

Doctoral Dissertation

Development of Hand Posture Classification  
and Food Constituent Estimation  
as Welfare Technology  
Using Convolutional Neural Network

March, 2020

Sulfayanti F. Situju

Graduate School of Advanced Systems Engineering

Okayama Prefectural University

## Acknowledgement

The work presented in this dissertation has been carried out at Okayama Prefectural University under the mentorship of Professor Akihiro Kanagawa.

I would first like to express my gratitude to Professor Akihiro Kanagawa for continuous encouragement and careful guidance throughout the research. His kindness and support made it possible for me to concentrate completely on my studies without having to worry about anything unrelated.

I also wish to express my sincerest appreciation to Associate Professor Hitoshi Yamauchi of Okayama Prefectural University for his supervision and support. I have greatly benefited from his advice to complete this research.

I would like to express my deepest gratitude to Associate Professor Hironori Takimoto of Okayama Prefectural University. As one of my advisors, his advice and guidance have been invaluable to me throughout this work, and his valuable comments provided great help towards the accomplishment of this dissertation. He also provided me with the opportunity to acquire much essential knowledge and skills for a person who aspires to become a researcher.

My gratitude to Professor Armin Lawi of Hasanuddin University, whom I learned from in the research collaboration.

My thanks are also owed to a member of Mathematical and Information Media Engineering Laboratory, Okayama Prefectural University, who has provided significant assistance not only for research and academic matters, but also for the social life in Japan.

I am also immensely thankful to Okayama Prefectural University for giving me the chance to study there. I am grateful to have received a lot of support from this college so that I could comfortably perform the research.

Lastly, I want to thank my parents and my family for support and prayers that always accompany me.

# Contents

<b>1</b>	<b>Introduction</b>	<b>8</b>
1.1	Welfare Technology . . . . .	8
1.2	Purpose of Dissertation . . . . .	14
<b>2</b>	<b>Deep Learning</b>	<b>17</b>
2.1	Overview of Deep Learning . . . . .	17
2.2	Convolutional Neural Network . . . . .	22
2.3	Preprocessing . . . . .	31
2.4	Transfer Learning CNN . . . . .	33
<b>3</b>	<b>Hand Posture Classification</b>	<b>34</b>
3.1	Background . . . . .	34
3.2	Related Works . . . . .	37
3.3	Proposed Method . . . . .	40
3.3.1	Hand Segmentation . . . . .	41
3.3.2	Data Augmentation . . . . .	42
3.3.3	Hand Posture Classification . . . . .	47
3.4	Experimental Setup and Results . . . . .	50
3.4.1	Experimental Setup . . . . .	50
3.4.2	Results and Discussions . . . . .	52
3.5	Conclusions . . . . .	55

<i>CONTENTS</i>	3
<b>4 Food Constituent Estimation</b>	<b>58</b>
4.1 Background . . . . .	58
4.2 Related Works . . . . .	61
4.3 Proposed Method . . . . .	66
4.3.1 Dataset Construction . . . . .	67
4.3.2 Architecture of Multi-task CNN . . . . .	68
4.3.3 Two-stage Fine-tuning . . . . .	73
4.4 Experimental Setup and Results . . . . .	75
4.4.1 Experimental Setup . . . . .	75
4.4.2 Results and Discussions . . . . .	76
4.5 Conclusions . . . . .	80
<b>5 Conclusions</b>	<b>83</b>

# List of Figures

1.1	General handcrafted feature approach scheme . . . . .	11
2.1	Linear process on fully connected layer inspired by human biological neuron [81] . . . . .	19
2.2	Convolution process by the filter set to the input [100] . . . . .	23
2.3	Pooling layers process in convolutional network [101] . . . . .	26
2.4	Activation function and logistic regression operation at the end of convolutional network in image classification . . . . .	28
2.5	Illustration of convolutional neural network architecture [102] . . . . .	28
2.6	Depthwise separable convolution on Xception . . . . .	30
2.7	Differences between Inception and Xception [107] . . . . .	31
2.8	Pixel values in an image [108] . . . . .	31
2.9	Three channels of an RGB color image . . . . .	32
3.1	Hand posture variation due to viewpoint difference and subject difference [115] . . . . .	36
3.2	User wearing a wristband [119] . . . . .	38
3.3	Image segmentation result [115] . . . . .	38
3.4	Overview of the proposed hand posture classification on training . . . . .	40
3.5	Examples signs with depth images of Kang's dataset . . . . .	41
3.6	The result of hand region segmentation method on raw data . . . . .	42
3.7	Possible transformation imposed on the image of sign 'A' . . . . .	43

3.8	Direction of hand rotation . . . . .	45
3.9	Examples of rotation results. (a) Original image, (b) result rotated by $-15^\circ$ in the pitch direction, (c) result rotated by $-15^\circ$ in the yaw direction, and (d) result rotated by $-15^\circ$ in the roll direction . . . . .	46
3.10	Example of hand augmentation with/without specifying the thickness. (a) Original image, (b) augmented depth images without specifying the thickness, and (c) augmented depth image with specification of the thickness . . . . .	47
3.11	Architecture of the Xception model . . . . .	49
3.12	Comparison of confusion matrices . . . . .	56
3.13	Appearance of the ‘3’ sign posture in the dataset . . . . .	57
3.14	Appearance of ‘D’ and ‘1’ sign postures in the dataset . . . . .	57
4.1	Four feature pairwise. (a) Information table of pairwise feature, (b) (c) (d) (e) are illustrations of the pairwise feature [142] . . . . .	62
4.2	Food detection and food-balance estimation method using the global feature [146] . . . . .	63
4.3	Four different architectures of deep CNN for multi-task learning of food category and ingredient recognition [158] . . . . .	65
4.4	Multi-task CNN based on VGG-16 architecture . . . . .	65
4.5	Architecture of proposed multi-task CNN . . . . .	66
4.6	Representations of food images for each collected category . . . . .	68
4.7	Distribution of calorie content throughout the dataset . . . . .	70
4.8	Distribution of salinity content across the dataset . . . . .	71
4.9	Xception architecture in multi-task CNN . . . . .	72
4.10	Hard parameter sharing and soft parameter sharing structures on multi-task learning [172] . . . . .	73
4.11	Two-stage fine-tuning using Xception on collected food images dataset .	74

4.12 Confusion matrix from single-task and multi-task on food category classification . . . . .	78
4.13 Successful and failed cases of calorie estimation results . . . . .	80
4.14 Successful and failed cases of salinity estimation results . . . . .	80

# List of Tables

1.1	Examples of assistive and monitoring technology implementation . . . . .	10
3.1	Comparison of classification accuracy of some CNNs on ImageNet dataset	48
3.2	Comparison of size and training speed between Inception and Xception	49
3.3	Details of datasets for experiments . . . . .	52
3.4	Results of the proposed method for classification of ASL dataset . . . . .	53
3.5	Comparison of results in the classification accuracy based on the method	55
4.1	Details of collected ingredient-annotated food images for multi-task CNN	69
4.2	Details of collected category-annotated food images for two-stage fine-tuning . . . . .	70
4.3	Detailed setup for the experiments of the tasks . . . . .	76
4.4	Comparison of results of each task . . . . .	77
4.5	Comparison of calories estimation on each class . . . . .	81
4.6	Comparison of salinity estimation on each class . . . . .	82



# Chapter 1

## Introduction

### 1.1 Welfare Technology

The number of aging people is increasing across the world. This growing population of aging people, who need health and care services is not equivalent with the number of young people entering the workforce, which is decreasing. This problem is apparent in developed countries such as Japan, Europe, and Northern America [1]. Human resources will be faced with more difficulty to provide assistance to people in need if they merely depend on the same services and technologies. Another similar situation is the caring for people with a sickness or disability, who likewise require help from human resources [1, 2]. Moreover, most of these individuals prefer to stay at home instead of entering a healthcare center. This leads to concerns of how to provide the services, such that this society can maintain health and well-being independently at their place of residence [3, 4]. This concern does not exclude the possibility of services to citizens outside this society as well. The availability of facilities capable of supporting society's activities is able to improve their life quality.

Welfare technology is used to enhance human welfare in daily life, especially for the welfare society using services. Welfare technology is provided as a solution to fulfill the needs of the welfare society, and it is expected to support human activity

making people more independent and obtaining a higher quality of life, with regard to healthcare, homecare, and social activity [5, 6]. Some examples of welfare technology functions are helping to remedy disability and provide independent service, create the possibility for people with the disease lives in their own home longer under better conditions, and present efficient service to be used by staff/family in caring and assisting citizens in providing a better quality of life by providing these services. Welfare technology can be realized in each home and through a mobile device, such that it can reduce the dependence on other human resources such as citizen service centers, schools, or hospitals [4, 7].

Welfare technology mainly offers two technology services, namely assistive technology and monitoring technology. Assistive technology is a system or device used to support tasks that are difficult to achieve for people and maximize the independence of disabled people by physical, sensory and cognitive training. The examples of assistive technology are smart home [8, 9, 10], robot [11, 12, 13, 14, 15, 16], rehabilitation software [17, 18], etc. A smart home is designed to provide comfort for residents by automatically controlling the equipment. Robots are made to help humans in carrying out one or more activities. Rehabilitation software can recently be developed with a virtual reality game that uses motion as input [17, 19]. This makes the process of therapy more pleasant for patients [20]. Rehabilitation software also allows the therapy process to be organized at home under the monitoring of doctor [18]. Monitoring technology involves equipment used inside or outside the home to monitor circumstances and provide alarms or information to the caregiver if an unusual condition occurs [21]. The roles of monitoring technology can be set in the system [22] such as medicine reminder [23, 24, 25], health/chronic care system [26, 27, 28, 29, 30], emergency alarm [31, 32, 33], location monitoring system [31, 34, 35], etc. More examples of assistive and monitoring technology implementation are provided in Table 1.1.

The development of a wide variety of sensors is accelerating the realization of more sophisticated welfare technology. Various types of sensors are widely used to

Table 1.1: Examples of assistive and monitoring technology implementation

Welfare technology	Implementation	Examples
Assistive technology	Smart home	<ul style="list-style-type: none"> <li>- Automatic controlling :</li> <li>· Lighting [36, 37]</li> <li>· Climate [38]</li> <li>· Entertainment [39]</li> <li>· Smart equipment [40, 41]</li> <li>· Security/alarm [10, 37, 42]</li> </ul>
	Robot	<ul style="list-style-type: none"> <li>- Feeding robot [11, 12, 13, 14]</li> <li>- Handy 1 [15] (support daily tasks: eating, drinking, washing, shaving, teeth cleaning, and applying make-up)</li> <li>- Vacuum cleaner robot [16]</li> </ul>
	Rehabilitation software	<ul style="list-style-type: none"> <li>- Virtual reality game for rehabilitation [17, 18, 19]</li> </ul>
Monitoring technology	Medicine reminder	<ul style="list-style-type: none"> <li>- Android-based application [23]</li> <li>- Internet of Things (IoT) based [24]</li> <li>- Automatic pill dispenser [25]</li> </ul>
	Health/chronic care system	<ul style="list-style-type: none"> <li>- Electrochemical glucose biosensors [26]</li> <li>- Mobile health monitoring [27, 29, 30]</li> <li>- Dietary mobile application [28]</li> </ul>
	Emergency alarm	<ul style="list-style-type: none"> <li>- Mobile device and application [31]</li> <li>- Sensor network [32, 33]</li> </ul>
	Location monitoring system	<ul style="list-style-type: none"> <li>- Device and application [31, 34]</li> <li>- RFID mat sensor system based [35]</li> </ul>

realize the assistive system for humans. Sensors, GPS, and cameras are the hardwares frequently used to develop welfare technology [2]. The utilization of these devices is occasionally combined, as in the sensor and camera, sensor and GPS, or camera and GPS. The camera can contribute to monitoring using a mobile phone application. This camera capability is owing to the growth of increasingly sophisticated camera performance and embedment in almost all mobile devices. The camera is employed to obtain the input images, which in some health applications are used to determine/monitor the patient's health conditions and within the smart home project to find human activities in the house. The capability is sustained by the development of

image processing techniques in computer vision [43, 44].

Most of the welfare technologies with camera devices are realized based on various computer vision techniques. Computer vision offers a digital solution by analyzing the captured image or video to generate interesting information. As applied technology is closely related to human life, Zhu *et al.* proposed a prototype system utilizing the mobile phone camera to measure nutrient intake from daily food [45]. As described above, computer vision contributes considerably to the realization of sign language recognition and food constituent estimation, which are representative of the support system for human communication and healthcare in welfare technology.

However, in the research field of computer vision that uses only RGB images captured by a visible light camera, it is difficult to execute tasks stably under various conditions or different environments such as lighting, weather, and background. Robustness with respect to these conditions can be achieved by integrating with data obtained from the external sensor [46]. Multi-modal sensors (depth and thermal sensors) and the computer vision approach are leveraged in a system to detect and monitor aging people's activity continuously. The system was developed for direct monitoring and analyzing patterns of aging people's activities in daily life to improve the caregiver's ability to assist [44]. Other research developed computer vision-based games for physical exercise [47]. This game was built using some gesture recognition algorithms and the Kinect sensor, making it possible to deal with motion commands.

Many of the recognition tasks in computer vision are conventionally solved using the handcrafted feature-based approach. The experts specifically design the approach for feature detectors and descriptors, and subsequently the classification task is usually

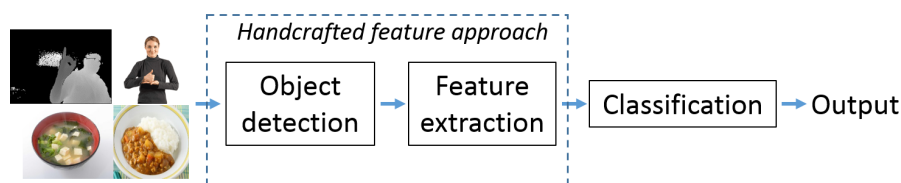


Figure 1.1: General handcrafted feature approach scheme

followed by trained classifiers [48]. Figure 1.1 shows the example of a general scheme using the handcrafted feature approach for object recognition.

I describe representative handcrafted feature-based approaches for sign language recognition and the food constituent estimation. As a typical handcrafted feature-based approach for gesture recognition, Kuznetsova *et al.* proposed a fingerspelling recognition of static gesture using a consumer depth camera based on handcrafted features [49]. The study applied the ensemble of shape function (ESF) descriptors for feature extraction and multi-layer random forests (MLRFs) for classification. Research in Ref. [50] leverages RGB and depth images from the Kinect sensor to make a fingerspelling recognition framework. The study was performed by utilizing a depth map in hand segmentation, using the kernel descriptor and scale-invariant feature transform (SIFT) for hand feature extraction and the support vector machine (SVM) for classification. The handcrafted feature approach in food recognition was previously addressed in the researches [51, 52]. Joutou and Yanai proposed a multiple kernel learning (MKL)-based feature fusion method, which adaptively integrates various kinds of image features such as color, texture, and bag-of-feature (BoF) for food recognition [51]. Rahman *et al.* proposed a method for generating scale and/or rotation invariant global texture features using the output of Gabor filter banks and demonstrated that the method provides greater classification accuracy [52]. Some food types have a high intra-class and low inter-class variance, since food consists of typically deformable objects. Handcrafted feature-based approaches have achieved low classification accuracy because of these characteristics of food images. Moreover, generally in image recognition, environmental circumstances such as background and illumination affect the object appearances in the image. Therefore, it is difficult to manually construct powerful feature descriptors that entirely illustrate all kinds of objects [53].

Nevertheless, the visual recognition paradigm changed rapidly by the appearance of some large image datasets such as the ImageNet dataset, demonstrating the power

of data-driven feature learning [54, 55]. The problem in handcrafted feature can be avoided by using the deep learning that represents learning in a computational model using multiple layer processing. These layers automatically extract features and determine them as data input features [56]. During the past years, the convolutional neural network (CNN), which is part of deep learning has become the most effective architecture to perform visual recognition.

There are some critical points in CNN practice. CNN automatically extracts relevant features and shapes from a large-scale training dataset for classification; therefore, a sufficient dataset is required [55]. Nevertheless, data augmentation (such as vertical/horizontal flipping, scaling, rotation, etc.) and transfer learning methods are frequently applied to resolve the issue of data needs in small datasets [57]. The research conducted by Zhao employed CNN for object recognition on small sample data. The studies showed that the implementation of data augmentation and transfer learning are sufficient to improve the accuracy of the result on object classification [58]. However, this focused solely on two-dimensional (2D) data augmentation [57]. The quality of the input image is another critical point on CNN. The study in Ref. [59] reported that the images under quality distortions such as blur and noise reduce CNN performance in image classification.

In the gesture recognition task, the RGB-D sensor is generally used as the input device for gesture recognition. Since it is a burden to capture many gestures from multiple subjects using RGB-D sensors for the creation of a large-scale training dataset, data augmentation is desired. However, a data extension specialized in three-dimensional (3D) data, which is obtained from the RGB-D sensor, has not been proposed. In turn, in the food constituent estimation task, a dataset with large amounts of food images given detailed constituent information – including calorie and salinity – does not exist to date. Creating a large food image dataset with detailed nutrition information comes at a high cost.

## 1.2 Purpose of Dissertation

This dissertation focuses on sign language recognition and nutritional estimation based on deep learning with the aim of realizing a sophisticated welfare technology using sensors. In particular, two methods by employing CNN for classification and estimation are proposed. First, a data augmentation for effective hand posture classification has been proposed. The proposed data augmentation strategy generates a large number of hand images with various appearances based on the three-dimensional rotation for depth image data. This is to overcome the difficulty in manually constructing large datasets, especially hand datasets, as this involves human resources, and the human hand is a complex articulated object. Second, the Xception model is applied for feature extraction and classification, since the Xception model is a state-of-the-art CNN with high computational efficiency, and has outperformed other CNN models.

On the other hand, automatic food category classification and food constituent estimation methods from food images were proposed by applying a multi-task CNN. With the aim of achieving lifestyle disease prevention, the focus is on the recognition of the food category and the estimation of calories and salinity. I realize the effective estimation of calories and salinity using a multi-task learning with food category classification, by defining both calorie and salinity estimation as a regression problem. The underlying assumption for a multi-task learning algorithm is that different tasks are related to each other. However, a large food images dataset annotated by calories and salinity is not available. Therefore, I collected a lot of food images and constructed the dataset by excluding low-resolution images before the classification task. Despite the constructed dataset being relatively small, effective and efficient learning is achieved by applying two-stage transfer learning. In our two-stage transfer learning, the CNN model is fine-tuned by using only a category-annotated food image dataset. Then, the CNN model is again fine-tuned by a small dataset for food category classification and food ingredient estimation, which are the primary tasks.

In this dissertation, Chapter 2 describes deep learning, particularly with CNN. This discussion covers the stages of preprocessing input for CNN, CNN as feature extraction, and the modification of the last CNN layer that is utilized not simply for the classification, but also for the regression. Here, the approaches to increase the effectiveness of using CNN such as fine-tuning and CNN architectures are also discussed.

In Chapter 3, the theme is to present the effectiveness and efficiency of the hand posture classification method in sign language recognition using CNN. A hand posture classification based on the Xception model and data augmentation for hand depth data is proposed. The proposed data augmentation method using the 3D rotation on depth data is effective in generating various appearances of the hand posture and increasing classification accuracy on the manually obtained dataset. On the other hand, the Xception model, which is one of the state-of-the-art CNN models, is applied to hand classification. Furthermore, the proposed method is evaluated and compared with state-of-the-art researches.

The development of an automatic food constituent estimation method from food images using multi-task CNN is detailed in Chapter 4. The research focuses on the recognition of food categories and the estimation of calories and salinity. First, a new food image dataset has been constructed by using public images from several recipe-gathering websites because there is no large food image dataset with detail information on calorie and salinity. However, the number of food images with calorie and salinity obtained from the Internet is not sufficient for the effective learning of CNN. In order to address this issue, two-stage transfer learning using a large number of food categories recognition was proposed. The effectiveness of calories and salinity estimation using multi-task learning with food category classification by defining both calorie and salinity estimation as a regression problem is demonstrated. Here, the relationship between the food category and salinity is also experimentally shown by using multi-task CNN.



Chapter 5 provides the conclusions of the overall system and comments on some future possibilities to improve the system.

# Chapter 2

## Deep Learning

### 2.1 Overview of Deep Learning

Deep learning is a subfield of machine learning with a learning model that learns its features in multiple levels. The way the human brain processes, learns, and responds to information has inspired the level of feature learning [60]. Deep learning also is known as deep neural network (DNN). While modest neural networks consist of one or multiple nodes in one layer applied for learning the weight, the deep learning mechanism is modeled in hierarchical layers for successive transformation and chained together where the deep level takes input from the output of prior levels [61]. In deep learning, each layer transforms the input data information into a more abstract representation. Deep learning analyzes the complex structure in the large dataset by employing the backpropagation algorithm to reveal how the internal parameters that are used to calculate the representation of each layer generated from the layer ahead of time should be changed. This makes deep learning capable of accomplishing various challenging issues in artificial intelligence and even overcome other machine learning techniques' capability [56, 62].

