[22] “Keras: the Python deep learning API.” https://keras.io/ (accessed May 10, 2021).

[23] A. Sehgal and N. Kehtarnavaz, “Guidelines and benchmarks for deployment of deep learning models on smartphones as real-time apps,” *arXiv*, pp. 450–465, 2019, doi: 10.3390/make1010027.

[24] “TensorFlow Lite | ML for Mobile and Edge Devices.” https://www.tensorflow.org/lite/ (accessed Apr. 02, 2021).

[25] Y. Kawano and K. Yanai, “Automatic Expansion of a Food Image Dataset Leveraging Existing Categories with Domain Adaptation,” *ECCV Work.*, 2014.

Acknowledgements

Finally, I am about to finish my master's studies at Pukyong National University. It was a long journey, full of difficulties and obstacles. I heartedly thank the **Almighty God** for the gift of life and spirit of hard work that he has always stowed in me especially during this work.

This thesis would not have been possible without the guidance and the help of several People who in one way or another contributed and extended their valuable assistance in the preparation and completion of this study.

First and foremost, I am extremely grateful to my advisor, **Prof. Ha-Joo Song** for his invaluable advice, continuous support, and patience during my study. His immense knowledge and plentiful experience have encouraged me in all the time of my academic research and daily life. I will forever be thankful to him.

I would like to express my sincere gratitude to **SeAh Networks members** who gave me the scholarship opportunity and fully supported me during my studies. Their support and guidance will mark me forever.

My sincere appreciations to my **IDB lab mates** for their partnership and friendship. It is their kind help and support that made my studies and life in South Korea less tough. I cherish every moment we spent together.

I would like to express my sincere appreciation to **my Parents and Sisters** for their unconditional love, care, and encouragement. They

contributed a lot along this journey, I would not have made it this far without them. My profound gratitude to my husband **NUMUGISHA J. de Dieu**. He sacrificed a lot to support me achieving my goals. Whenever I got stuck or get discouraged, he was always there to brace me for keep going. Without his love, his encouragement, his trust in me and his patience, this work would never be completed. I am also grateful to **all my family and friends**; they were all my strongest supporters. I love you dearly!

There are no such meaningful words to express my deepest gratitude to you, may the Almighty God bless you.

Thank you so much!