Deep learning received most attention after Krizhevsky proposed a deep convolutional neural network to classify an ImageNet dataset consisting of 1.2 million im-

ages [55]. However, previously, deep learning passed through a long history starting from the period of gaining trust, which made the Neural Network (NN) an alternative choice in machine learning, then passed through two forgotten periods, until the period when NN changed its name to deep neural network [63, 64]. The growth of deep learning began in 1940-1960, which attempted to imagine a perceptron as a simple mathematical model inspired by the workings of neurons in the brain [65, 66, 67], as shown in Figure 2.1. The development period of the backpropagation method occurred in 1960-1980. Although NN lost trust in the late 1960s because the perceptron could not be trained in multiple layers as it could not learn the simple Boolean function XOR [64], research using the NN model had been carried out until a neocognitron was proposed [68]. In 1980-1990, research by Rumelhart *et al.* succeeded in answering the doubt about NN from the previous period by implementing the backpropagation approach on NN [69]. Moreover, this research also introduced weight sharing (convolution) and became a popular approach for learning representations. LeCun *et al.* demonstrated the backpropagation approach on NN in the real-world application for handwritten recognition in 1989 [70]. In this era, researchers also tried to develop speech recognition by modifying the NN to process input as input flows [71] and explore NN for the unsupervised learning approach [72, 73, 74]. The period 1990-2000 witnessed the implementation of NN in other real fields such as control of dynamic systems [75], robots [76, 77] and games [78, 79]. Even though NN with the backpropagation method has shown better results, in the middle of 1990s the NN encountered a problem as backpropagation did not work well for normal NN with many layers. Backpropagation depends on finding the error in the output layer and successively dividing it among prior layers; thus, this matter can give rise to vanishing or exploding gradients [63]. This made NN once again lose trust, and researches turned to SVM, which was considered more reliable [80]. Research with efforts to improve the performance of NN continued in 2000-2012 to regain the trust of researchers in NN, including by changing the term NN to deep learning, until Hinton *et al.* succeeded in publishing a

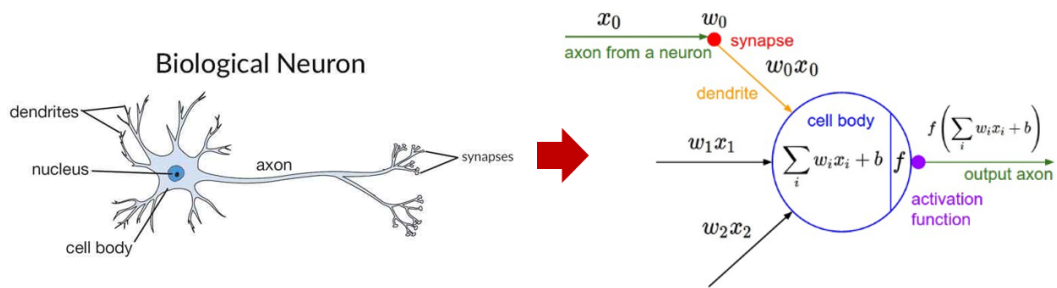


Figure 2.1: Linear process on fully connected layer inspired by human biological neuron [81]

paper in 2006 that was considered a significant breakthrough in NN [82]. The research point is that multiple layers NN can be trained well, if weights are initialized smartly rather than randomly by training each layer one by one with unsupervised learning, starting with a much better weight than just assigning random weights and completing it with a round of supervised learning as in regular NN. After this breakthrough, efforts to improve deep learning are increasing [83, 84, 85, 86, 87, 88, 89, 90], which includes the training of deep learning using graphics processor [91]. The availability of the ImageNet dataset in 2009 to facilitate machine learning also affected the deep learning performance [54]. The 2012 ImageNet competition became a notable moment that showed the success of deep learning by Krizhevsky's research. The research applied a deep convolutional neural network, which combines ideas from the scientist's previous researches [55]. After this period, deep learning was increasingly employed in research or software applications.

The tremendous achievement currently acquired by deep learning is owed to the availability of a fundamental theory for NN, large datasets, and the presence of supported hardware, such as GPU and memory. Along with these accompanying supports, deep learning exhibited several advantages [56, 92]. The first is robustness, as the features design is not required in deep learning since the features are automatically learned to be optimal for the task, including learning natural variations in the data. The second is the ability to generalize, as the NN architecture can be adapted relatively easily for many different applications and data types. Subsequently, another

advantage of the NN is scalability, as its performance will improve by appending more data and parallelizing it.

The practice of deep learning in various fields makes it possible to divided deep learning into several architecture types based on their purposes, as follows:

- Unsupervised learning network

Unsupervised learning trains a set of inputs without requiring labeled data to locate some hidden structure within that data [82]. The structure can be utilized to find groups of identical/pertain data that are useful for clustering, anomaly detection, association meaning, and latent variable models [48, 93]. Deep belief networks and autoencoders are two network models of unsupervised learning. The effective training strategy, namely the unsupervised pre-training network, also has been achieved by applying unsupervised learning before supervised learning. This strategy helps prevent overfitting for smaller datasets [94]. The approach is pre-training one layer in one time and using the unsupervised learning algorithm for each layer to capture main variations from the input and hold information. Then, deep architecture is fine-tuned correspond to the criteria of supervised training with gradient-based optimization [82, 84].

- Convolutional neural network (CNN)

CNN is a DNN variation that has been designed to process input data in the form of multiple arrays. However, it is mostly applied to the image classification task. The key of CNN is to create many feature detectors that take the spatial arrangement from the input by using local connections, shared weights, pooling, and many layers [95]. The first stage comprises the convolutional layer and the pooling layer to create the features. The convolutional layer utilizes a kernel to locally connect the receptive field (a particular region of the input layer) to a feature map. The kernel is slid over the pixels of input and performs scalar products against the receptive field to create the feature maps. This process of

creating the feature is called convolution. The convolutional layer is commonly woven together with the pooling layer. The pooling layer aggregates multiple feature values into a single value using a max, mean, or summation operation to reduce the number of features for the next layer. At the last stage, the features from the stage ahead of time are flattened and continued to a fully connected (FC) layer, which is a regular NN for the classification task. Some CNN models consist of some convolutional layers and a rectified linear unit (ReLU) as an activation function, which is operated before the FC layer to increase the non-linearity. Further discussion on CNN layers is provided in Section 2.2.

- Recurrent neural network (RNN)

RNN is the NN designed to recognize the pattern in sequences of data, such as text, handwriting, speech, or numerical time series data emanating from sensors or other sources. The networks are called recurrent since they perform the same computations for all elements from an input sequence. RNN architecture can have some different structures. The fundamental feature of RNN at least consists of a standard multi-layer perceptron (MLP) with an additional time variable so that activations can flow in a loop. The time component allows the output of each element to rely on all previous computations (not only the current input, but also the input that it encounters earlier) by updating a kind of vector state that contains information about all past elements of the sequence [60, 96]. Recursive neural network and long short term memory (LSTM) network are two instances of RNN. The recursive neural network is a generalization of a recurrent neural network with a tree structure with a fixed number of branches, and the LSTM network is the extension of RNN to overcome the exploding/vanishing gradient problem occurring due to the repeated use of the recurrent weight matrix [60, 97].

## 2.2 Convolutional Neural Network

In recent years, CNN has become the first choice to accomplish various challenges in computer vision, such as recognition, classification, and estimation. This is due to the impact of the ability shown by CNN that can surpass human capacity in performing the classification [55]. Computer vision researchers mostly apply a NN to accomplish complex problems that hinder the recognition of an object like image segmentation, lighting, viewpoint, etc. Traditional feed-forward neural network process the images in neurons and classify them into an output of True and False Likelihood. However, this method cannot attain a good result for the deformed image, since the network only knows one pattern [98]. However, a CNN can recognize an object even when the object experiences variation translation, since CNN collects and models small information sequentially and combines them in the deeper network [99].

The complex architecture of CNN stacks multiple and different layers for classification. A CNN consists of one or more convolutional layers often followed by a pooling layer, and thereafter there are one or more fully connected layers where the biological neuron inspired the calculation. The first few layers extract particular features like edges and form the templates for edge detection. The deeper layers merge them into simpler configurations and then into templates of different object positions, illumination, scales, etc., which show the object of interest. The final layer matches an input image with all the templates and makes a prediction [99].

Formally, each input image used in learning is divided into compact topological portions arranged in three dimensions: width, height, depth (number of channel). Every sub-portion put on the convolutional with the filters (kernels) is set to look for particular patterns [60]. The filter moves along the input matrix, performs the scalar product with the receptive fields in the input matrix to generates a new matrix (convolution matrix or feature map). Figure 2.2 describes the convolution process. A set of filters  $w$  is convolved across the input image  $x$  to generate the two-dimensional

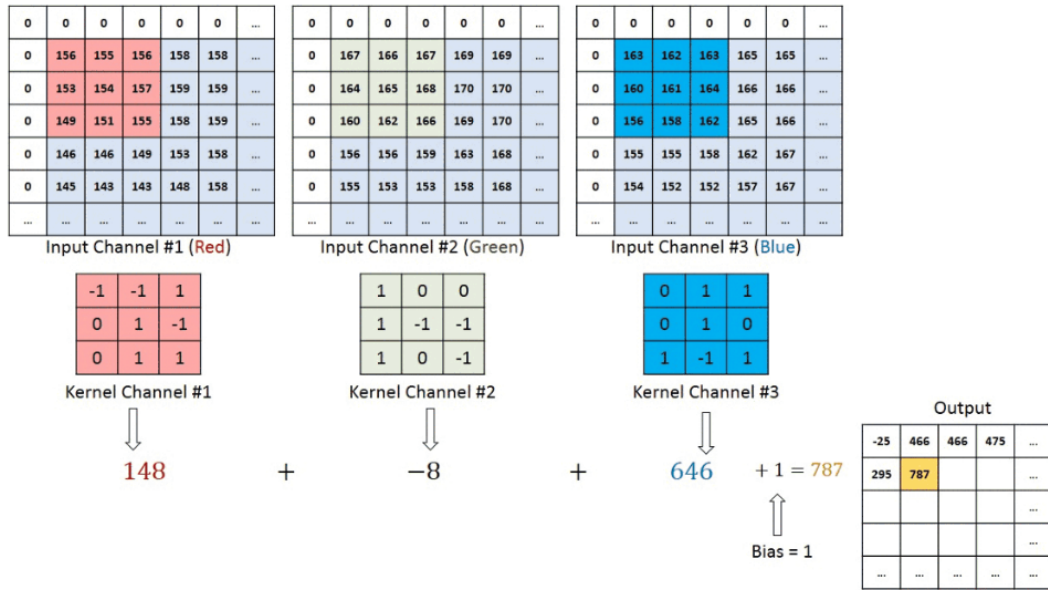


Figure 2.2: Convolution process by the filter set to the input [100]

feature maps  $y$  can be formulated as follows:

$$y_{i,j} = \sum_{k=1}^n \sum_{l=1}^n w_{k,l} x_{i+k-1,j+l-1} + b, \quad (2.1)$$

where  $n$  is the width and height of the kernel, and  $b$  is the bias. The filter depends on four parameters: size, depth, stride, and zero-padding. The larger size of the filter will tend to overlap and create increasingly large outputs. The depth filter refers to the number of filters that are used for learning to seek something different from the input. Stride is a measure of spacing neurons; Stride 1 means that the filters move one pixel at a time, while stride 2 makes the filters jump two pixels at a time when the filter slides around. Increasing the stride will produce less overlap and reduce the output size. Using a small stride generally works well and enables us to manage down-sampling and scale reduction at pooling layers. The zero-padding hyperparameter is padding the input border with zero value. It enables adjustment of the input and output sizes to be the same and makes the spatial size of output volumes controllable. It is very complicated to make and arrange complex kernels and show a specialized



feature engineering technique. Moreover, the feature engineering technique has the challenge of working well on one task but not on the other. The convolution process can reduce the need for features engineering, which is its advantage.

The implementation of the activation function after the convolution layer can increase the non-linearity by taking vectors and performing a certain fixed pointwise operation on the output of the convolution layer. There are three activation functions: sigmoid function, hyperbolic tangent, and the rectified linear unit (ReLU) function.

- Sigmoid function:

$$y = \sigma(x) = \frac{1}{1 + e^{-x}} \quad (2.2)$$

The sigmoid function displaces the value between 0 and 1 to distribute the probability of the output.

- Hyperbolic tangent function:

$$y = 2\sigma(2x) - 1 \quad (2.3)$$

The advantage of this function is mapping the negative values on strongly negative and mapping the zero input near zero on the tanh graph.

- ReLU function:

$$f(x_i) = \max(x_j, 0) \quad (2.4)$$

ReLU charts all negative values into zero without changing zero and positive values. This makes the computation more efficient, as ReLU does not activate all neurons at the same time. Currently, this function is the most widely used in CNN.

Following convolutional layers, pooling layers aggregate multiple layers into a single layer to reduce the computational time of the following layers and enhance the robustness of the feature related to its spatial position. A pooling layer operates by

dividing the convolutional region into several sub-regions and subsequently selecting a single representative value (by max-pooling, min-pooling, or average pooling). The size of the pooling layer result is based on the spatial extent and stride with the following calculation:

- The size of the convolutional layer result:  $W_1 \times H_1 \times D_1$

- Hyperparameter requirement

The spatial extent:  $F = k \times l$

The stride:  $S$

- The calculation of the size of the pooling layer output:  $W_2 \times H_2 \times D_2$ ,

$$W_2 = \frac{W_1 - k}{S} + 1, H_2 = \frac{H_1 - l}{S} + 1, \text{ and } D_2 = D_1$$

Thus, the features generating from the pooling layer can be obtained as follows:

$$y_{i,j} = \max \{x_{i+k-1,j+l-1} \forall 1 \leq k \leq n \text{ and } 1 \leq l \leq n\} \quad (2.5)$$

The formula above describes the pooling process using max-pooling, with  $y$  as the pooling results. Max-pooling is a pooling layer theory that focuses on the most responsive features, making it the most common and best choice for image classification. The pooling process with max-pooling theory is also shown in Figure 2.3. Min-pooling is preferred as an additional step in situations that need to prevent overly specific classification. The pooling layers avoid the overfitting where the network learns the position of features too specifically, so that the network no longer focuses on a particular location of features and gains better capabilities to generalize.

The convolutional layer and pooling layer extract and learn the features using the backpropagation method to calculate gradients during training. The method comprised of three procedures looks as follows:

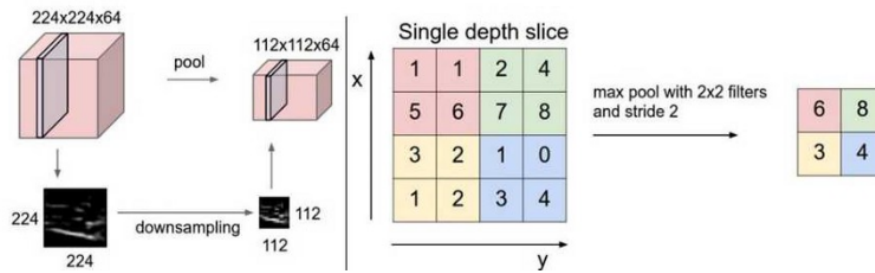


Figure 2.3: Pooling layers process in convolutional network [101]

- Forward pass: embraces the computing process as a sum of all feature maps (output) obtained from the convolution action, traversing through all neurons from the first to the last layer.
- Calculation of loss functions from each kernel output allows setting the individual weight of each kernel as needed.
- Backward pass: refers to the computation to obtain changes in weights by calculating the gradients of loss concerning output. Computing is done recursively, applying the chain rule starting from the last layer backward to the first layer.

Backward and forward pass together make up one iteration.

The last hidden layer of the convolutional network is generated by the fully connected (FC) layer, which is equal to the number of classes in classification purpose or amount of possibilities in the regression task. After the convolution layer rolls for the forward pass, the output layer has a loss function that is responsible for backpropagation and updating weights and biases to reduce error and loss. In a mathematical formula, a linear transformation on the fully connected layer process is shown as follows:

$$y = A \cdot x + b, \quad (2.6)$$

where  $A$  is kernel matrix (filter),  $x$  is the input matrix,  $b$  is the bias, and  $y$  is the result.

Finally, CNN also appends logistic regression, which is responsible for allocating the probability of each class either in a binary or multi-class problem. The particular activation function is often placed at the end of the FC layer as logistic regression, that is a soft-max function (for the multi-class logistic regression) or sigmoid function (for the two-class logistic regression):

$$S(y_i) = \frac{e^{y_i}}{\sum_{j=1}^k e^{y_j}}, \quad (i = 1, 2, \dots, k) \quad (2.7)$$

where  $y$  is output of the fully connected layer. The soft-max function produces a discrete probability vector and can be used as a fancy normalizer.

The loss function guides the training process in neural networks to measure the inconsistency between predicted value ( $S$ ) and the actual label ( $L$ ). The mean square error (L2 Loss), cross-entropy, and mean absolute error (L1 loss) are some loss functions that are commonly applied in the convolutional neural network for the classification or regression task.

Cross-entropy:

$$D(S, L) = - \sum_i L_i \log(S_i) \quad (2.8)$$

Mean square error:

$$MSE = \frac{1}{n} \sum_{i=1}^n (L_i - S_i)^2 \quad (2.9)$$

Mean absolute error:

$$MAE = \frac{1}{n} \sum_{i=1}^n |L_i - S_i| \quad (2.10)$$

Figure 2.4 reveals the operation of logistic regression on the last layer of the CNN and loss function in measuring the performance of the model, whose classification output is between 0 and 1.

The general form of the CNN is stacking of several convolutional layers followed by a pooling layer, and repeating this pattern until the image is merged spatially to a small size, as seen in Figure 2.5. It is also common to transition the networks to the

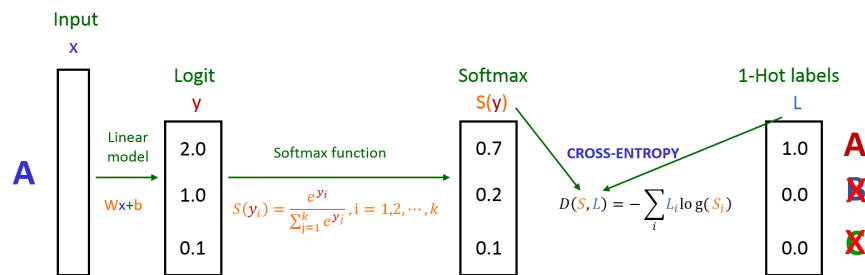


Figure 2.4: Activation function and logistic regression operation at the end of convolutional network in image classification

FC layer with the last layer of the FC layer is a class score output. In practice, the CNN architecture that works well using ImageNet is the most widely used. Some of these architectures, among others, are [101]:

1. LeNet (1989): the first successful application of CNN that was used to read handwritten zip code digits. LeNet architecture is composed of seven layers [70].
2. Alexnet (2012): the architecture was submitted to the 2012 ILSVRC and became a winner. This research also became the first study to succeed in popularizing CNN. Architecture AlexNet is similar to LeNet, but deeper owing to the convolutional features layer stacked on top of each other. This research also utilizes two new concepts at that time, namely max-pooling and ReLU activation, which enhance the advantages of their architecture [55].
3. ZFNet (2013): the research won the ILSVRC 2013 by improving the AlexNet

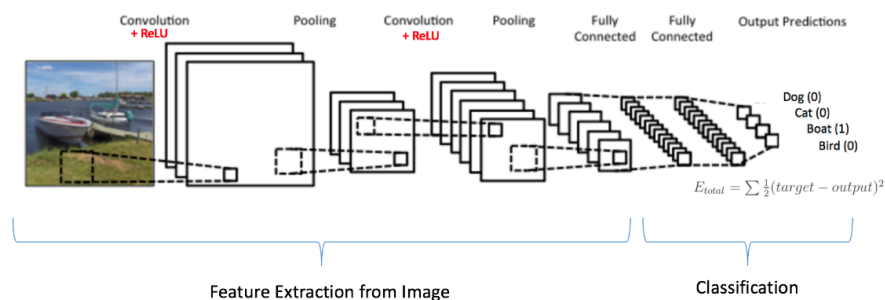


Figure 2.5: Illustration of convolutional neural network architecture [102]

architecture. Research tweaked the architecture hyperparameter, specifically by expanding the size of middle convolutional layers and decreasing the stride and filter [103].

4. VGG (2014): the runner-up of ILSVRC 2014, which has contributed in demonstrating that the network depth is an essential component for better CNN performance. They arranged several network models, where the best network model consisted of 16 Conv/FC layers. The architecture is displayed modestly in uniformity, which only has a  $3 \times 3$  configuration for the convolutional layer and  $2 \times 2$  for the pooling layer from the beginning to the end. The lack of this architecture makes it more expensive to evaluate and employs more memory and parameters (140 M), with the highest number of parameters being in the first fully connected layers [104].
5. GoogleNet/Inception (2014): the winner of ILSVRC 2014. This architecture constructs the inception module, with a smaller convolution that allows reducing the number of parameters to a mere four million. In the Inception modules, each type of layer extracts information that is different from the input. Information obtained from the  $3 \times 3$  layer will be distinguished from the information collected from the  $5 \times 5$  layer. In inception as well, dimensionality is reduced using a  $1 \times 1$  convolution [105].
6. ResNet (2015): this architecture wins the ILSVRC 2015. Additional layers to the deeper network estimate the mapping better than the shallower network. Thus, deeper networks should show better accuracy. However, the experiment that has been carried out cannot demonstrate this, since when the deeper network starts converging, a degradation problem appears where accuracy saturates and then decreases due to overfitting. This architecture attempts to deliver a solution to this problem by displaying a residual learning framework that can facilitate the training on deeper networks. The architecture uses  $3 \times 3$  filters at most and

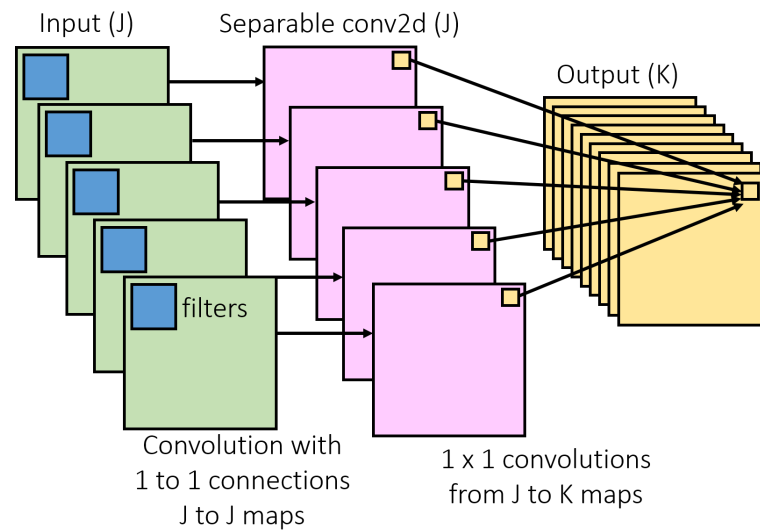


Figure 2.6: Depthwise separable convolution on Xception

downsamples the CNN layers with stride 2 [106].

7. Xception (2016): Xception, meaning “extreme inception” is an extension of the Inception architecture that replaces the standard inception modules with depthwise separable convolutions. Depthwise separable convolutions (Figure 2.6) consist of a depthwise convolution (a spatial convolution performed independently for each channel) followed by a pointwise convolution (a  $1 \times 1$  convolution across channels). This architecture idea is based on the inception module hypothesis, which separates cross-channel correlations and spatial correlations to simplify and efficiency the convolution process. As is known, in a traditional convolution, convolutional layers seek out correlations across both space and depth. The differences between Inception and Xception can be seen in Figure 2.7. Xception maps the spatial correlations for each output channel separately, followed by depthwise convolution  $1 \times 1$  to capture cross-channel correlations without implementing non-linearity. Meanwhile, Inception performs the  $1 \times 1$  convolution first and applies ReLU after the two operations [107].

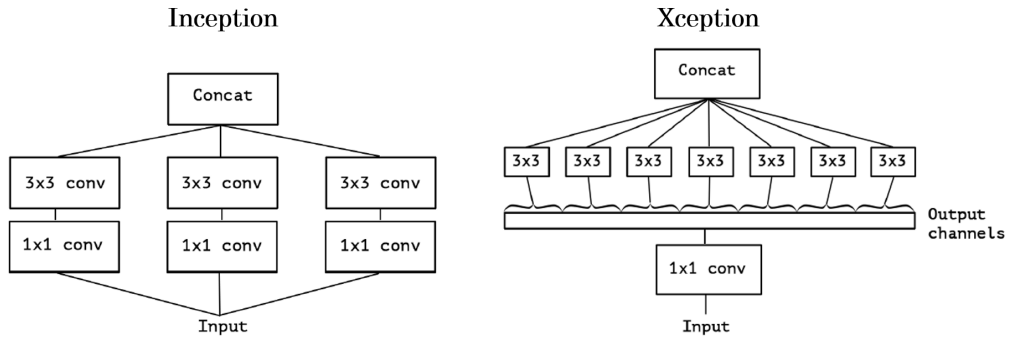


Figure 2.7: Differences between Inception and Xception [107]

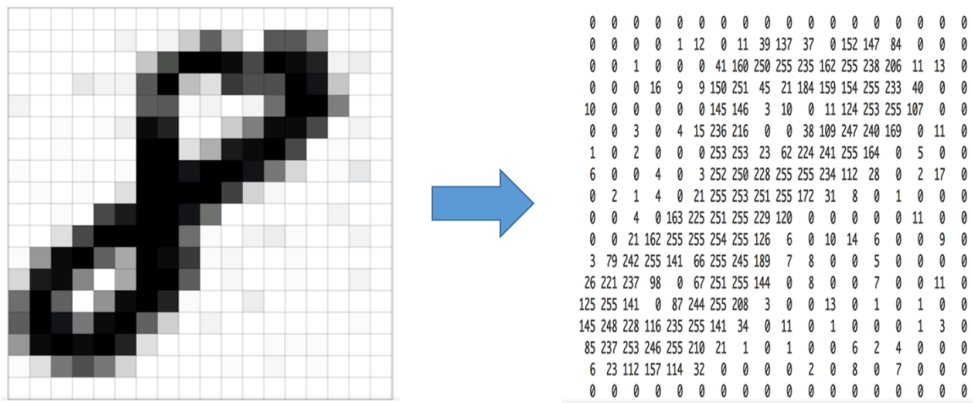


Figure 2.8: Pixel values in an image [108]

### 2.3 Preprocessing

An image is a set of numbers data in the range of 0 ~ 255 displayed on a grid of pixels, as shown in Figure 2.8. The pixel values of the color image are represented by a group of three matrices, indicating variations of red, green, and blue, as shown in Figure 2.9. A grayscale image has one channel, and a color image has three channels, each reflecting RGB information. Classification on CNN also requires input images that have been accompanied by labels for training.

There is no pledge that in the raw image, the distribution ranges of feature values are the same. Thus, the learning rate that is used multiplicatively will lead to a different correction for each feature, and during the gradient descent, it will burdensome features more than others. Preprocessing is applied to the image before entering the



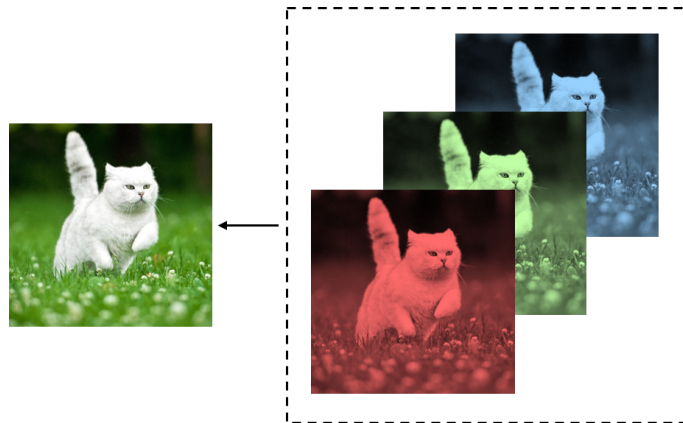


Figure 2.9: Three channels of an RGB color image

convolutional network to control this effect. As the first step for preprocessing, all images for CNN input are converted to the uniform size and generally in square shape. A lot of preprocessing options can be used to the input images before feeding them to the network, such as mean and standard deviation, data normalization, dimensionality reduction, and even data augmentation [109]. Data normalization is significant to assure that each input parameter has a similar data distribution because it will make convergence faster during the network training. Considerations for reducing dimensions are often needed as well, because it makes training problems easier to control. This is done by applying dimensionality reduction, where the RGB channels are changed into a single grayscale channel. The augmentation of a dataset with several versions of an image is to allow the NN to obtain exposure from various objects to degrade the possibility of recognizing unwanted characteristics. Minor alterations to existing image datasets will be thought of as distinct images in the NN. The variations can be gained from horizontal/vertical flipping, rotation, rescaling, cropping, or shifting on two-dimensional data calculations.

## 2.4 Transfer Learning CNN

Transfer learning is a machine learning method where a model built for a task (initial task) is reused as a starting point (to transfer its knowledge) on a model to complete the second task (main task). Transfer learning is chosen as a way to enhance the performance of the second task model when the dataset used is small. Therefore, in most cases, a pre-trained convolutional network on a huge dataset (e.g. ImageNet, which contains 1.2 million images with 1000 categories) is used as initialization or a fixed feature extractor for the second task [110]. Transfer learning only works in deep learning if the model features learned from the initial task are general. A pre-trained convolutional network on a large dataset has learned to recognize the trifling shapes and small parts from different kind objects in its first few layers. Simply adding dense layers at the end of the pre-trained network is very beneficial in recognizing objects in the new dataset, since the pre-trained network will learn what combination of these learned features.

There are two common approaches in transfer learning, namely the develop model approach and the pre-trained model approach [111]. First, the develop model approach reuses the model that was designed and used in the initial task for the next (main) task. In the initial task, the model is trained from the beginning (scratch) on an abundance of data that has a relationship with input data, output data, or concepts during the mapping from input to output data against the next task. Second, the pre-trained model approach utilizes a pre-trained model from available models for the second task. Optionally, both approaches allow for fine-tuning. Fine-tuning itself can be applied by choosing one of the following three strategies: training the entire model, training some and leaving the others frozen (weights do not change during training), or freezing the convolutional base to keep the convolutional layers in their original form and use their output for the classification layer [112].

## Chapter 3

# Hand Posture Classification of Augmented Depth Data using A Convolutional Neural Network

### 3.1 Background

Welfare technology provides support for humans in their activities, such that people receive the amenities enhancements in daily life. In general, welfare technology is not only intended for elderly people and people with a disease, but also disabled people such as deaf, speech impaired, visually impaired, etc. The sign language recognition system is a welfare technology that can be used as a communication tool for deaf people. Communication with deaf people often requires a human interpreter because of the limited number of people who apprehend sign language. However, employing a human interpreter is inconvenient and expensive; therefore an automatic sign language recognition system is needed.

Sign language recognition has long been the research object, because the human hand is a complex articulated object consisting of many connected parts and joints [113], hence it becomes interesting to define and classify hand features into as

attributes and characteristics of the hand. Sign language generally consists of two components: hand posture, which has a function to describe the particular concept such as name, acronym, brand, etc.; and arm motion, whose purpose is to explain word by word until its sequences form a sentence. The hand posture forms fingerspelling with unique and discrete hand configurations, while arm motion forms a movement continuously with hand configurations [114]. Therefore, the development of sign language recognition depends on two things: hand posture classification and arm motion recognition. Hand posture classification is essential in sign language recognition, since the sign is done by posturing each alphabet/number to describe the name of an object. This process is likewise not easy, as it is faced with several challenges. The challenges are the similarities between some signs, the variation in appearance (due to viewpoint difference), and the variation in performance (due to subjects difference) [115]. The examples of both variations of the hand sign are shown in Figure 3.1.

The presence of deep neural networks makes conventional hand posture classification be realized by two approaches: the handcrafted feature-based and deep learning-based approaches. The handcrafted feature-based approach entails several stages in feature extraction, such as hand detection and hand segmentation up to feature extraction itself. In some cases, employing additional devices such as gloves to make hand segmentation easier, however this is cumbersome for the user. In the current case, applying the sensor, particularly the depth sensor is preferred because, in addition to facilitating the process of hand detection and hand segmentation, the depth image of the sensor can also be utilized in the feature extraction process. Recently, the achievement of deep learning, in particular the CNN in computer vision has made researchers shift over from handcrafted feature-based to deep learning-based methods. This is because deep learning automatically generates and learns features at a low computational cost.

Hand posture classification using a CNN requires a large of hand posture dataset. However, it is difficult to obtain a large dataset involving human resources. Data aug-

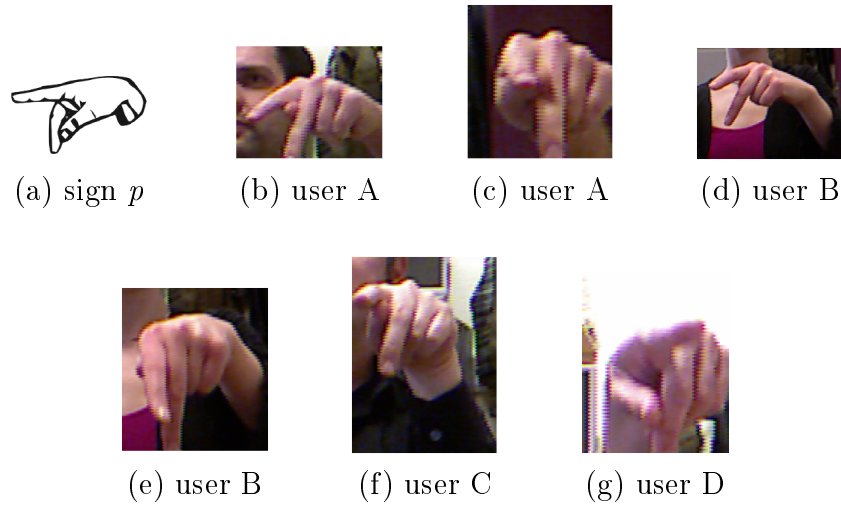


Figure 3.1: Hand posture variation due to viewpoint difference and subject difference [115]

mentation has been employed to solve the problem of a small dataset by augmenting data with slight variations for the training dataset. Existing data augmentation employs image processing methods such as horizontal/vertical flipping, rotating, scaling, and others. It should be noted that all of this data augmentation processes is merely done on a 2D image; therefore, it cannot be applied to the depth image data. Data augmentation using depth image data is possible by utilizing 3D operations. The broader scope of the augmentation process using depth data will also create more variations of the original data. Thus, this 3D data augmentation not only serves to solve the problem on a small dataset, but also to complete the issues of appearance and performance variations in sign language.

In the research presented in this dissertation, to realize the effective and efficient hand posture classification, a hand posture classification method based on the data augmentation and the state-of-the-art CNN model is proposed. To overcome difficulties in collecting hand posture data and accomplish the challenge in hand posture classification, I focus on presenting data augmentation on a 3D operation to generate the various appearance of the hand postures by utilizing depth data captured from

the depth sensor. This augmentation method is applied to collect a large dataset with various appearances before the dataset is fed into a CNN model. Feature extraction and classification problems are completed in the Xception model as a state-of-the-art CNN model, since Xception has high computational efficiency and has outperformed other CNN models. In addition, ASL dataset for sign language recognition is used as a benchmark of hand posture classification to evaluate the effectiveness of the proposed method.

## 3.2 Related Works

Many hand posture classifications have been proposed because of the importance of sign language in assisting human life [116, 117]. Feature extraction, as the most important part in hand posture classification is optimized starting from the application of handcrafted feature-based to the current popular approach, i.e., deep learning.

Studies on hand posture classification initially use color images as input [118, 119]. Kim *et al.* [118] use a color-based model (mixture of Gaussian for skin color vs. single Gaussian per background pixel) for automatically segment the hand from each image, apply SIFT to extract the feature vector based on the local histogram of oriented image gradients, and utilize a multi-layer perceptron as the classifier to recognize ASL. Gautam and Kaushik [119] propose a novel technique to recognize fingerspelled ASL. This research requires the subject to wear a wristband (see Figure 3.2) for distinguishing the hand from the elbow and detects the skin color by specifying threshold ranges in the hue frame for hand detection. Point descriptors were used for feature extraction, and Euclidean distance was employed as the classifier.

The advancement of the depth sensor, which yields the depth data, provides distinctive convenience for the hand posture classification. Depth information simplifies the hand detection and hand segmentation. Pugeault and Bowden [115] proposed a system to classify fingerspelling hand shapes in the real-time condition. As shown

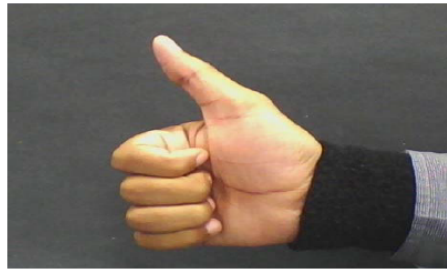


Figure 3.2: User wearing a wristband [119]



Figure 3.3: Image segmentation result [115]

in Figure 3.3, the hand is segmented by assuming that it is a connected region with depth alterations that are less than or equal to 20 cm. Their method extracts the hand shape feature by convolving the intensity and depth images with the Gabor filter and classifies them on a multi-class random forest.

Kuznetsova *et al.* [49] propose the recognition of sign language using ESF descriptor for feature extraction and an MLRF for classification. This research seeks to solve the problem in recognizing sign language: variance in the appearance and variations in the performance. Therefore, after the hand detection and the hand segmentation by thresholding depth values, the depth data are used to derive rotation, translation, and the scale invariant feature in the descriptor.

The visual recognition paradigm changed rapidly after the appearance of the ImageNet dataset, demonstrating the power of data-driven feature learning. During the past years, CNNs have become the most effective architecture for performing vi-

sual recognition. In research [120], color images of sign language are generated from the demonstration video by sampling and concatenating screenshots as input for the CNN. Ji *et al.* [120] proposed a sign language learning system using a CNN with three convolutional layers for extracting and learning the features, and two fully connected layers as the classification method. In the research by Pigou *et al.* [121], CNN is performed to recognize the signs and the gesture in sign language recognition. They apply the CNN with three layers on depth images to extract the hand and upper body features, then use an artificial neural network (ANN) on the last layer as a classifier to distinguish between each action or sign. They worked on ChaLearn Looking at People 2014 (CLAP14). Moreover, Kang *et al.* also employed CNN for sign language recognition on their hand posture dataset. The dataset consists of letters and numbers signs of the American sign language. They employ the wristband to make a gap around wrist and utilize CNN with an architecture similar to Caffenet.

Data augmentation has been fitted to increase classification performance. Simonyan *et al.* utilize color shifting, flipping, and scale jittering for augmentation; He *et al.* use resize/scaling, color augmentation, and flipping; and Aquino *et al.* show the effectiveness of online data augmentation with and without balancing the training set [104, 106, 122]. There are two approaches of data augmentation on CNN; the first is to generate augmented data before training the classifier, whereas the second approach attempts to learn augmentation through a prepended NN [123]. However, both approaches that have been performed are limited to 2D operations. Takimoto *et al.* [113] propose a robust hand posture recognition method for posture alteration by rotating the hand depth information gained from the depth sensor in 3D directions. This method is the foundation for the proposed data augmentation method to compose more hand postures. Each output generated from the calculation is stored first to become the input for the CNN model.



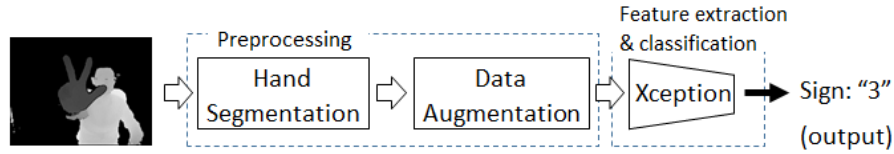


Figure 3.4: Overview of the proposed hand posture classification on training

### 3.3 Proposed Method

CNN requires a sufficiently large dataset to attain high performance in recognition tasks. However, it is difficult to construct a large amount of hand posture image in a dataset moreover with different appearances, since taking the image of hand posture involves human resources. Our research aims to virtually generate a large hand posture dataset, by applying data augmentation methods for a manually collected small hand depth dataset. In addition, sufficient hand posture classification is achieved by using the generated large dataset and Xception which is state-of-the-art of CNN.

The overview of the proposed hand posture classification is shown in Figure 3.4. At the initial step, fingerspelling indicated by a subject is captured using an RGB-D sensor as input data for the hand posture classification. In many cases, the body of the subject is included in the entire captured image. Therefore, only the hand region is detected and segmented for initial preprocessing in our method. To construct a large hand posture dataset, the proposed data augmentation method is applied to specified for hand depth data to the segmented hand image. Then, the proposed CNN for classification of hand postures is based on the Xception architecture. In the training phase, CNN is efficiently trained using many generated hand images.

I utilize the dataset provided by Kang, which consists of two kinds of hand posture data – JPEG data and raw data [124]. In this case, the raw data is used for the proposed method. The raw data is captured using a Senz3D RGB-D camera, which is able to capture depth images with  $320 \times 240$  pixels resolution. The dataset has 31 sign classes of hand depth data – 24 alpha and seven numeric, which are collected from five different subjects. The number of images for each sign is different; overall,

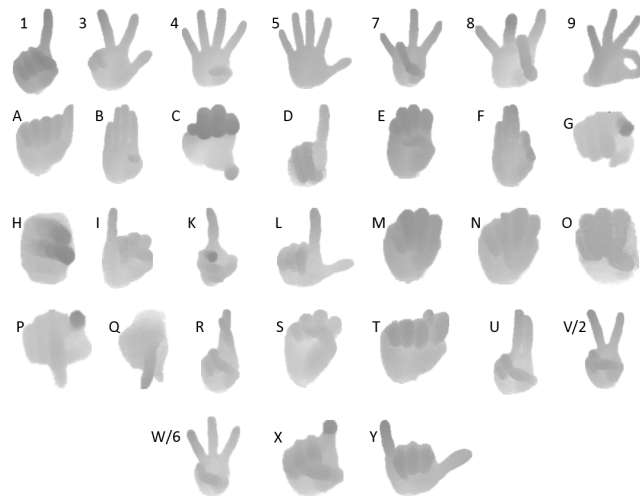


Figure 3.5: Examples signs with depth images of Kang's dataset

the total number of images is 33,739. The appearance of 31 signs on depth data is shown in Figure 3.5. Since the proposed method focus merely on the classification of hand postures, the identical sign posture of the alpha sign and numeric sign, 'V' – '2' and 'W' – '6', are only used to visualize the alpha sign. In addition, alpha signs that require a motion, such as 'J' and 'Z', are excluded from the dataset.

### 3.3.1 Hand Segmentation

Hand segmentation is utilized as an important preprocessing step in feature extraction on both handcrafted feature and deep learning approaches. Hand segmentation consists of the detection of the hand position from a captured image, and the separation of only detected hand region from the background. Hand depth data information resulting from hand segmentation is then utilized in the data augmentation process. An instance of captured raw data is shown in Figure 3.6(a). The gray level of each pixel in this raw data means the distance from an RGB-D sensor. Whitish pixels depict distances further from the sensor. An utterly black pixel indicates that the distance could not be measured. First, only a hand region is extracted from the raw image as preprocessing of hand posture classification, since the captured raw image includes not only a hand, but also a body and a background.

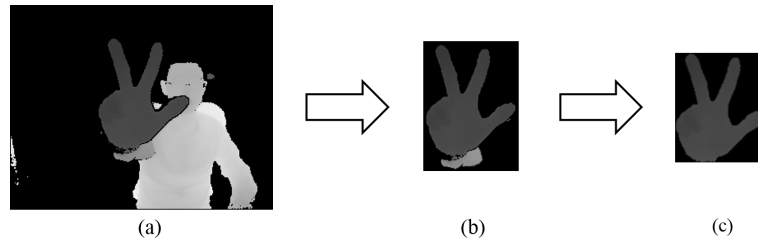


Figure 3.6: The result of hand region segmentation method on raw data

The subject's hand is considerably closer to the RGB-D sensor than the body in this dataset. In addition, there are places where the distance cannot be measured around the hand region, since the hand is located near the sensor. Therefore, hand segmentation is achieved by applying thresholding, labeling, and morphological operation to the raw image on the basis of these preconditions. Thresholding is implemented to detect the Subject hand and localize them, so noise around the hand area may occur, as shown in Figure 3.6(b). The application of the labeling algorithm and morphological operation, such as dilation and erosion, reduce the noise around the hand region. A rectangle region containing the extracted hand region is stored as a monochrome image. Figure 3.6(c) shows the extracted hand region result obtained from all processes.

### 3.3.2 Data Augmentation

Even if the same sign is formed by the subject's hand, the appearance of a formed hand sign is different depending on the time and situation, since the human hand is a complex articulated object consisting of many connected parts and joints. Individual differences such as hand size or thickness also affect the diversity of appearance in hand signs. Furthermore, when a subject performs the sign to a camera, there is the possibility that the hand does not precisely direct to the front of the camera. Figure 3.1 shows various appearances of the same posture demonstrated by different subject. Therefore, to improve the classification accuracy of CNNs, a large hand posture dataset that includes individual variations and hand directions is required.

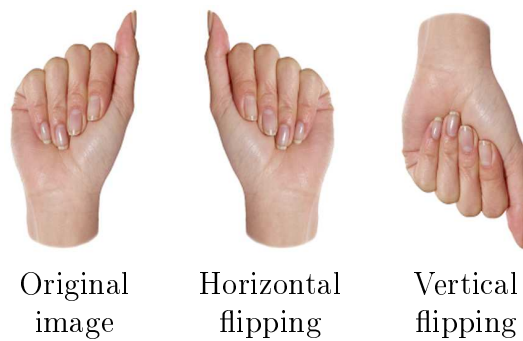


Figure 3.7: Possible transformation imposed on the image of sign ‘A’

Data augmentation is the technique utilized to artificially expand the number of data and variation appearances. This is done by modifying the original data in the dataset with one or more transformations [125]. The transformations are performed using a range of 2D operations specifically on the 2D image such as flip, rotate, shifts, and others. In particular on CNN, the data augmentation purpose is to extend the training dataset with new plausible examples generated from the transformation. As an example, in Figure 3.7, horizontal flipping can be applied to the following fingerspelling, since there is a possibility fingerspelling done by left-handed people. However, it makes no sense to do vertical flipping on the image due to the fact fingerspelling posturing or fingerspelling image capturing is not possible in this appearance. Thus, transformation must be carefully chosen.

The presence of depth data allowed us to gain abundant augmentation results by harnessing the depth information and using the 3D transformation. The proposed data augmentation took the hand depth data results from the data segmentation and processed them using 3D rotation transformation rather than 2D rotation to virtually generate more hand depth data. The large of the angle chosen for rotation is not very wide, considering the impossibility of its appearance based on the fact and the avoidance of similar appearance to the result of vertical flipping. The amount of new data appearances in the dataset depends on the length and width of the rotation and

the direction of rotation.

In this research, a 3D voxel model of the hand surface is constructed from a depth image. First, the extracted hand region is represented by a 3D voxel model. Voxelization of the proposed 3D voxel model is described by

$$Voxel\_Hand(x, y, z) \in \{0, 1\} \quad (x, y, z = 1, 2, \dots, N), \quad (3.1)$$

where the parameter  $N$  depicts the width of the proposed model, and each voxel represents 1 mm in real space;  $x$ ,  $y$ , and  $z$  are the width, height, and depth, respectively. In the 3D voxel model, hand shape is represented by the voxel with 1, and voxels other than the hand region are represented by zero. In the voxelization to convert a pixel to the voxel, calibration is necessary due to the different units of height and width for pixels (in pixel) and voxel (in mm). For this calibration, the characteristics of the depth sensor were analyzed as a preliminary experiment.

The width and the height of hand size changed after unit transformation; therefore nearest neighbor interpolation is applied to form a new hand depth information. The proposed voxelization is performed on the basis of the centroid of the hand region. The center of the voxel model,  $Voxel\_Hand(N/2, N/2, N/2)$ , is defined as the centroid of the hand region. Each hand region is modeled by

$$Voxel\_Hand\left(\frac{N}{2} + i, \frac{N}{2} + j, \frac{N}{2} + (z - z_c)\right) \in \{0, 1\}, \quad (3.2)$$

where  $i, j, z$  mean distance between the centroid of the hand and the target hand region in real space; and  $z_c$  means the distance between the sensor and the centroid of the hand.

The human hand is able to rotate in three directions—yaw, pitch, and roll [121], as shown in Figure 3.8. A rotation model for hand posture recognition with the center

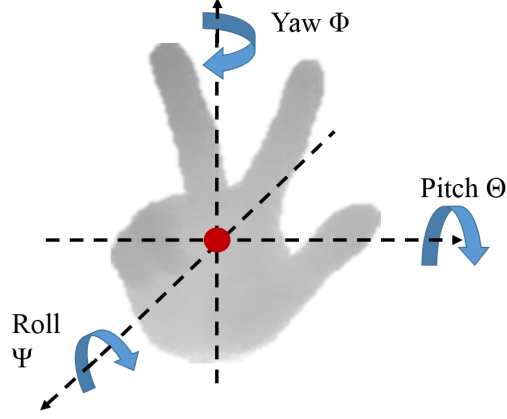


Figure 3.8: Direction of hand rotation

point  $(0, 0, 0)$  is described by

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = R_\phi R_\theta R_\psi \begin{bmatrix} x \\ y \\ z \end{bmatrix}. \quad (3.3)$$

Therefore, a rotation model with the centroid  $(x_c, y_c, z_c)$  as a center point is

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = R_\phi R_\theta R_\psi \begin{bmatrix} x - x_c \\ y - y_c \\ z - z_c \end{bmatrix}, \quad (3.4)$$

where  $x, y,$  and  $z$  are the coordinates of a hand voxel; and  $x', y',$  and  $z'$  are the coordinates in the voxel transformed by rotation matrices. The hand voxel is rotated by making the centroid of the hand the rotation center. By applying this rotation model to the 3D voxel model, virtual hand images that rotate in various directions are created from one frontal hand image.

To create the 2D depth image for input to the CNN, the virtually rotated shape of the hand in the 3D voxel model  $Voxel\_Hand(x, y, z)$  is mapped onto the  $xy$  plane. In this step, the distance of each  $xy$  coordinate in the voxel model nearest to the

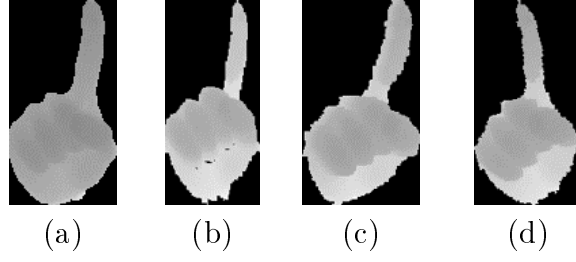


Figure 3.9: Examples of rotation results. (a) Original image, (b) result rotated by  $-15^\circ$  in the pitch direction, (c) result rotated by  $-15^\circ$  in the yaw direction, and (d) result rotated by  $-15^\circ$  in the roll direction

sensor is set to each pixel in the 2D pixel model  $Image\_Hand(x, y)$ . Thus, the 2D pixel model  $Image\_Hand$  has distance information of the surface of the rotated hand shape. Moreover, to normalize the distance information of the 2D pixel model, the distance of each pixel is normalized by

$$Image\_Hand(x, y) = Image\_Hand(x, y) - th, \quad (3.5)$$

$$th = \min \{ Image\_Hand(x, y) \} (x, y = 1, 2, \dots, N). \quad (3.6)$$

The virtually rotated results are stored as monochrome images as well and the examples are given in Figure 3.9. The most important point is that the depth data of the hand region obtained from the RGB-D sensor is merely the hand surface. When the voxel model is rotated with a large angle of rotation, some holes appear on the rotated image due to the influence of the hand thickness; Figure 3.10(b) shows the result of rotation if the thickness is not considered in the voxel model. The original depth image, Figure 3.10(a), is rotated  $-15^\circ$  on the yaw. In Figure 3.10(b), there are some holes without depth information because the proposed hand rotation method only rotates voxels on the surface of the hand.

To address this problem, the virtual thickness to the hand surface in the voxel model is append before the rotation. If the surface of the hand region exists in

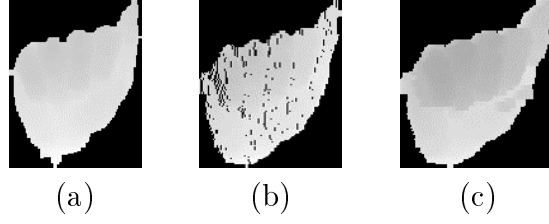


Figure 3.10: Example of hand augmentation with/without specifying the thickness. (a) Original image, (b) augmented depth images without specifying the thickness, and (c) augmented depth image with specification of the thickness

$Voxel\_Hand(x, y, z)$ , that is,  $Voxel\_Hand(x, y, z) = 1$  and the virtual thickness is added by

$$Voxel\_Hand_{th}(x, y, z + i) = 1 \quad (1 \leq i \leq \alpha), \quad (3.7)$$

where  $Voxel\_Hand_{th}$  is the voxel model with virtual thickness, and  $\alpha$  is the parameter of the thickness. Although the thickness of the hand depends on the hand shape and the part of the hand, a uniform thickness is added to each surface in order to realize a simple thickness model. In this research,  $\alpha$  is set to 11. This parameter is defined based on the average thickness of the Japanese little finger [126]. An example of the rotated depth image of Figure 3.10(a) is shown in Figure 3.10(c).

The new images created by this data augmentation increase the number of data in the dataset and provide more informative data by the variation appearances of original data. To obtain similar data manually would be certainly very exhausting. Moreover, regarding the application of CNN for classification, these results can surely make the CNN model more skillful by increasing the ability of the models to generalize what they have learned to the new images.

### 3.3.3 Hand Posture Classification

The CNN model based on deep learning is used for feature extraction and classification of the hand shape. Our CNN architecture is based on the Xception model, which



Table 3.1: Comparison of classification accuracy of some CNNs on ImageNet dataset

	Top-1 accuracy	Top-5 accuracy
VGG-16	0.715	0.901
ResNet-152	0.770	0.933
Inception V3	0.782	0.941
Xception	<b>0.790</b>	<b>0.945</b>

has replaced the Inception module with depthwise separable convolution (a spatial convolution performed independently for each channel) and is followed by a pointwise convolution (a  $1 \times 1$  convolution across channels). In a paper written by Chollet [107], the comparison of classification results from several CNN architecture are provided in Table 3.1.

The accuracy value in Table 3.1 is obtained in the experiment to compare Xception and Inception-V3 as benchmark using the same dataset and hyperparameters setting, while the accuracy value of VGG-16 and ResNet-152 is included as a note. The experiment was done by utilizing the ImageNet dataset, which is the image database organized in a hierarchical system to satisfy the data needs in the computer vision problem. Each node in the hierarchy of ImageNet is composed of a hundred even thousand images with quality-controlled and human-annotated. ImageNet dataset besides being widely used to benchmarking state-of-the-art model, it is also used to training resource, introducing new semantic relations for visual modeling, and researching human vision. The same exact hyperparameters are applied on both model such as optimizer is SGD, momentum is 0.9, initial learning rate is 0.045, and decay of rate is 0.94 every two epochs. On the contrary, Top-1 accuracy in the table depicts the conventional accuracy when the top class (with the highest probability) is same with target label and Top-5 accuracy means the target label (expected answer) is one of the five highest probability answers.

The Xception showed higher classification accuracy than previous architectures with the smaller parameter amount and training speed compared to Inception, as

Table 3.2: Comparison of size and training speed between Inception and Xception

	Parameter count	Steps/second
Inception V3	23,626,728	31
Xception	22,855,952	28

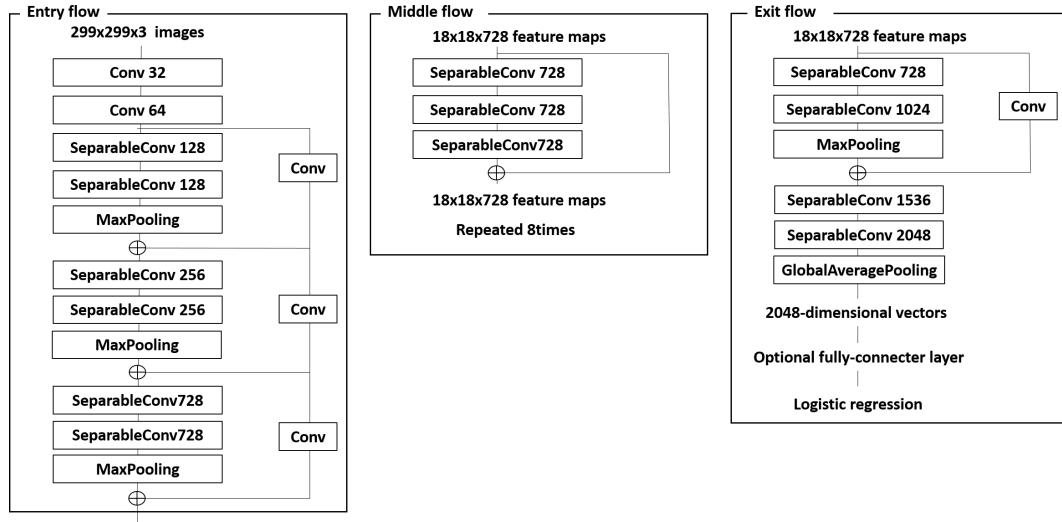


Figure 3.11: Architecture of the Xception model

shown in Table 3.2.

The original architecture of the Xception model is shown in Figure 3.11 [107]. This model accepts an RGB color image of  $299 \times 299$  pixels as the input image. The Xception model has 36 convolutional layers for feature extraction, which are framed in 14 modules. Each module has linear residual connections around them, except for the first and last modules. All the convolution and separable convolution in the model are followed by batch normalization, however they were not included in the diagram. Furthermore, all the separable convolution layers employed a depth multiplier of one (no depth expansion). The model consists of three flows, entry flow, middle flow, and exit flow. First, data enters the entry flow, then passes the middle flow repeated eight times, and finally, exits through the exit flow. In the exit flow, although Xception has the optionally fully connected layer and logistic regression, these layers are replaced to a global average pooling (GAP) layer.

GAP is an ordinary average pooling layer with a pool size equal to the size of the input. This method outputs one feature map for each appropriate category of the classification task in the last convolutional layer and directly feeds it to the soft-max layer. The soft-max function transforms the output of each unit to a value between 0 and 1. The output of the soft-max function is equivalent to a categorical probability distribution; it gives the probability that any of the classes are true. The soft-max activation function is given by the following equation:

$$f(x_i) = \frac{e^{x_i}}{\sum_{j=0}^k e^{x_j}}, \quad (i = 0, 1, 2, \dots, k), \quad (3.8)$$

where  $x$  is the vector input to the output layer; and  $i$  is the index of the output units.

A pre-trained network is a popular transfer learning approach to improve the model performance on a second task by transferring or fine-tuning the knowledge using the weights by continuing the backpropagation [127]. Generally, the ImageNet dataset is used for pre-training networks in CNNs.

The weights from the pre-trained network on ImageNet are relevant and useful for most classification tasks, because they can capture global features such as curves and edges in the early layers. Therefore, the weight pre-trained by ImageNet is used as the initial weights of our CNN.

## 3.4 Experimental Setup and Results of Hand Posture Classification

### 3.4.1 Experimental Setup

To show the effectiveness of the proposed hand posture classification model and data augmentation method, classification accuracies using five training datasets created from Kang's dataset are compared. As mentioned in Sec. 3.3, Kang's dataset includes

31 sign classes of hand depth data – 24 alpha and seven numeric. The dataset is collected from five subjects, and the number of images in each class is 200 or more.

Data augmentation generates more images of hand posture that can be used as input data for classification on CNN. In this data augmentation, the hand voxel rotated in all combination from  $-15^\circ$  to  $+15^\circ$  with a  $5^\circ$  interval in each direction. As a result, 343 images (including one original image from  $0^\circ-0^\circ-0^\circ$  combination of angle rotation) in various rotation directions are virtually generated from just a single hand image. As preprocessing for classification on Xception, all data resulting from data augmentation is resized according to the input size of the Xception model, which is  $299 \times 299$  pixel and then interpolated with the nearest neighbor interpolation method.

The details of the five datasets that are used for this experiment are shown in Table 3.3. First, three datasets (Dataset A, B, and C) containing solely original hand data are created. The original hand data means that the proposed data augmentation is not applied to the hand data that has been segmented from raw data. The difference between these datasets is the number of images for each class. They consist of 2, 20, and 200 images respectively for each sign of one subject. In contrast, by applying the proposed data augmentation with/without adding thickness to Dataset A (containing only 2 original images), two datasets (Dataset D and E.) are created.

I separate each dataset into training and test data. In each dataset, the four subject's images are used as training data, and the remaining subject images are used as test data. For appropriate assessment, although a different amount of data, depicted in Table 3.3, are used as training data, 6,200 images from one subject in Dataset C are used as test data. For example, the first accuracy from the classification result in Table 3.4 is obtained using images from Subject 2, 3, 4 and 5 from the Dataset A as training data and utilizing images from Subject 1 on the Dataset C as testing data. The second classification accuracy was obtained by applying images from Subject 1, 3, 4, and 5 on Dataset A as training data and images from Subject 2 on Dataset C as testing data.

Table 3.3: Details of datasets for experiments

Dataset ID	Total number of images	Number of subjects	Number of original images for each class captured from one subject	Number of augmented images for each class resulting from one subject
<b>Dataset A</b> Only original hand data	310	5	2	-
<b>Dataset B</b> Only original hand data	3,100	5	20	-
<b>Dataset C</b> Only original hand data	31,000	5	200	-
<b>Dataset D</b> Original hand data & virtually generated data without adding hand thickness	106,330	5	2	684
<b>Dataset E</b> Original hand data & virtually generated data with adding hand thickness	106,330	5	2	684

The adaptive gradient (AdaGrad) algorithm is used as an optimizer. The learning rate is set to 0.0001, the batch size is 20, and the number of epoch is 50.

### 3.4.2 Results and Discussions

The results of the accuracy for each test subject are shown in Table 3.4. In this table, test subject ID specifies the subject used for testing data. From the results of Dataset A, B, and C, I confirmed the importance of using a large amount of training data for classification using a CNN, because classification accuracy using a large number of images for training improved.

Table 3.4: Results of the proposed method for classification of ASL dataset

Dataset ID	Number of training images for one subject	Test subject ID (of dataset C)	Accuracy (%)	Average (%)
Dataset A	62	1	58.96	50.04
		2	62.09	
		3	45.16	
		4	40.24	
		5	43.75	
Dataset B	620	1	85.39	78.18
		2	89.12	
		3	73.64	
		4	68.53	
		5	74.20	
Dataset C	6,200	1	95.15	89.39
		2	91.85	
		3	91.74	
		4	86.16	
		5	82.06	
Dataset D	21,266	1	87.98	79.18
		2	82.15	
		3	70.73	
		4	79.97	
		5	75.06	
Dataset E	21,266	1	88.00	81.44
		2	86.18	
		3	74.21	
		4	80.00	
		5	78.83	

In contrast, Dataset D and E are constructed by applying the proposed data augmentation method to Dataset A. By comparing the accuracy between Datasets A, D, and E, the classification accuracy of Datasets D and E improved significantly. This improvement has occurred because the proposed data augmentation makes the appearance of hand signs more diverse. In addition, the accuracy of Dataset E is higher than that of Dataset D. The classification results of data augmentation by specifying the thickness yielded higher accuracy than without specification of the thickness, because virtually specified hand thickness can overcome the noise generated

by 3D rotation of the hand surface. Although the accuracy of Dataset C is best in all results, construction of this large dataset requires a lot of time and effort. The most important point is that Dataset D and E based on the proposed augmentation method achieved highly accurate classification using just two original images.

Two confusion matrices of subject 3 in Dataset B and E are shown in Figure 3.12. Using the proposed data augmentation method, several misclassifications are efficiently reduced in Dataset E. The effect is noticeable in sign classes ‘3’, ‘O’, and ‘Y’. As an example, Figure 3.13 represents the sign posture ‘3’ included in the test dataset and Dataset B and E. The hand posture ‘3’ in the test data has various appearances. In contrast, compared with Dataset E, which includes various appearances of sign posture ‘3’ by using the proposed data augmentation and adding thickness, Dataset B has a less diverse appearance. Therefore, various posture appearances generated by the proposed method are sufficient to accurately classify a hand posture, which is a complex articulated object consisting of many connected parts and joints.

The general cause of misclassification is the similarity between the sign postures. The misclassification can be seen in Figure 3.14, which shows the similarity of sign posture ‘D’ and ‘1’ on each Dataset B and E. Augmentation also shows a small impact in the misclassification that occurred. Misclassification in some classes in Dataset B is lower than for Dataset E. Figure 3.14 also shows that the proposed method forcibly generates more various appearances of sign ‘D’ in Dataset E compare with Dataset B. However, the appearance of sign posture ‘D’ in the original dataset (test data) is not diverse. It makes the test data of sign ‘D’ more recognizable as a sign ‘1’ than as a sign ‘D’. Although some misclassifications occur by the effect of data augmentation, overall, this can improve accuracy. I confirmed the effectiveness of the proposed method for classification of hand postures captured from an RGB-D sensor.

The effectiveness of the CNN model with Xception architecture and the fine-tuning technique for hand posture classification are also showed. This research using the Xception model (Dataset C) showed better performance around 3.9% compared with

Table 3.5: Comparison of results in the classification accuracy based on the method

Method		Number of training images for one subject	Average (%)
Dataset C	Proposed method (Xception)	6,200	89.39
	Previous research (similar to CaffeNet)	6,200	85.49

the previous research using a similar architecture with CaffeNet by Kang, as shown in Table 3.5. In both studies, the same number of datasets (31,000 images) were trained, and both had the same ratio of separation for training and test. For training and classification of the posture of the hand for the CNN, the same fine-tuning technique was applied in both studies. As a result, I confirmed the effectiveness of the proposed method utilizing the Xception model for hand posture classification.

### 3.5 Conclusions

In this study, a method for effective hand posture classification using data augmentation and a state-of-the-art CNN model was proposed. The proposed augmentation method using 3D rotation on the depth data can be applied to increase the number of image datasets, because of the difficulty of collecting hand datasets from several subjects and showing the efficacy by increasing the classification accuracy for the same original images in a CNN. This research also demonstrated higher performance for classification using the Xception model. The availability of a sufficient number of datasets and a better classification method can accelerate the development of a sign language recognition system as a communication interpreter using deep learning.





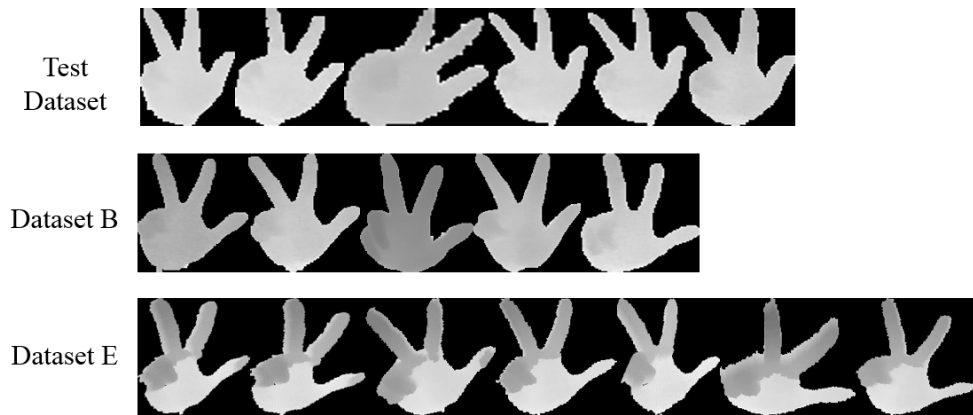


Figure 3.13: Appearance of the '3' sign posture in the dataset

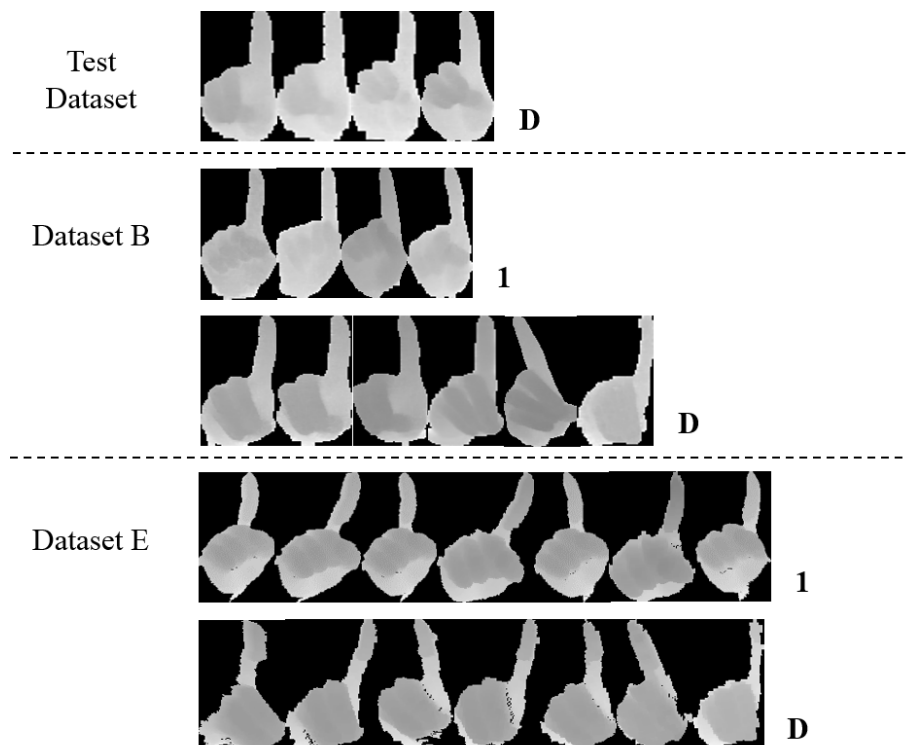


Figure 3.14: Appearance of 'D' and '1' sign postures in the dataset

# Chapter 4

## Food Constituent Estimation for Lifestyle Disease Prevention by Multi-task CNN

### 4.1 Background

Progress in distribution systems and food production technology in recent years enabled people to obtain their preferred food at any time quickly. However, consumption of retort food and opportunities for dining out are increasing as more people lead solitary lives and the dual income ratio increases, along with the expansion of the diversity of lifestyle habits in developed countries. Maintaining such eating habits increases the risk of lifestyle diseases owing to biased nutritional balance. Lifestyle-related diseases include hypertension, dyslipidemia, diabetes, and others. These diseases can progress undiagnosed to damage the brain, heart, blood, vessels, etc., as they have few subjective symptoms.

One way to reduce the risk due to lifestyle-disease is measuring the amount of food intake consumed. Calculation of the amount of calories is frequently used in the diet, as calories are a very influential factor in obesity, which is one of the leading

causes of numerous diseases, such as heart disease, hypertension, and diabetes [128]. The accurate estimation of the calorie content of food is effective to prevent obesity. In similar condition, salt is likewise an essential nutrient in the human body even though it is only needed in relatively small quantities. Salt does not affect the number of calories; however, health risks such as high blood pressure can increase with high salinity intake.

As a traditional approach, people proceed to diet by manually recording and counting the intake of calories from the food that enter the body. However, that is tedious and inefficient, sometimes even requires relying on memory, and less effective owing to the lack of nutritional information [128, 129]. The growing need for more specific and accurate diet assessment methods gave rise to efforts to improve these methods, including integrating them with technology [130]. Further, the number of scattered food images on the internet supports interest in the presence of automatic food recognition systems using images. Hence, several automatic food recognition systems using food images have been developed to record everyday foods with the aim of increasing the extent to which people are conscious of developing eating habits for good health [131, 132, 133, 134].

The advancement and portability of mobile phones make mobile applications as an optional device of the practical food recognition system. These applications are especially useful for dietary assessment and planning. Automatic food image recognition methods have been widely proposed to enhance the capability of an application. Automatic food image classification could potentially alleviate the process of food-intake estimation and dietary assessment. In most of the cases, we can easily estimate food materials and ingredients based on the food category classified from the food image captured by a mobile device. However, the estimated values are only standard guidelines for each category, because food materials and ingredients are not unique to the captured food. In other words, the outputs would be standard values according to the category classification result.

It is possible to roughly estimate how high the calorie content of the food in an image based on prior knowledge, although humans would find it difficult to estimate the precise calorie content from a food image. Previous studies proposed methods capable of directly estimating the calorie content of food from an image in an attempt to implement this human ability on a mobile device. Additionally, it is clear that estimating the salinity of food from an image is a problem that is a more difficult for humans, and to date no studies have focused on the automatic estimation of food salinity from food image.

In this research, an automatic method for food ingredient estimation from a food image using a multi-task CNN is proposed. Intending to achieve lifestyle disease prevention, this research focuses on the recognition of the food category and the estimation of calories and salinity. CNN based on deep learning has been demonstrated to achieve excellent results for image classification and object detection. The effective estimation of calories and salinity using multi-task learning with food category classification is realized by defining both calorie and salinity estimation as a regression problem. The underlying assumption for multi-task learning algorithms is that different tasks are related to each other [135]. Although a previous study reported that a food category is closely related to its calorie content, research to clarify the relationship between the food category and salinity has not yet been reported. In this research, the relationship between the food category and salinity is also experimentally shown by using multi-task CNN.

The Xception model [107], which has achieved a high recognition rate in image classification tasks using ImageNet, is used as the basis for the proposed architecture. In addition, new food image dataset is constructed by using public images from several recipe-gathering websites because there is no large food image dataset with detail information on calorie and salinity. Here, CNN is considered to require a large number of training images to achieve comparable or superior performance to the conventional local-feature-based methods. The two-stage transfer learning using a

large number of food image databases for food category classification is proposed because it is difficult to collect many food images whose calorie content and salinity are known.

## 4.2 Related Works

Chronic diseases caused by obesity requires a proper diet in an effort of its prevention. Often, people who diet should note their daily meals regularly, both for self-monitoring and to acquire useful statistics for dietitians. The recording of the calorie intake developed from paper-based to the use of information and communication technology such as the website [136, 137], computer application [138], and subsequently inspired the presence of a food diary application for mobile phones with the convenience and practicality of recording as a goal [139, 140]. Manually journaling is quite tiring, so image processing is offered to facilitate the assessment of the amount of food nutrition through the website [133] and mobile applications that are more preferred [141].

Image processing is utilized in food image recognition that enables nutrient estimation and health-care analysis corresponding to people's eating habits. Therefore, food image recognition methods and food image databases have been widely developed. Food recognition methods likewise can be viewed from two feature extraction approaches that are: the handcrafted feature-based and deep learning-based approaches.

First, I describe the handcrafted feature-based approach. Many researchers have attempted to solve the problem of food image classification by using simple low-level feature extraction and coding methods. Yang *et al.* proposed a method to analyze the ingredient relations in the food image by computing pairwise statistics between the local features [142]. First for feature extraction is to obtain pixel-wise soft labels for each pixel in the images, the probability with which a pixel belongs to each food category is calculated using the semantic texton forests [143]. The soft-

Pairwise Feature	Feature Description	Expression	Fig.
Distance	The distance between two pixels $P_1$ and $P_2$	$D(P_1, P_2)$	(b)
Orientation	The orientation of the line between two pixels $P_1$ and $P_2$	$O(P_1, P_2)$	(c)
Midpoint category	The type of the pixel in the middle of two pixels $P_1$ and $P_2$	$M(P_1, P_2)$	(d)
Between-pair category	The type of all pixels between two pixels $P_1$ and $P_2$	$B(P_1, P_2)$	(e)

(a)

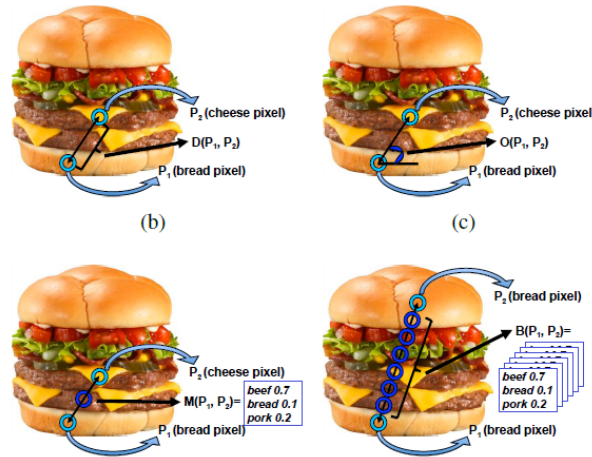


Figure 4.1: Four feature pairwise. (a) Information table of pairwise feature, (b) (c) (d) (e) are illustrations of the pairwise feature [142]

labels picture is utilized to create the histogram of global ingredient representation, however the histogram cannot capture the spatial relationship between ingredient for distinguishing one food from another. The pairwise feature is used to capture spatial relationship and is employed to extract statistics of pairwise local features, to form a multi-dimensional histogram. Figure 4.1 represents more information about the pairwise feature. Finally, the food image is classified by applying the obtained histogram to a multi-class SVM using a  $X^2$  kernel.

Kong and Tan proposed “DietCam”, which is an automatic camera-phonned-based multi-view food classifiers as part of a food intake assessment system [144]. DietCam detects food ingredients by using a deformable part-based model and a texture verification model. A food category is classified by using the detected ingredients and a multi-view kernel SVM. He *et al.* proposed an image segmentation and classification

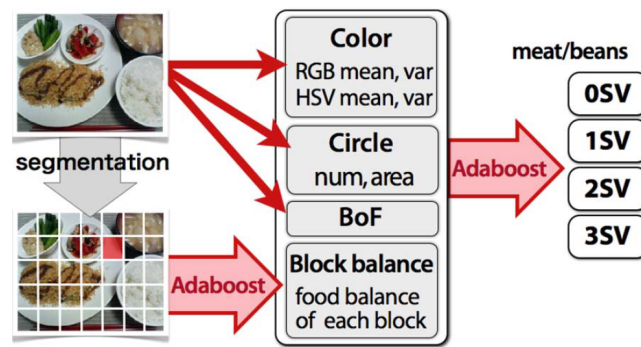


Figure 4.2: Food detection and food-balance estimation method using the global feature [146]

method to detect the food regions in an image, and to classify the food category [145]. They extracted the color with four color descriptors and the texture with three texture descriptors for food classification. In addition, this method estimated the weight of food to extract the nutrient content from a food image using a shape template for foods with regular shapes and area-based weight estimation for foods with irregular shapes. Aizawa *et al.* developed a food detection and food-balance estimation method [146]. First, food-image detection categorizes the image into “food” or “non-food” by employing supervised learning based on multiple image features and SVM, then, food-balance estimation is used to estimate the food balance of the meal shown in each photograph by using the global color, circle, BoF, and block features, which are shown in Figure 4.2. All features merged into a feature vector and food-balance each food category were estimated by classifying the vector into one or more classes by AdaBoost. They proposed improving the performance with personal likelihood.

Anthimopoulos *et al.* offer a BoF-based system for food image classification. First, a visual dictionary of 10,000 visual words is created from dense local features based on SIFT features on the HSV color space [147]. The results of food image recognition are shown by comparing three machine learning-based classifiers such as SVM, ANN, and random forests (RFs).

Considering all of the previously described methods, it seems the best approach is to use a complex combination of a large number of image features. Some food



types have a high intra-class and low inter-class variance, since foods are typically deformable objects. Even though researchers have addressed the problem of developing the food feature, these characteristics still lead to the low classification accuracy on handcrafted feature-based approaches.

During the past years, CNN has become the most effective architecture to perform visual recognition. The ability of CNN to extract the feature automatically is expected to alleviate the feature extraction and gain better results on food recognition. Hence, many studies using the deep learning-based approach for food recognition have been reported [148, 149, 150, 151, 152, 153]. These researchers demonstrated that it is possible to classify with higher precision than of the existing handcrafted feature-based approach by conducting experiments using food image datasets. Myers *et al.* reported a deep learning-based approach to recognize the food item and predict the nutritional content of the food [154]. They propose the Im2Calories system for food recognition, which made extensive use of CNN, and utilized the architecture of GoogleNet [105], fine-tuning the pre-trained model on Food101 [155]. Pouladzadeh *et al.* proposed a method to classify the food and to estimate the calories contained in images of food taken by the user [156]. These works integrate with the mobile calorie measurement application and require reference information to estimate the quantity of food on the plate.

Multi-task learning has been used successfully in several applications of machine learning, including computer vision [157]. Multi-task learning on CNN aims to improve generalization performance by completing multiple related tasks at the same time. Abdalnabi *et al.* proposed a joint multi-task learning algorithm to effectively estimate several image attributes using CNN [135]. In this multi-task CNN, effective learning of CNN is achieved because attributes in the same group are prompted to share more knowledge, whereas attributes in different groups generally compete with each other. By using multi-task CNN, Chen and Ngo achieved food categorization by recognizing the ingredients composition from food images [158]. Their paper reported

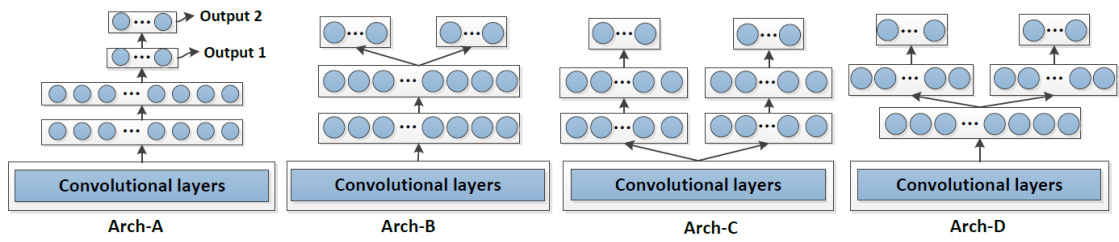


Figure 4.3: Four different architectures of deep CNN for multi-task learning of food category and ingredient recognition [158]

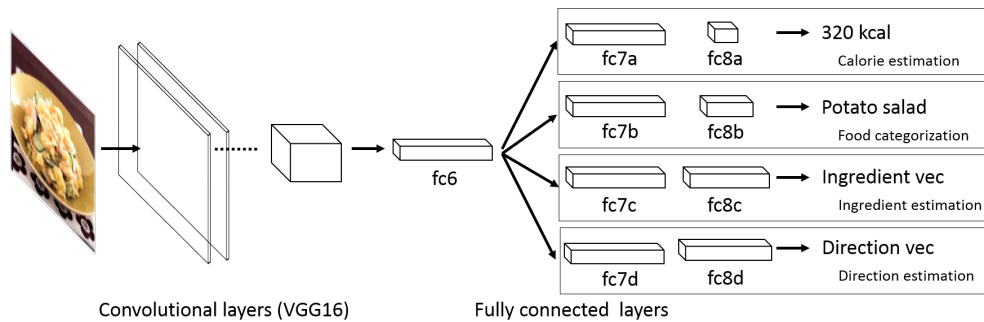


Figure 4.4: Multi-task CNN based on VGG-16 architecture

that simultaneous estimation boosted the estimation performance in both tasks, ingredient recognition and food categorization. They considered four architectures of deep CNN as well, which shown in Figure 4.3, and whose best performance is depicted by Arch-D.

Ege and Yanai proposed a simultaneous estimation method of food category and calories with multi-task CNN [159]. This research implemented multi-task CNN based on VGG16, and multi-task indicated from the seventh layer of the fully connected layer, which branches to each task. The model of multi-task CNN is expressed in Figure 4.4. Although the accuracies of both tasks were improved compared with single-task CNN, the improvement is slight due to the number of datasets for training the multi-task CNN is insufficient.

Although many studies on category identification and calorie estimation have been published, the estimation of salinity from a food image has not yet been reported. In addition, a large training dataset is required to derive the performance of multi-task

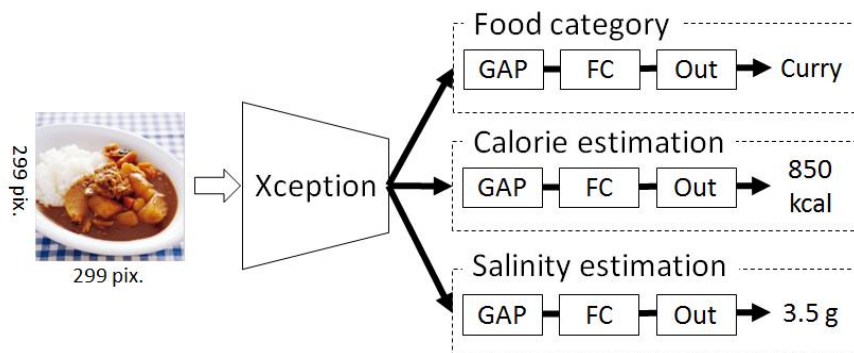


Figure 4.5: Architecture of proposed multi-task CNN

CNN even though it is effective to use multi-task CNN with category classification to estimate food ingredients. However, it is extremely costly to collect ingredient-annotated data of food images.

### 4.3 Proposed Method

This research aim is to more accurately estimate the food category and ingredients from a food image by a multi-task CNN. Excess calorie and salt intake pose strong health risks such as heart disease, hypertension, and high blood pressure. However, no research has reported the salinity estimation. This led us to focus on the estimation of the calorie content and salinity of food.

The architecture of the multi-task CNN is shown in Figure 4.5. The proposed architecture is based on the Xception model with the applied hard parameter sharing concept. After the feature extraction by Xception, the model branches into three tasks: food category classification, calorie estimation, and salinity estimation. Each task arranged by the global average pooling layer, the fully connected layer, and the output layer, which are denoted as GAP, FC, and Out in Figure 4.5.

A food ingredient-annotated food image dataset including both calorie content and salinity does not yet exist because other researchers have not focused on salinity estimation. In this study, a food image dataset annotated by calories and salin-

ity is constructed. In contrast, effective and efficient training using multi-task CNN is achieved by continuously fine-tuning the multi-task CNN by using a small-scale ingredient-annotated dataset and a middle-scale dataset with only the categories annotated, as a two-stage fine-tuning procedure.

### 4.3.1 Dataset Construction

A large food image dataset annotated by calories and salinity is not available to date. A large number of food images annotated with both types of information are collected from six commercial cooking recipe sites on the web [160, 161, 162, 163, 164, 165]. The recipe information published on these sites was provided by experts such as cooks and cooking researchers. Although the calorie and the salinity information provided on these recipe sites are for one person, the amounts specified for some food images are for multiple people. These food images for multiple people are excluded from the dataset since the focus is on estimating the calorie and salinity content from a food image for one person.

As a category, the proposed method focus on the representative 14 categories included in the UEC Food-100 [166], which are shown in Figure 4.6. The UEC Food-100 contains food images of 100 kinds of Japanese foods. The 14 selected categories are included in the food category considered in the research by Ege and Yanai [159].

Here, the low-resolution images or those with multiple category labels from the collected images are excluded. All images were resized to  $299 \times 299$  pixels. As a result of the aforementioned processing, a total of 3,051 images were collected on 14 categories. Details amount of the food image dataset together with the mean and standard deviation of their calorie and salinity are shown in Table 4.1. The distribution of calorie and salinity of the entire dataset is shown in Figure 4.7 and Figure 4.8 .

I collected another dataset, a category-annotated food images dataset from the Rakuten recipe dataset, which is part of the Rakuten dataset [167]. This dataset has



Figure 4.6: Representations of food images for each collected category

14 categories, which is the same as the dataset for the main task. The collected image limit for each category is 3,000. Here, low-resolution images or those with multiple category labels from the collected images are excluded. All images were resized to  $299 \times 299$  pixels. As a result of the above processing, a total of 28,359 images were collected in 14 categories. Note that calorie and salinity are not annotated in these images dataset. The number of collected images for each category is shown in Table 4.2.

### 4.3.2 Architecture of Multi-task CNN

The proposed architecture, which is shown in Figure 4.5, is based on the Xception model. Xception is a CNN architecture inspired by Inception. This architecture displayed a higher efficiency because, with the same number of parameters as Inception, Xception significantly outperforms Inception V3 on a larger image classification dataset comprising 350 million images and 17,000 classes. The details of the Xcep-

Table 4.1: Details of collected ingredient-annotated food images for multi-task CNN

Category	Number of images	Calorie (kcal)		Salinity (g)	
		Ave.	S.D.	Ave.	S.D.
Curry and rice	214	531.14	220.57	2.630	1.045
Fried rice	217	469.83	165.32	2.310	0.995
Chow mein	140	552.62	118.86	3.202	0.921
Spaghetti	565	573.50	116.98	2.656	0.868
Gratin	264	397.99	160.43	1.985	0.836
Miso soup	373	84.65	52.20	1.946	0.643
Stew	136	379.53	119.70	2.331	0.914
Beef and potato stew	154	378.24	126.35	2.357	1.026
Hamburg steak	226	395.57	108.60	2.254	0.823
Cold tofu	114	141.56	57.30	1.235	0.594
Scattered sushi	107	534.91	129.93	2.961	1.122
Omurice	105	683.63	132.44	2.917	0.776
Potato salad	210	230.40	93.49	1.205	0.583
Mixed rice	226	409.65	97.20	1.814	0.751
Whole	3,051	416.65	211.90	2.280	0.999

tion architecture has been shown in Figure 3.11 of section 3.3.3. Figure 4.9 shows the global average-pooling layer, the optional fully connected layer, and the logistic regression at exit flow are excluded from the original Xception architecture. This architecture’s layers are used as the common layers for all tasks.

A multi-task learning mechanism makes it possible to transfer shared knowledge across multiple prediction tasks so that only task-specific features need to be learned [168, 169] and utilize valuable information conceived in several related tasks to increase the ability to generalize all tasks [170]. The CNN model learns each prediction task simultaneously through the shared representation [171]. The two most commonly used approaches to performing multi-task learning in deep neural networks are hard parameter sharing, which applied by sharing the hidden layers between all tasks while keeping several task-specific output layers, and soft parameter sharing, where each task has its model with its own parameters [172]. Figure 4.10 represent hard parameter sharing and soft parameter sharing.

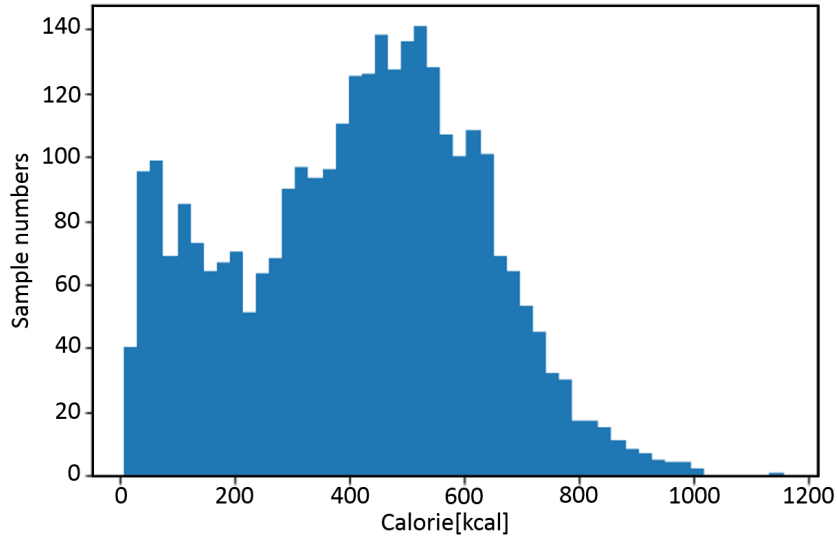


Figure 4.7: Distribution of calorie content throughout the dataset

Table 4.2: Details of collected category-annotated food images for two-stage fine-tuning

Category	Number of images	Category	Number of images
Curry and rice	2,787	Fried rice	2,999
Chow mein	2,883	Spaghetti	3,000
Gratin	2,095	Miso soup	3,000
Stew	1,619	Beef and potato stew	736
Hamburg steak	2,500	Cold tofu	536
Scattered sushi	612	Omurice	908
Potato salad	1,684	Mixed rice	3,000

The network of Xception model in this research branches to each task from the global average-pooling layer. Each task has a global average pooling layer, a fully connected layer with a dropout, and an output layer, respectively. The branched networks are adjusted by specializing in different tasks, namely the classification and regression tasks. The food classification task has a fully connected layer with 512 dimensions and an output layer corresponding to each food category. The calorie and salinity estimation task comprises a fully connected layer with 512 dimensions and an output layer composed of one unit, respectively.

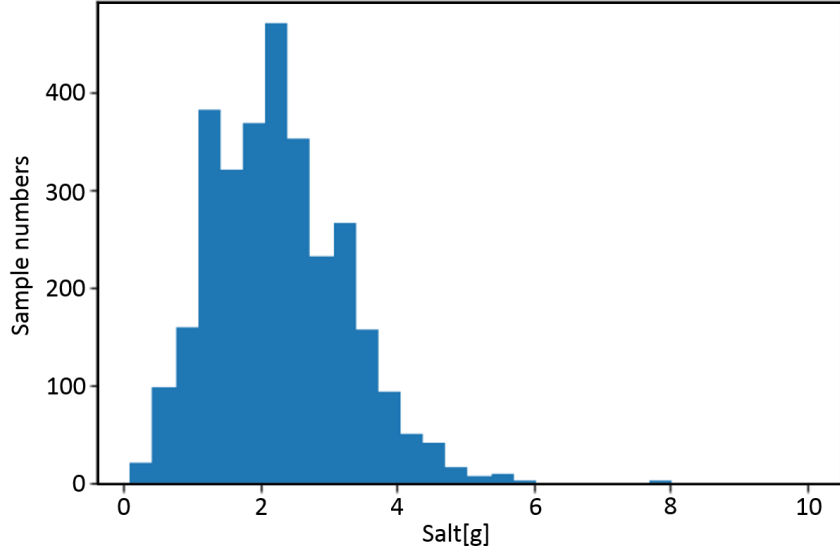


Figure 4.8: Distribution of salinity content across the dataset

The proposed multi-task CNN is trained based on the following loss function  $L$ .

$$L = \frac{1}{N} \sum_{i=0}^N (w_{cat}L_{cat}^i + w_{cal}L_{cal}^i + w_{sal}L_{sal}^i), \quad (4.1)$$

where  $L_{cat}$ ,  $L_{cal}$ , and  $L_{sal}$  are the loss functions of the food classification, food calorie estimation, and salinity estimation tasks, respectively. Further,  $w_{\alpha}$  ( $\alpha = \{cat, cal, sal\}$ ) are the weight coefficients for each loss function, so as to balance the value scales of the three loss function.  $N$  is the number of image samples.

A soft-max function is used for the output layer of the food classification task, since this task is a multi-class classification problem.  $L_{cat}$  is defined based on the standard soft-max cross-entropy.

$$L_{cat} = - \sum_{i=0}^M t_i \log y_i, \quad (4.2)$$

where  $t_i$  represents the ground-truth of the  $i$ th unit, which is binary,  $y_i$  is the output of the  $i$ th unit, and  $M$  is the number of food categories.

The food calorie task and salinity estimation task are treated as regression prob-



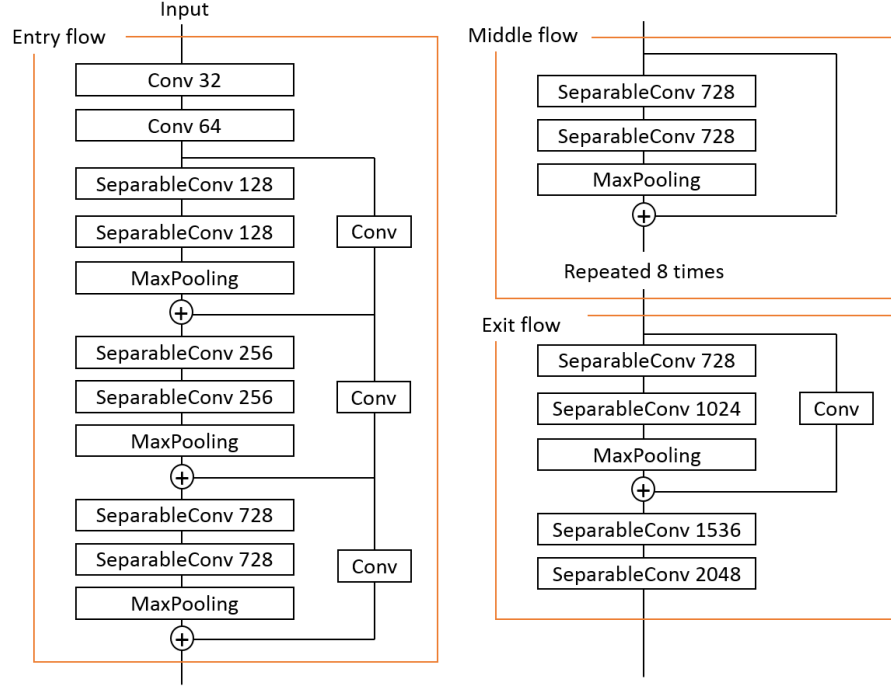


Figure 4.9: Xception architecture in multi-task CNN

lem. Although the mean square error is generally used as the loss function in a regression problem, the loss function proposed by Ege and Yanai [159] is used in the proposed method.

$$L_{cal} = w_{cal,re}E_{cal,re} + w_{cal,ab}E_{cal,ab}, \quad (4.3)$$

$$L_{sal} = w_{sal,re}E_{sal,re} + w_{sal,ab}E_{sal,ab}, \quad (4.4)$$

where each  $w$  is the weight coefficient so as to balance the value scales of these errors. Each of the errors is defined by

$$E_{cal,ab} = |y_{cal} - g_{cal}|, \quad (4.5)$$

$$E_{cal,re} = |y_{cal} - g_{cal}|/g_{cal}, \quad (4.6)$$

$$E_{sal,ab} = |y_{sal} - g_{sal}|, \quad (4.7)$$

$$E_{sal,re} = |y_{sal} - g_{sal}|/g_{sal}, \quad (4.8)$$

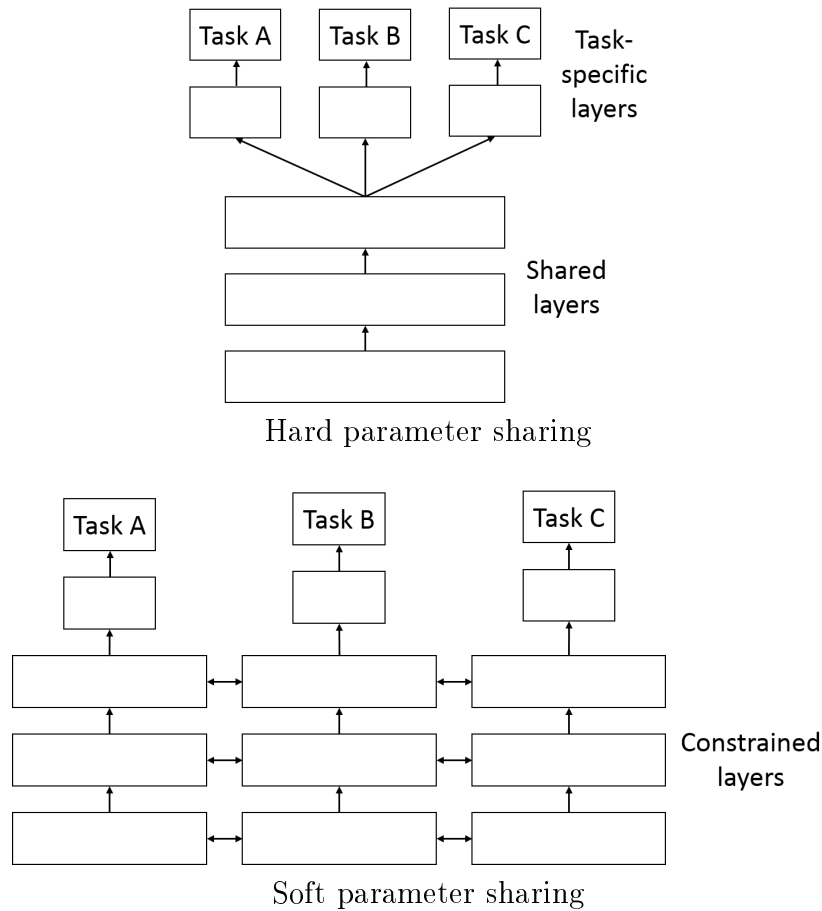


Figure 4.10: Hard parameter sharing and soft parameter sharing structures on multi-task learning [172]

where  $y$  and  $g$  of each error are the estimated value and the ground truth, respectively. The relative error  $E_{(\star, re)}$  is the ratio of the absolute error to the ground truth. The absolute error  $E_{(\star, ab)}$  is the absolute value of the difference between the estimated value and the ground-truth.

### 4.3.3 Two-stage Fine-tuning

When applied to areas where large-scale data are much more difficult to gather, CNN has still proven effective through the use of transfer learning [173]. Pre-trained CNNs are used as weight initializers for fine-tuning to the new target task. In general, the ImageNet dataset, which contains 50,000,000 images with 1,000 categories, is

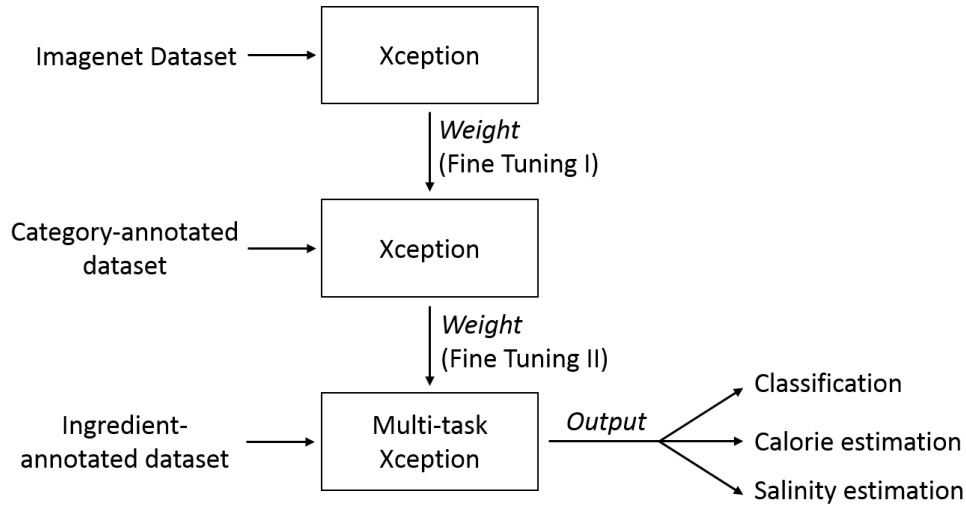


Figure 4.11: Two-stage fine-tuning using Xception on collected food images dataset

used for pre-training the CNN to solve various tasks. ImageNet is not necessarily strongly related to the food ingredient estimation task in this study, because it is a dataset for generic object recognition. When fine-tuning a multi-task CNN for food category classification and food ingredient estimation using the pre-trained model with ImageNet, the previous study reports clearly that the accuracy improvement is slight if the number of image per class is not large.

The proposed method address this issue by focusing on two-stage fine-tuning. First, a category-annotated food image dataset is prepared. The number of images in this dataset is larger than the dataset consisting of collected images with both ingredients annotated for the main task, however it is smaller than ImageNet. Subsequently, CNN pre-trained by ImageNet is once more fine-tuned for food category classification by using only the category-annotated food image dataset that consists of 28,359 images. Finally, the multi-task CNN model is fine-tuned again for food category classification and food ingredient estimation using the main task dataset, an ingredient-annotated dataset that contains 3,051 images, as the second stage of the fine-tuning process. Figure 4.11 explains the two-stage fine-tuning process on Xception utilizing the provided dataset.

## 4.4 Experimental Setup and Results of Food Constituent Estimation

### 4.4.1 Experimental Setup

In this study, two food image datasets, an ingredient-annotated dataset and a category-only-annotated dataset for two-stage fine-tuning, are constructed. In the ingredient-annotated dataset, 90% of the images in each category were used for training the CNN, and the remaining images were used for testing. In the category-only-annotated dataset, 80% of the images in each category was utilized to train the CNN, and the remainder was used for testing.

The performance of the proposed multi-task CNN with two-stage fine-tuning are evaluated. I demonstrated the effectiveness of multi-task learning and two-stage fine-tuning, respectively, by comparing three methods: single-task CNN, multi-task CNN without two-stage fine-tuning, and multi-task CNN with two-stage fine-tuning. The ImageNet dataset fine-tunes all models. The original Xception architecture is used for the single-task CNN. Single-task CNN is optimized for every task: food image classification, calorie estimation, and salinity estimation, each using the ingredient-annotated dataset. Multi-task CNN is implemented by utilizing Xception architecture as well with branching after the feature extraction layer so that the classification process can be carried out simultaneously with the estimation process. The experiment was done by applying the ingredient-annotated dataset. While on the multi-task CNN with two-stage fine-tuning, a category-annotated dataset is also used for fine-tuning before the main task was accomplished by using ingredient-annotated dataset. Table 4.3 shows the details set up for three task experiments on each model.

The initial learning rate is set to 0.001 and gradually decreases to 0.0001. All of the models were trained by Adagrad in 100 epochs. The dropout rate for each task is set to 0.5. The batch size is set to 16. I experimentally set the weight for each loss

Table 4.3: Detailed setup for the experiments of the tasks

Model	Task	Dataset		Xception	
		Only Category-annotated	Category and Ingredient-annotated	Without branches (Single-task)	With branches (Multi-task)
Single-task	Category class.	-	✓	✓	-
	Calorie est.	-	✓	✓	-
	Salinity est.	-	✓	✓	-
Multi-task CNN	All	-	✓	-	✓
Multi-task CNN with 2 stage of fine tuning	2 <sup>nd</sup> Fine tuning	✓	-	-	✓
	All	-	✓	-	✓

function to  $w_{cat} = 1.0$ ,  $w_{cal} = 0.4$ ,  $w_{sal} = 0.9$ , respectively. I also set the weight for each loss function of each task to  $w_{cal,re} = 1.0$ ,  $w_{cal,ab} = 0.3$ ,  $w_{sal,re} = 1.0$ ,  $w_{sal,ab} = 0.5$ , respectively.

The classification accuracy was used as the criterion for evaluating the food image classification. The absolute error  $E_{ab}$ , relative error  $E_{re}$ , and correlation coefficients were also used as criteria for calorie and salinity estimation.

## 4.4.2 Results and Discussions

### a. Category classification result

The results of the three tasks from all methods can be seen in Table 4.4. On the category classification result as the first task, the accuracy of the proposed multi-task CNN is higher than the single-task CNN and the accuracy of the multi-task CNN with two-stage fine-tuning is the highest. Figure 4.12 presented the confusion matrix for single-task and multi-task. The label numbers correspond to each category. Some misclassifications were shown on the single-task CNN, however they were reduced in multi-task CNN with and without two-stage fine-tuning. These misclassifications

Table 4.4: Comparison of results of each task

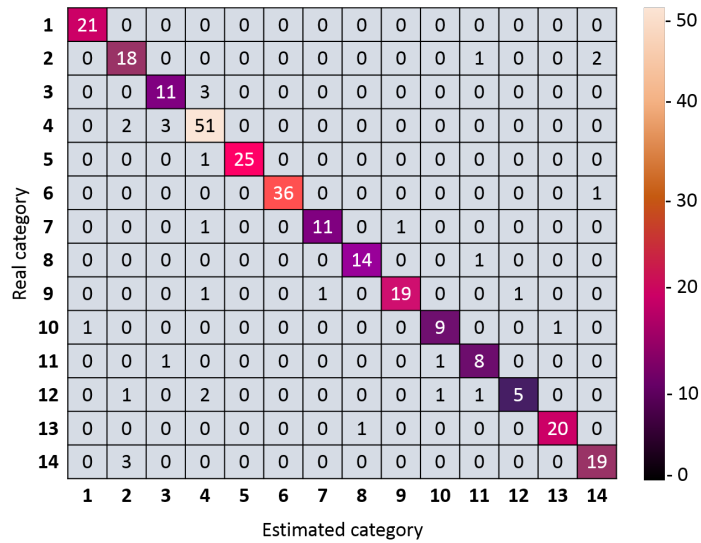
Methods	Calorie			Salinity			Category
	$E_{ab}$ (kcal)	$E_{re}$ (%)	CC	$E_{ab}$ (g)	$E_{re}$ (%)	CC	Accuracy (%)
Single-task CNN	100.2	41.7	0.80	0.75	37.2	0.40	90.0
Multi-task CNN	94.6	36.6	0.82	0.76	36.8	0.43	91.0
Multi-task CNN with two-stage fine-tuning	<b>89.6</b>	<b>31.2</b>	<b>0.84</b>	<b>0.74</b>	<b>36.1</b>	<b>0.45</b>	<b>92.6</b>

occurred between two categories in single-task: ‘Fried rice’ and ‘Mixed rice’, ‘Chow mein’ and ‘Spaghetti’. These food categories mostly contain the same basic ingredient: rice and noodles, making them difficult to be distinguished and causing misclassification. However, misclassification between ‘Fried rice’ and ‘Mixed rice’ is improved in multi-task CNN with and without two-stage fine-tuning.

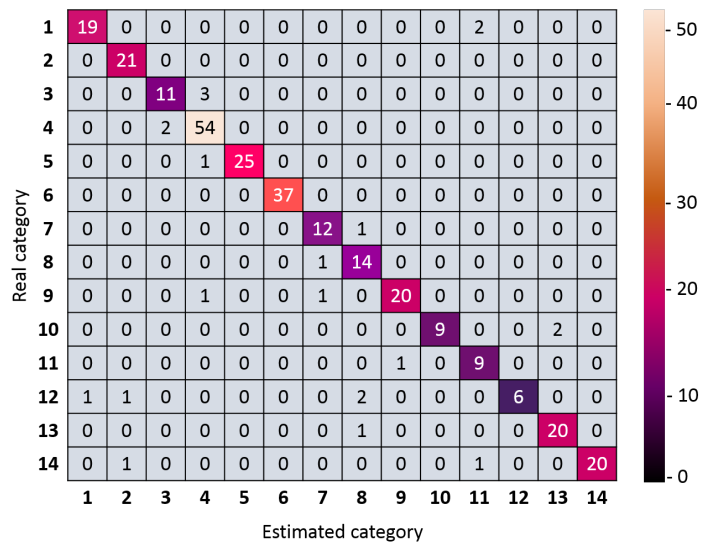
### b. Ingredient estimation

The result of the ingredient estimation as the next task showed a significant improvement of calorie estimation on the multi-task CNN with two-stage fine-tuning over the single-task CNN. In contrast, the salinity estimation with the multi-task CNN with two-stage fine-tuning relative to the single-task CNN showed slight improvement. Table 4.1 shows that the ratio of intra-class variance to the interclass variance of the salt content is larger than that of the calories. Seemingly, the task of estimating the salinity is more difficult compared to that of estimating the calories. Even in such a situation, both the errors and correlation coefficient of the salinity estimation were improved in these methods. I demonstrated that the proposed multi-task CNN with two-stage fine-tuning has correctly classified the food category and estimated the calories and salinity. From these results, I confirmed that the multi-task CNN is superior to the single-task CNN, and the multi-task CNN with two-stage fine-tuning outperforms the multi-task CNN without two-stage fine-tuning.

The multi-task CNN is advantageous to effectively and efficiently learn the com-



(a) Single-task



(b) Multi-task

Figure 4.12: Confusion matrix from single-task and multi-task on food category classification

mon features that contribute to each task if tasks are strongly correlated. In addition, the CNN is fine-tuned to enable the extraction of useful features for nutrient estimation in two-stage fine-tuning using a large number of images for category classification. Therefore, the difference in improvement between calorie and salinity estimation is

assumed to be attributable to the correlation difference between the food category classification task and the estimation of each ingredient. As a result, it is assumed that the correlation between category classification and salinity estimation to be weaker than calorie estimation, from the point of view of an image feature.

The relations between the results of category classification and nutrient estimation are analyzed from the experimental results. I initially categorized the result of calorie and salinity quantity estimation into success and failure cases. The successful cases of calorie and salinity quantity estimation as shown in Figure 4.13 and Figure 4.14, indicates that the category classification was completed and the difference (error) between the estimation result and the ground-truth of calorie or salinity quantity is relatively small. Meanwhile, the failure case is the state where the obtained error value is relatively large except the misclassification. In this failure case, the significant error is influenced by the result of the incorrect category classification result and also caused by the ground-truth calorie/salinity quantity, which is too large. In particular for salinity estimation, the large error acquired due to the influence of the ground-truth value is greater than the obtained error on calorie estimation. This is likely due to the ground truth value being too far from the mean data. Furthermore, there are the difficulties to estimate the salinity quantity on ‘Omurice’. The difficulties arose since the various appearances in serving the food makes the ‘Omurice’ difficult to classify, which will affect the estimation result.

Detailed results of the calorie and salinity estimation for each category are listed in Table 4.5 and 4.6. In the calorie estimation, the error in categories with a large mean and large intra-class variance is large. In contrast, it was difficult to find some correlation in the salinity estimation results. However, I confirmed the effectiveness of the proposed method since each absolute estimated error is smaller than the standard deviation in each category.









	<i>Successful case</i>			<i>Failure case</i>			
							
Ground-truth category	Curry Rice	Miso Soup	Hamburg Steak	Ground-truth category	Gratin	Spaghetti	Spaghetti
Ground-truth calorie	690 kcal	72 kcal	344 kcal	Ground-truth calorie	324 kcal	903 kcal	1157 kcal
Estimation category	Curry Rice	Miso Soup	Hamburg Steak	Estimation category	Spaghetti	Chow Mein	Spaghetti
Estimation calorie	674 kcal	62 kcal	350 kcal	Estimation calorie	551 kcal	569 kcal	604 kcal
Error	-16 kcal	-9 kcal	+6 kcal	Error	+227 kcal	-334 kcal	-553 kcal

Figure 4.13: Successful and failed cases of calorie estimation results







	<i>Successful case</i>			<i>Failure case</i>			
							
Ground-truth category	Spaghetti	Gratin	Stew	Ground-truth category	Fried rice	Spaghetti	Omurice
Ground-truth salinity	2.70 g	1.80 g	2.90 g	Ground-truth salinity	7.80 g	6.10 g	3.70 g
Estimation category	Spaghetti	Gratin	Stew	Estimation category	Fried rice	Chow mein	Beef and Potato stew
Estimation salinity	2.89 g	1.65 g	2.98 g	Estimation salinity	2.48 g	3.04 g	1.52 g
Error	+0.19 g	-0.15 g	-0.02 g	Error	-5.32 g	-3.06 g	-2.18 g

Figure 4.14: Successful and failed cases of salinity estimation results

## 4.5 Conclusions

In this study, a method to estimate food ingredients from a food image by multi-task CNN was proposed. As a benchmark dataset for evaluation of the proposed method, the new food image dataset was constructed by using publicly available images from several recipe-gathering websites. The two-stage transfer learning using a large number of category-annotated food image database was proposed to improve the estimation accuracy, because it is difficult to collect a large number of food images where the calorie content and salinity are known. I demonstrated the effectiveness of multi-task learning with food category classification to estimate the calorie and

Table 4.5: Comparison of calories estimation on each class

Category	Single-task		Multi-task		Multi-task w/ 2-F.T.	
	$E_{ab}$ (kcal)	$E_{re}$ (%)	$E_{ab}$ (kcal)	$E_{re}$ (%)	$E_{ab}$ (kcal)	$E_{re}$ (%)
Curry and rice	113.8	20.1	103.3	17.6	95.6	17.3
Fried rice	73.8	13.1	81.9	15.2	80.1	14.3
Chow mein	76.6	12.7	81.8	13.9	68.2	11.4
Spaghetti	98.7	16.2	97.2	16.0	86.3	14.0
Gratin	132.6	60.0	110.9	51.1	123.7	54.1
Miso soup	37.5	104.9	31.3	81.2	24.8	55.6
Stew	140.0	47.8	134.6	41.0	126.0	33.1
Beef and potato stew	94.8	32.0	79.5	28.6	104.8	33.6
Hamburg steak	101.8	31.6	117.3	34.6	81.6	24.5
Cold tofu	67.7	62.4	61.5	56.3	68.3	52.6
Scattered sushi	152.4	23.6	139.8	21.4	112.4	16.6
Omurice	172.4	25.5	119.3	20.3	179.1	27.5
Potato salad	128.0	84.7	112.6	73.0	92.1	60.4
Mixed rice	109.5	22.6	113.1	23.7	121.0	24.4

salinity content by showing the better performance compared with the single-task result. The existence of a relationship between the food category and salinity was experimentally confirmed.

Table 4.6: Comparison of salinity estimation on each class

Category	Single-task		Multi-task		Multi-task w/ 2-F.T.	
	$E_{ab}$ (g)	$E_{re}$ (%)	$E_{ab}$ (g)	$E_{re}$ (%)	$E_{ab}$ (g)	$E_{re}$ (%)
Curry and rice	0.64	28.5	0.51	24.5	0.40	18.5
Fried rice	0.91	51.9	1.10	64.9	1.04	56.8
Chow mein	0.86	22.8	0.75	21.1	0.94	28.2
Spaghetti	0.77	32.5	0.80	34.6	0.74	31.0
Gratin	0.70	43.7	0.61	32.2	0.60	33.2
Miso soup	0.54	35.7	0.50	34.8	0.57	38.1
Stew	0.71	31.5	0.78	38.7	0.81	36.8
Beef and potato stew	0.92	59.4	1.26	67.9	1.03	62.7
Hamburg steak	0.74	37.0	0.86	39.4	0.88	41.2
Cold tofu	0.66	57.5	0.66	49.5	0.52	40.6
Scattered sushi	1.42	33.5	1.15	25.0	1.30	28.8
Omurice	0.87	25.4	0.72	26.4	0.77	26.9
Potato salad	0.45	40.3	0.33	26.1	0.50	42.5
Mixed rice	0.90	32.7	1.03	39.6	0.80	30.1

# Chapter 5

## Conclusions

In this dissertation, I studied the implementation of deep learning for the development of welfare technology in two areas; hand posture classification and food constituent estimation. Both showed the importance of a large amount of data in presenting CNN performance with implemented deep learning method. I demonstrated the deep learning in both areas and presented the steps taken to solve this problem. I discuss deep learning in Chapter 2 of this dissertation, as the references method in particular CNN. CNN is the primary method used in the classification process and even in the feature extraction. Besides, I added an explanation regarding approaches to improve the performance of CNN, such as preprocessing (including data augmentation) and transfer learning.

In Chapter 3, an effective hand posture classification method by employing a progressive data augmentation method and utilizing the state-of-the-art CNN architecture was proposed. I suggested a data augmentation method based on 3D rotation on hand depth data to generate more new hand image data. This proposed idea successfully generated more new hand depth image data, which would otherwise require considerable effort to collect manually, and these data have enhanced CNN performance. Furthermore, Xception model as state-of-the-art CNN architecture for feature extraction and classification was applied. I demonstrated that our study produced

a better performance (3.9%) compared with previous research. The availability of a large hand image data and the effective hand posture classification technique can facilitate the development of a sign language recognition system.

In Chapter 4, a food constituent estimation method by multi-task CNN was proposed. In this research, two food image datasets, ingredients-annotated (including calories and salinity) dataset and category-annotated dataset, were constructed. An ingredient-annotated dataset is used for multi-task CNN, and the category-annotated dataset is applied for the proposed two-stage transfer learning. Transfer learning using a large number of category-annotated food image database is used to improve the estimation accuracy, because the collection of a large number of food images whose calorie content and salinity are known is difficult. As a result, our multi-task learning with food category classification to estimate the calorie and salinity content showed better performance compared with the single-task result. I experimentally confirmed the existence of a relationship between the food category and salinity. The availability method for salinity estimation can improve the food recognition system performance as an appliance for helping disease prevention.

Overall, the application of deep learning in this research makes the development of image-based welfare technology more straightforward, especially for the establishment of sign language recognition system and food constituent estimation system for lifestyle disease prevention. This research contributed by presenting the classification method using CNN and offered a way to improve its performance.

Given that the research I presented focused on the effectiveness of classification and estimation methods, assuredly this study still requires another supported way for developing a whole welfare technology system. Notably, in hand posture classification research, sign postures (alpha signs) that require finger motion were not included. Considering RNN for this problem, opens opportunities for deep learning to increase the recognition ability for sign posture involving finger/arm motion. Moreover, the use of color images together with depth images as the dataset is expected to expand

the range of the method performance. In food constituent estimation research, I only estimated the food image served for one person. Therefore, the estimation of the calorie and salinity from the food image serving for multiple people is necessary to be considered, along with quantity estimation. On the other hand, the robustness of the food images to the light conditions also need to be considered in the dataset construction.

# Bibliography

- [1] United Nations, Department of Economics and Social Affairs, Population Division: “World population ageing 2017 - Highlights (ST/ESA/SER.A/397),” New York: United Nations, 2017.
- [2] T. Mørk, G. Vidje, M. Gudnason, M. Harrikari, L. Hertzberg, H. Lagercrantz, N. Simic, H. Svensson, E. Winterberg, and L. W. Wehner, Eds.: “Focus on Welfare Technology,” Sweden: Nordic Centre for Welfare and Social Issues, 2011.
- [3] C. F. Pfeiffer, N. O. Skeie, S. Hauge, I. Lia, and I. Eilertsen: “Towards a Safer Home Living - Behavior Classification as a Method to Detect Unusual Behavior for People Living Alone,” Porsgrunn: Telemark University College, HiT Report, no. 15, 2015.
- [4] L. Liu, E. Stroulia, I. Nikolaidis, A. Miguel-Cruz, and A. Rios Rincon: “Smart Homes and Home Health Monitoring Technologies for Older Adults: A Systematic Review,” *Int. J. of Medical Informatics*, vol. 91, pp. 44–59, 2016.
- [5] A. Pande: “Science and Technology for Human Welfare and Its Effect on Environment,” *Advances in Life Science and Technology*, vol. 34, pp. 122–130, 2015.
- [6] M. Cozza, L. Crevani, A. Hallin, and J. Schaeffer: “Future Ageing: Welfare Technology Practices for Our Future Older Selves,” *Futures*, 2018.
- [7] B. Reeder, E. Meyer, A. Lazar, S. Chaudhuri, H. J. Thompson, and G. Demiris: “Framing the Evidence for Health Smart Homes and Home-based Consumer Health

- Technologies as a Public Health Intervention for Independent Aging: A Systematic Review,” *Int. J. of Medical Informatics*, vol. 82, no. 7, pp. 565–579, 2013.
- [8] D. J. Cook: “Health Monitoring and Assistance to Support Aging in Place,” *Journal of Universal Computer Science*, vol. 12, no. 1, 2006.
- [9] J. A. Kientz, S. N. Patel, B. Jones, E. Price, E. D. Mynatt, and G. D. Abowd: “The Georgia Tech Aware Home,” *Proc. of the twenty-sixth annual CHI conference extended abstracts on Human factors in computing systems - CHI’08*, pp. 3675–3680, 2008.
- [10] A. Gaddam: “Wireless Sensor Network based Smart Home for Elder Care,” New Zealand: Massey University, 2011.
- [11] Obi Robot, [Online] available from: “<https://meetobi.com/>”, accessed 17th January 2019.
- [12] Neater Eater Robotic, [Online] available from: “<http://www.neater.co.uk/neater-eater-2-2/>”, accessed 17th January 2019.
- [13] R. Soyama, S. Ishii, and A. Fukase: “The Development of Meal-Assistance Robot ‘My Spoon,’” *Proc. of the ICORR2003*, 2003.
- [14] F. Mälberg and L. Truong: “Feeding Robot,” Sweden: KTH Royal Institute of Technology, 2017.
- [15] M. Topping and J. Smith: “The Development of Handy 1, a Robotic System to Assist the Severely Disabled,” *ICORR ’99*, pp. 244–249, 1999.
- [16] iRobot Roomba, [Online] available from: “<https://www.irobot.com/for-the-home/vacuuming/roomba>”, accessed 17th January 2019.



- [17] S. Zhang, H. Hu, and H. Zhou: “An Interactive Internet-based System for Tracking Upper Limb Motion in Home-based Rehabilitation,” *Medical & Biological Engineering & Computation*, vol. 46, no. 3, pp. 241–249, 2008.
- [18] Y.-P. Wuang, C.-S. Chiang, C.-Y. Su, and C.-C. Wang: “Effectiveness of Virtual Reality Using Wii Gaming Technology in Children with Down Syndrome,” *Research in Development Disabilities*, vol. 32, no. 1, pp. 312–321, 2011.
- [19] P. H. Wilson, J. Duckworth, N. Mumford, R. Eldridge, M. Guglielmetti, P. Thomas, D. Shum and H. Rudolph: “A Virtual Tabletop Workspace for the Assessment of Upper Limb Function in Traumatic Brain Injury (TBI),” *Proc. of 2007 Virtual Rehabilitation*, pp. 14–19, 2007.
- [20] A. Y. Wang: “Games for Physical Therapy.” *CHI’12*, p. 5, 2012.
- [21] Mental Welfare Commission for Scotland: “Decisions about Technology,” Edinburgh: Thistle House, 2015.
- [22] The Danish Government: “Digital Welfare Empowerment, Flexibility and Efficiency, Common Public-Sector Strategy for Digital Welfare 2013-2020,” Denmark: The Danish Agency for Digitisation, 2013.
- [23] D. Ameta, K. Mudaliar, and P. Patel: “Medication Reminder and Healthcare – An Android Application,” *Int. J. of Managing Public Sector Information and Communication Technologies*, vol. 6, no. 2, pp. 39–48, Jun. 2015.
- [24] S. V. Zanjali and G. R. Talmale: “Medicine Reminder and Monitoring System for Secure Health Using IOT,” *Procedia Computer Science*, vol. 78, pp. 471–476, 2016.
- [25] M. D. Hatagundi, K. J. Uttarkar, T. Barker, and K. Hiremath: “Automatic Pill Dispenser,” *Int. J. of Advanced Research in Computer and Communication Engineering*, vol. 5, no. 7, pp. 543–547, 2016.

- [26] J. Wang: “Electrochemical Glucose Biosensors,” *Chem. Rev.*, no. 108, pp. 814–825, 2008.
- [27] V. Jones, V. Gay, and P. Leijdekkers: “Body Sensor Networks for Mobile Health Monitoring: Experience in Europe and Australia,” *Proc. of Sensors and Actuators*, pp. 204–209, 2010.
- [28] T. Chang: “Food Fight: A Social Diet Network Mobile Application,” Berkeley: University of California, Technical UCB/EECS-2012-133, 2012.
- [29] Y. Zhang, H. Liu, X. Su, P. Jiang, and D. Wei: “Remote Mobile Health Monitoring System Based on Smart Phone and Browser/Server Structure,” *J. of Healthcare Engineering*, vol. 6, no. 4, pp. 717–738, 2015.
- [30] H. Mora, D. Gil, R. Muñoz Terol, J. Azorín, and J. Szymanski: “An IoT-Based Computational Framework for Healthcare Monitoring in Mobile Environments,” *Sensors*, vol. 17, no. 10, 2017.
- [31] Oysta Safer Technology, [Online] available from: “<https://oystatechnology.co.uk/asset-monitoring/>”, accessed 17th January 2019.
- [32] T. Hori, Y. Nishida, H. Aizawa, S. Murakami, and H. Mizoguchi: “Sensor Network for Supporting Elderly Care Home,” *Proc. of IEEE*, vol. 2, pp. 575–578, 2004.
- [33] F. Fernández-Luque, J. Zapata, R. Ruiz, and E. Iborra, in J. Mira, J. Ferrández, J. Álvarez, F. de la Paz, and F. J. Toledo, Eds.: “A Wireless Sensor Network for Assisted Living at Home of Elderly People,” In *Bioinspired Applications in Artificial and Natural Computation*, Berlin: Springer Berlin Heidelberg, vol. 5602, pp. 65–74, 2009.
- [34] Fold Telecare, [Online] available from: “<https://www.foldtelecare.com/>”, accessed 17th January 2019.

- [35] M. Miura, S. Ito, R. Takatsuka, and S. Kunifuji, in I. Lovrek, R. J. Howlett, and L. C. Jain, Eds.: “Aware Group Home Enhanced by RFID Technology,” In Knowledge-Based Intelligent Information and Engineering Systems, Berlin: Springer Berlin Heidelberg, vol. 5178, pp. 847–854, 2008.
- [36] Division Luminaires ZVEI: “DALI manual,” Digital Addressable Lighting Interface Activity Group, 2001.
- [37] P. B. Rao and S. K. Uma: “Raspberry Pi Home Automation with Wireless Sensors Using Smart Phone,” *Int. J. of Computer Science and Mobile Computing*, vol. 4, no. 5, pp. 797–803, 2015.
- [38] A. B. Altayeva, B. S. Omarov, and Y. I. Cho: “Intelligent Microclimate Control System Based on IoT,” *Int. J. of Fuzzy Logic and Intelligent System*, vol. 16, no. 4, pp. 254–261, 2016.
- [39] Amazon Echo, [Online] available: “<https://www.amazon.com/Echo-2nd-Generation-International-Version/dp/B075RSCZHD>”, accessed 17th January 2019.
- [40] Z. Hu: “A Data Acquisition and Control System in Smart Home Based on the Internet of Things,” *Int. J. of Simulation – System, Science & Technology*, vol. 17, no. 7, pp. 17.1–17.5, 2016.
- [41] U. A. Syed and U. K. Muniandy: “The Smart Shower,” arXiv preprint arXiv:1407.1466, 2014.
- [42] J. A. Luis, J. A. G. Galán, and J. A. Espigado: “Low Power Wireless Smoke Alarm System in Home Fires,” *Sensors*, vol. 15, no. 8, pp. 20717–20729, 2015.
- [43] T. Linner, M. Kranz, L. Roalter, and T. Bock: “Robotic and Ubiquitous Technologies for Welfare Habitat,” *J. of Habitat Engineering*, vol. 3, no. 1, pp. 101–110, 2011.

- [44] Z. Luo, J. T. Hsieh, N. Balachandar, S. Yeung, G. Pusiol, J. Luxenberg, G. Li, L. J. Li, N. L. Downing, A. Milstein, and L. Fei-Fei: “Computer Vision-Based Descriptive Analytics of Seniors’ Daily Activities for Long-Term Health Monitoring,” *Proc. of Machine Learning Research*, pp. 1–18, 2018.
- [45] F. Zhu, M. Bosch, I. Woo, S. Y. Kim, C. J. Boushey, D. S. Ebert, and E. J. Delp: “The Use of Mobile Devices in Aiding Dietary Assessment and Evaluation,” *IEEE J. of Selected Topic in Signal Processing*, vol. 4, no. 4, pp. 756–766, 2010.
- [46] A. Guan, S. H. Bayless, and R. Neelakantan: “Connected Vehicle Insights: Trends in Computer Vision,” *ITS America Research*, 2012.
- [47] H. Erdogan, Y. Palaska, E. Masazade, D. E. Barkana, and H. K. Ekenel: “Vision-based Game Design and Assessment for Physical Exercise in a Robot-assisted Rehabilitation System,” *J. of the Institution of engineering Technology (IET) Computer Vision*, vol. 12, 2017.
- [48] H. H. Aghdam and E. J. Heravi: “Guide to Convolutional Neural Networks,” Switzerland: Springer International Publishing, 2017.
- [49] A. Kuznetsova, L. Leal-Taixe, and B. Rosenhahn: “Real-Time Sign Language Recognition Using a Consumer Depth Camera,” *Proc. of 2013 IEEE International Conference on Computer Vision Workshops*, pp. 83–90, 2013.
- [50] K. O. Rodriguez and G. C. Chavez: “Finger Spelling Recognition from RGB-D Information Using Kernel Descriptor,” *Proc. of XXVI Conference on Graphics, Patterns and Images*, pp. 1–7, 2013.
- [51] T. Joutou and K. Yanai: “A Food Image Recognition System with Multiple Kernel Learning,” *Proc. of 16th IEEE International Conference on Image Processing (ICIP)*, pp. 285–288, 2009.

- [52] M. H. Rahman, M. R. Pickering, D. Kerr, C. J. Boushey, and E. J. Delp: “A New Texture Feature for Improved Food Recognition Accuracy in a Mobile Phone Based Dietary Assessment System,” Proc. of IEEE International Conference on Multimedia and Expo Workshops, pp. 418–423, 2012.
- [53] Z.-Q. Zhao, P. Zheng, S. Xu, and X. Wu: “Object Detection with Deep Learning: A Review,” arXiv preprint arXiv:1807.05511, 2018.
- [54] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and F.-F. Li: “ImageNet: A Large-Scale Hierarchical Image Database,” Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 248–255, 2009.
- [55] A. Krizhevsky, I. Sutskever, and G. E. Hinton, in F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds.: “ImageNet: Classification with Deep Convolutional Neural Networks,” Proc. of Advances in Neural Information Processing Systems 25 (NIPS 2012), Curran Associates, Inc., pp. 1097–1105, 2012.
- [56] Y. LeCun, Y. Bengio, and G. Hinton: “Deep learning,” Nature, vol. 521, no. 7553, pp. 436–444, 2015.
- [57] F. Chollet: “Home - Keras Documentation,” [Online] available from: “<https://keras.io/>”, accessed 24th January 2019.
- [58] W. Zhao: “Research on the Deep Learning of the Small Sample Data based on Transfer Learning,” AIP Conference Proceedings, vol. 1864, no. 020018, 2017.
- [59] S. Dodge and L. Karam: “Understanding How Image Quality Affects Deep Neural Networks,” arXiv preprint arXiv: 1604.04004 [cs], 2016.
- [60] G. Zaccane, M. R. Karim, and A. Menshawy: “Deep Learning with Tensorflow,” Birmingham: Packt Publishing Ltd., 2017.
- [61] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola: “Dive into Deep Learning,” preprint, 2019.

- [62] R. Cadène, N. Thome, and M. Cord: “Master’s Thesis: Deep Learning for Visual Recognition,” arXiv preprint arXiv:1610.05567 [cs], vol. abs/1610.05567, 2016.
- [63] J. Schmidhuber: “Deep Learning in Neural Networks: An Overview,” *Neural Networks*, vol. 61, pp. 85–117, 2015.
- [64] A. Kurenkov: “A ‘Brief’ History of Neural Nets and Deep Learning,” Andrey Kurenkov’s Web World, 24th December 2015, [Online] available from: “<http://www.andreykurenkov.com/writing/ai/a-brief-history-of-neural-nets-and-deep-learning/>”, accessed 31st January 2019.
- [65] W. S. McCulloch and W. H. Pitts: “A Logical Calculus of The Ideas Immanent in Nervous Activity,” *J. of Bulletin of Mathematical Biophysics*, vol. 5, pp. 115–133, 1943.
- [66] D. O. Hebb: “The Organization of Behavior: A Neurophysiological Theory,” New York: John Wiley & Sons, Inc, 1949.
- [67] F. Rosenblatt: “The Perceptron, A Perceiving and Recognition Automaton (Project Para),” New York: Cornell Aeronautical Laboratory, no. 85-460-1, 1957.
- [68] K. Fukushima: “Neocognitron: A Self-organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position,” *J. of Biological Cybernetics*, vol. 36, no. 4, pp. 193–202, 1980.
- [69] D. E. Rumelhart, G. E. Hinton, and R. J. Williams: “Learning Internal Representations by Error Propagation,” CA: Institute for Cognitive Science, University of California, Technical, no. ICS 8506, 1985.
- [70] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel: “Backpropagation Applied to Handwritten Zip Code Recognition,” *J. of Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

- [71] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. J. Lang: "Phoneme Recognition Using Time-Delay Neural Networks," *IEEE Transactions on Acoustic, Speech and Signal Processing*, vol. 37, no. 3, pp. 328–339, 1989.
- [72] T. Kohonen: "Self-organized Formation of Topologically Correct Feature Maps," *J. of Biological Cybernetics*, vol. 43, no. 1, pp. 59–69, 1982.
- [73] G. A. Carpenter and S. Grossberg: "The ART of Adaptive Pattern Recognition by a Self-Organizing Neural Network," Boston: Center of Adaptive Systems, Department of Mathematics, Boston University, no. AFOSR-TK-88-0430, 1988.
- [74] G. E. Hinton and R. S. Zemel, in J. D. Cowan, G. Tesauro, and J. Alspector, Eds.: "Autoencoders, Minimum Description Length and Helmholtz Free Energy," *Proc. of Advances in Neural Information Processing Systems 6*, Morgan-Kaufmann, pp. 3–10, 1994.
- [75] K. S. Narendra and K. Parthasarathy: "Identification and Control of Dynamical Systems Using Neural Networks," *IEEE Transactions on Neural Networks*, vol. 1, no. 1, pp. 4–27, 1990.
- [76] D. A. Pomerleau, in D. S. Touretzky, Ed.: "ALVINN: An Autonomous Land Vehicle in a Neural Network," *Proc. of Advances in Neural Information Processing Systems 1*, Morgan-Kaufmann, pp. 305–313, 1989.
- [77] L.-J. Lin: "Reinforcement Learning for Robots Using Neural Networks," Pittsburgh: Carnegie Mellon University, 1993.
- [78] N. N. Schraudolph, P. Dayan, and T. J. Sejnowski, in J. D. Cowan, G. Tesauro, and J. Alspector, Eds.: "Temporal Difference Learning of Position Evaluation in the Game of Go," *Proc. of Advances in Neural Information Processing Systems 6*, Morgan-Kaufmann, pp. 817–824, 1994.

- [79] G. Tesauro: “Temporal Difference Learning and TD-Gammon,” *J. of Communication of the ACM*, vol. 38, no. 3, pp. 58–68, 1995.
- [80] Y. LeCun, L. Jackel, L. Bottou, A. Brunot, C. Cortes, J. Denker, H. Drucker, I. Guyon, U. Muller, E. Sackinger, P. Simard, and V. Vapnik, in F. Fogelman and P. Gallinari, Eds.: “Comparison of Learning Algorithm for Handwritten Digit Recognition,” *Proc. of International Conference on Artificial Neural Networks*, Cie Publishers, pp. 53–60, 1995.
- [81] CS231N: “Modeling one neuron,” CS231n Convolutional Neural Networks for Visual Recognition. [Online] available from: “<https://cs231n.github.io/neural-networks-1/#intro>, accessed 21st October 2019.
- [82] G. E. Hinton, S. Osindero, and Y.-W. Teh: “A Fast Learning Algorithm for Deep Belief Nets,” *J. of Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.
- [83] Y. Bengio and Y. LeCun, in L. Bottou, O. Chapelle, D. DeCoste, and J. Weston, Eds.: “Scaling Learning Algorithms towards AI,” in *Large-Scale Kernel Machines*, MIT Press, p. 41, 2007.
- [84] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, in B. Schölkopf, J. C. Platt, and T. Hoffman, Eds.: “Greedy Layer-Wise Training of Deep Networks,” *Proc. of Advances in Neural Information Processing Systems 19*, MIT Press, pp. 153–160, 2007.
- [85] K. Jarrett, K. Kavukcuoglu, M. A. Ranzato, and Y. LeCun: “What is the Best Multi-stage Architecture for Object Recognition?,” *Proc. of 2009 IEEE 12th International Conference on Computer Vision*, pp. 2146–2153, 2009.
- [86] V. Nair and G. E. Hinton: “Rectified Linear Units Improve Restricted Boltzmann Machines,” *Proc. of the 27th International Conference on machine Learning*, p. 8, 2010.



- [87] X. Glorot, A. Bordes, and Y. Bengio: “Deep Sparse Rectifier Neural Networks,” Proc. of the 14th International Conference on Artificial Intelligence and Statistics, vol. 15, pp. 315–323, 2011.
- [88] A. Mohamed, G. Dahl, and G. Hinton: “Deep Belief Networks for phone recognition,” Proc. of the NIPS Workshop on Deep Learning for Speech Recognition and Related Applications, p. 9, 2009.
- [89] A. L. Maas, A. Y. Hannun, and A. Y. Ng: “Rectifier Nonlinearities Improve Neural Network Acoustic Models,” Proc. of the 30th International Conference on Machine Learning, vol. 28, p. 6, 2013.
- [90] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov: “Improving Neural Networks by Preventing Co-adaptation of Feature Detectors,” arXiv preprint arXiv:1207.0580 [cs], 2012.
- [91] R. Raina, A. Madhavan, and A. Y. Ng: “Large-scale Deep Unsupervised Learning Using Graphics Processors,” Proc. of the 26th Annual International Conference on Machine Learning - ICML '09, pp. 1–8, 2009.
- [92] J. Bernauer: “Deep Learning and GPUs Intro and hands-on tutorial,” presented at the HPC Advisory Council Stanford, 2017.
- [93] DataRobot: “Unsupervised Machine Learning,” 23rd March 2018. [Online] available from: “<https://www.datarobot.com/wiki/unsupervised-machine-learning/>”, accessed 13rd February 2019.
- [94] Y. Bengio, A. Courville, and P. Vincent: “Representation Learning: A Review and New Perspectives,” arXiv preprint arXiv:1206.5538 [cs], 2012.
- [95] J. Hearty: “Advanced Machine Learning with Python,” Birmingham-Mumbai: Packt Publishing Ltd., 2016.

- [96] S. Rascka: “Python Machine Learning,” Birmingham-Mumbai: Packt Publishing Ltd., 2015.
- [97] O. Irsoy and C. Cardie: “Deep Recursive Neural Network for Compositionality in Language,” Proc. of Advances in Neural Information Processing Systems 27, pp. 2096–2104, 2014.
- [98] J. Le: “Convolutional Neural Networks: The Biologically-Inspired Model,” Towards Data Science, 29th August 2018. [Online] available from: “<https://towardsdatascience.com/convolutional-neural-networks-the-biologically-inspired-model-f2d23a301f71>”, accessed 18th February 2019.
- [99] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson: “Understanding Neural Networks Through Deep Visualization,” Proc of Deep Learning Workshop, 31st International Conference on Machine Learning, p. 12, 2015.
- [100] A. Kava: “That’s not enough, We have to go deeper — Deep Learning,” Medium, 09th August 2018. [Online] available from: “<https://medium.com/datadriveninvestor/thats-not-enough-we-have-to-go-deeper-24dd16d85828>”, accessed 10th December 2019.
- [101] CS231N: “Convolutional Neural Networks (CNNs / ConvNets),” CS231n Convolutional Neural Networks for Visual Recognition. [Online] available from: “<http://cs231n.github.io/convolutional-networks/>”, accessed: 22nd February 2019.
- [102] Ujjwalkarn: “An Intuitive Explanation of Convolutional Neural Networks,” The data science blog, 11st August 2016. [Online] available from: “<https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/comment-page-2/>”, accessed: 10th December 2019.
- [103] M. D. Zeiler and R. Fergus, in D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds.: “Visualizing and Understanding Convolutional Networks,” in Com-

- puter Vision – ECCV 2014, Cham: Springer International Publishing, vol. 8689, pp. 818–833, 2014.
- [104] K. Simonyan and A. Zisserman: “Very Deep Convolutional Networks for Large-Scale Image Recognition,” arXiv preprint arXiv:1409.1556, 2014.
- [105] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich: “Going Deeper with Convolutions,” Proc. of 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1–9, 2015.
- [106] K. He, X. Zhang, S. Ren, and J. Sun: “Deep Residual Learning for Image Recognition,” Proc. of 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778, 2016.
- [107] F. Chollet : “Xception: Deep Learning with Depthwise Separable Convolutions,” arXiv preprint arXiv:1610.02357, 2016.
- [108] A. Geitgey: “Machine Learning is Fun! Part 3: Deep Learning and Convolutional Neural Networks,” Medium, 14th June 2014. [Online] available from: “<https://medium.com/@ageitgey/machine-learning-is-fun-part-3-deep-learning-and-convolutional-neural-networks-f40359318721>”, accessed 10th December 2019.
- [109] N. Bobb: “Image Data Pre-Processing for Neural Networks,” Becoming Human: Artificial Intelligence Magazine, 11st September 2017. [Online] available from: “<https://becominghuman.ai/image-data-pre-processing-for-neural-networks-498289068258>”, accessed 19th February 2019.
- [110] CS231N: “Transfer Learning,” CS231n Convolutional Neural Networks for Visual Recognition. [Online] available from: “<http://cs231n.github.io/transfer-learning/#tf>”, accessed 27th February 2019.

- [111] J. Brownlee: “A Gentle Introduction to Transfer Learning for Deep Learning,” Machine Learning Mastery, 20th December 2017. [Online] available from: “<https://machinelearningmastery.com/transfer-learning-for-deep-learning/>”, accessed 27th February 2019.
- [112] P. Marcelino: “Transfer Learning from Pre-trained Models,” Towards Data Science, 23rd October 2018. [Online] available from: “<https://towardsdatascience.com/transfer-learning-from-pre-trained-models-f2393f124751>”, accessed 27th February 2019.
- [113] H. Takimoto, S. Yoshimori, Y. Mitsukura, and M. Fukumi: “Hand Posture Recognition Robust for Posture Changing in Complex Background,” *J. of Signal Process.*, vol. 14, no. 6, pp. 483–490, 2010.
- [114] H.-D. Yang : “Sign Language Recognition with the Kinect Sensor Based on Conditional Random Fields,” *Sensors*, vol. 15, no. 1, pp. 135–147, 2015.
- [115] N. Pugeault and R. Bowden: “Spelling it out: Real-time ASL fingerspelling recognition,” *Proc. of 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pp. 1114–1119, 2011.
- [116] A. Kurakin, Z. Zhang, and Z. Liu : “A Real Time System for Dynamic Hand Gesture Recognition with a Depth Sensor,” *Proc. of the 20th European Signal Processing Conference (EUSIPCO)*, pp. 1975–1979, 2012.
- [117] A. B. Jmaa, W. Mahdi, Y. B. Jemaa, and A. B. Hamadou: “A New Approach For Hand Gestures Recognition Based on Depth Map Captured by RGB-D Camera,” *J. of Computación y Sistemas*, vol. 20, no. 4, pp. 709–721, 2016.
- [118] T. Kim, K. Livescu, and G. Shakhnarovich : “American Sign Language Fingerspelling Recognition with Phonological Feature-based Tandem Models,” *Proc. of 2012 IEEE Spoken Language Technology Workshop (SLT)*, pp. 119–124, 2012.

- [119] A. K. Gautam and A. Kaushik: “American Sign Language Recognition System Using Image Processing Method,” *Int. J. of Computer Science and Engineering (IJCSE)*, vol. 9, no. 7, pp. 466–471, 2017.
- [120] Y. Ji, S. Kim, and K.-B. Lee: “Sign Language Learning System with Image Sampling and Convolutional Neural Network,” *Proc. of 2017 First IEEE International Conference on Robotic Computing (IRC)*, vol. 1, pp. 371–375, 2017.
- [121] L. Pigou, S. Dieleman, P.-J. Kindermans, and B. Schrauwen, in L. Agapito, M. M. Bronstein, and C. Rother, Eds.: “Sign Language Recognition Using Convolutional Neural Networks,” in *Computer Vision - ECCV 2014 Workshops*, Cham: Springer International Publishing, vol. 8925, pp. 572–578, 2015.
- [122] N. M. R. Aquino, M. Gutoski, L. T. Hattori, and H. S. Lopes: “The Effect of Data Augmentation on the Performance of Convolutional Neural Networks,” *Proc. of XIII Brazilian Congress on Computational Intelligence (CBIC 2017)*, p. 12, 2017.
- [123] J. Wang and L. Perez: “The Effectiveness of Data Augmentation in Image Classification Using Deep Learning,” *arXiv preprint arXiv:1712.04621*, p. 8, 2017.
- [124] B. Kang, S. Tripathi, and T. Q. Nguyen: “Real-time Sign Language Fingerspelling Recognition Using Convolutional Neural Networks from Depth Map,” *arXiv preprint arXiv:1509.03001*, 2015.
- [125] J. Brownlee: “How to Configure Image Data Augmentation When Training Deep Learning Neural Networks,” *Machine Learning Mastery*, 12nd April 2019. [Online] available from: “<https://machinelearningmastery.com/how-to-configure-image-data-augmentation-when-training-deep-learning-neural-networks/>”, accessed 24th May 2019.

- [126] AIST, 2012. [Online] available from: “<https://unit.aist.go.jp/hiri/dhrg/ja/dhdb/hand/data/list.html>”, accessed 18th March 2019.
- [127] S. Christodoulidis, M. Anthimopoulos, L. Ebner, A. Christe, and S. Mougiakakou: “Multi-source Transfer Learning with Convolutional Neural Networks for Lung Pattern Analysis,” *J. of IEEE Biomedical and Health Informatics*, vol. 21, no. 1, pp. 76–84, 2017.
- [128] A. Jenitha, L. P. Dhana, B. Jagannathan, M. Niranjanaiah, and R. Priyanka: “A Personal Assistive System for Monitoring Calorie and Nutrition from Food Image,” *Int. J. of Electrical, Electronics and Computer Systems (IJEECS)*, vol. 6, no. 3, p. 6, 2018.
- [129] S. Hoi: “FoodAI: Food Image Recognition by Deep Learning,” School of Information Systems Singapore Management University. [Online] available from: “<http://images.nvidia.com/content/APAC/events/ai-conference/resource/ai-for-research/FoodAI-Food-Image-Recognition-with-Deep-Learning.pdf>”, accessed 26th March 2019.
- [130] FAO: “Food and Agriculture Organization of the United Nations, Dietary Assessment A Resource Guide to Method Selection and Application in Low Resource Settings,” Rome: FAO, 2018.
- [131] MyFitnessPal, [Online] available from: “<https://www.myfitnesspal.com/>”, accessed 25th March 2019.
- [132] mynetdiary: “Online Food Diary and Calorie Counter, with free iPhone, Android, and iPad mobile apps,” [Online] available from: “<https://www.mynetdiary.com/>”, accessed 25th March 2019.

- [133] Foodlog, [Online] available from: "<http://www.foodlog.jp/en>", accessed 25th March 2019.
- [134] Caloriemama: "Calorie Mama Food AI - Food Image Recognition and Calorie Counter Using Deep Learning," [Online] available from: "<http://www.caloriemama.ai/>", accessed 25th March 2019.
- [135] A. H. Abdulnabi, G. Wang, J. Lu, and K. Jia: "Multi-task CNN Model for Attribute Prediction," *J. of IEEE Transactions on Multimedia*, vol. 17, no. 11, pp. 1949–1959, 2015.
- [136] myfood24: "myfood24 - Dietary Assessment for Researchers, Teaching and Health Professionals," [Online] available from: "<https://www.myfood24.org/>", accessed 2nd April 2019.
- [137] Nutritools: "Dietary Assessment Tools - Nutritools," [Online] available from: "<https://www.nutritools.org/tools>", accessed 2nd April 2019.
- [138] Perfect Diet Tracker: "Diet Information, Perfect Diet Tracker," [Online] available from: "<https://www.perfectdiettracker.com/information.html>", accessed 2nd April 2019.
- [139] C. C. Tsai, G. Lee, F. Raab, G. J. Norman, T. Sohn, W. G. Griswold, and K. Patrick: "Usability and Feasibility of PmEB: A Mobile Phone Application for Monitoring Real Time Caloric Balance," *J. of Mobile Networks and Applications*, vol. 12, pp. 173–184, 2007.
- [140] M. Carter, V. Burley, C. Nykjaer, and J. Cade, in G. Eysenbach, Ed.: "Adherence to a Smartphone Application for Weight Loss Compared to Website and Paper Diary: Pilot Randomized Controlled Trial," *J. of Medical Internet Research*, vol. 13, no. 4, pp. 45–61, 2013.

- [141] R. Z. Franco, R. Fallaiza, J. A. Lovegrove, and F. Hwang: “Popular Nutrition-Related Mobile Apps: A Feature Assessment,” *JMIR Mhealth Uhealth*, vol. 4, no. 3, p. 12, 2016.
- [142] S. Yang, M. Chen, D. Pomerleau, and R. Sukthankar: “Food Recognition Using Statistics of Pairwise Local Features,” *Proc. of 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2249–2256, 2010.
- [143] J. Shotton, M. Johnson, and R. Cipolla: “Semantic Texton Forests for Image Categorization and Segmentation,” *Proc. of the Conference on Computer Vision and Pattern Recognition (CVPR) 2008*, p. 8, 2008.
- [144] F. Kong and J. Tan: “Dietcam: Automatic Dietary Assessment with Mobile Camera Phones,” *Proc. of Pervasive and Mobile Computing*, pp. 147–163, 2012.
- [145] Y. He, C. Xu, N. Khanna, C. J. Boushey, and E. J. Delp: “Food Image Analysis: Segmentation, Identification and Weight Estimation,” *Proc. of IEEE International Conference Multimedia Expo (ICME)*, p. 15, 2013.
- [146] K. Aizawa, Y. Maruyama, H. Li, and C. Morikawa: “Food Balance Estimation by Using Personal Dietary Tendencies in a Multimedia Food Log,” *IEEE Transactions on Multimedia*, vol. 15, no. 8, pp. 2176–2185, 2013.
- [147] 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,” *IEEE J. of Biomedical and Health Informatics*, vol. 18, no. 4, pp. 1261–1271, 2014.
- [148] Y. Kawano and K. Yanai: “Food Image Recognition with Deep Convolutional Features,” *Proc. of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing Adjunct Publication - UbiComp’14 Adjunct*, pp. 589–593, 2014.



- [149] H. Kagaya, K. Aizawa, and M. Ogawa: “Food Detection and Recognition Using Convolutional Neural Network,” Proc. of the ACM Multimedia Conf., Florida, 2014.
- [150] K. Yanai and Y. Kawano: “Food Image Recognition Using Deep Convolutional Network with Pre-training and Fine-tuning,” Proc. of the 2015 IEEE International Conference on Multimedia & Expo Workshops (ICMEW), pp. 1–6, 2015.
- [151] A. Singla, L. Yuan, and T. Ebrahimi: “Food/Non-food Image Classification and Food Categorization Using Pre-Trained GoogLeNet Model,” Proc. of the 2nd International Workshop on Multimedia Assisted Dietary Management-MADiMa’16, pp. 3–11, 2016.
- [152] C. Liu, Y. Cao, Y. Luo, G. Chen, V. Vokkarane, and Y. Ma, in C. K. Chang, L. Chiari, Y. Cao, H. Jin, M. Mokhtari, and H. Aloulou, Eds.: “DeepFood: Deep Learning-Based Food Image Recognition for Computer-Aided Dietary Assessment,” in *Inclusive Smart Cities and Digital Health*, Cham: Springer International Publishing, vol. 9677, pp. 37–48, 2016.
- [153] H. Hassannejad, G. Matrella, P. Ciampolini, I. D. Munari, M. Mordonini, and S. Cagnoni: “Food Image Recognition Using Very Deep Convolutional Networks,” Proc. of the Int. Workshop on Multi. Assisted Dietary Manag., pp. 41–49, 2016.
- [154] A. Myers, N. Johnston, V. Rathod, A. Korattikara, A. Gorban, N. Silberman, S. Guadarrama, G. Papandreou, J. Huang, and K. Murphy: “Im2Calories: Towards An Automated Mobile Vision Food Diary,” Proc. of 2015 IEEE International Conference on Computer Vision (ICCV), pp. 1233–1241, 2015.
- [155] L. Bossard, M. Guillaumin, and L. V. Gool: “Food-101 -- Mining Discriminative Components with Random Forests,” 2014. [Online] available from: “[http://www.vision.ee.ethz.ch/datasets\\_extra/food-101/](http://www.vision.ee.ethz.ch/datasets_extra/food-101/)”, accessed 3rd April 2019.

- [156] P. Pouladzadeh, P. Kuhad, S. V. B. Peddi, A. Yassine, and S. Shirmohammadi: “Food Calorie Measurement Using Deep Learning Neural Network,” Proc. of the IEEE International Instrumentation and Measurement Technology Conference, p. 6, 2016.
- [157] R. Girshick: “Fast R-CNN,” Proc. of the IEEE International Conference on Computer Vision (ICCV), 2015.
- [158] J. Chen and C. Ngo: “Deep-based Ingredient Recognition for Cooking Recipe Retrieval,” Proc. of the 2016 ACM on Multimedia Conference - MM’16, pp. 32–41, 2016.
- [159] T. Ege and K. Yanai: “Simultaneous Estimation of Food Categories and Calories with Multi-task CNN,” Proc. of the 2017 Fifteenth IAPR International Conference on Machine Vision Applications (MVA), pp. 198–201, 2017.
- [160] AJINOMOTO PARK, [Online] available from: “<https://park.ajinomoto.co.jp/recipe/>”, accessed 9th April 2019.
- [161] Kikkoman homepage, [Online] available from: “<https://www.kikkoman.co.jp/homecook/index.html?version=&>”, accessed 9th April 2019.
- [162] Orangepagenet, [Online] available from: “<https://www.orangepage.net/>”, accessed 9th April 2019.
- [163] Lettuce Club News, [Online] available from: “<https://www.lettuceclub.net/recipe/?sns=fb>”, accessed 9th April 2019.
- [164] Mizkan, [Online] available from: “<http://www3.mizkan.co.jp/sapari/menu/cook/search/index.html>”, accessed 10th April 2019.
- [165] Kewpie, [Online] available from: “<https://www.kewpie.co.jp/recipes/>”, accessed 9th April 2019.

- [166] Y. Matsuda, H. Hoashi, and K. Yanai: “Recognition of Multiple-Food Images by Detecting Candidate Regions,” Proc. of the 2012 IEEE International Conference on Multimedia and Expo, pp. 25–30, 2012.
- [167] Rakuten Recipe, [Online] available from: “<http://www.nii.ac.jp/dsc/idr/rakuten/rakuten.html>”, accessed 10th April 2019.
- [168] M. Long, Z. Cao, J. Wang, and P. S. Yu: “Learning Multiple Tasks with Multi-linear Relationship Networks,” arXiv preprint arXiv:1506.02117, 2015.
- [169] Y. Lu, A. Kumar, S. Zhai, Y. Cheng, T. Javidi, and R. Feris: “Fully-Adaptive Feature Sharing in Multi-Task Networks with Applications in Person Attribute Classification,” in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, pp. 1131–1140, 2017.
- [170] Y. Zhang and Q. Yang: “A Survey on Multi-Task Learning,” arXiv preprint arXiv:1707.08114, 2017.
- [171] R. Caruana: “Multitask Learning,” in Machine Learning, Boston: Kluwer Academic Publishers, vol. 28, pp. 41–75, 1997.
- [172] S. Ruder: “An Overview of Multi-Task Learning in Deep Neural Networks,” arXiv preprint arXiv:1706.05098, 2017.
- [173] M. Oquab, L. Bottou, I. Laptev, and J. Sivic: “Learning and Transferring Mid-level Image Representations Using Convolutional Neural Networks,” Proc. of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1717–1724, 2014.