Thesis for the Degree of Master of Engineering

Fish Classification based on Convolutional Neural Networks

by

Nepomscene Nduwarugira

Department of IT Convergence and Application Engineering

The Graduate School

Pukyong National University

August 2019

Fish classification based on Convolutional Neural Networks

콘볼루션 신경망에 기초한 물고기 분류

Advisor: Prof. Bong-Kee Sin

by

Nepomscene Nduwarugira

A thesis submitted in partial fulfillment of requirements for

the degree of Master of Engineering

in Department of IT Convergence and Application Engineering

The Graduate School, Pukyong National University

August 2019

Fish Classification based on Convolutional Neural Networks

A thesis

by

Nepomscene Nduwarugira

Approved by:

_____

(Chairman) Prof. Jong Nam Kim

_____

(Member) Prof. Seon-Han Choi

_____

(Member) Prof. Bong-Kee Sin

August 2019

Contents

Contents i

List of figures ii

List of tables iii

Nepomscene Nduwarugira v

요 약 v

I. Introduction 1

II. Related work 2

III. Artificial intelligence 3

3.1. Machine learning 3
3.2. Artificial neural network 5
3.2.1. Training the model 8
3.2.2. Overfitting 9
3.2.3. Dropout 10
3.2.4. Batch Normalization 11
3.2.5. Activation Functions 12
3.2.6. Optimizer 13
3.2.7.Softmax 13
IV. Deep learning 14

Proposed methods 15

4.1. Convolutional neural network 15
4.1.1. Input layer 16
4.1.2. Convolutional Layer 17
4.1.3. Pooling layer 18
4.1.4. Flattening Layer 18
4.1.5. Fully connected layer 19

4.1.6. Output layer 20
4.2. Recurrent neural networks (RNN) 20
4.3. Long Short-Term Memory (LSTM) 21
4.4. Dropout 21
4.5. Top k candidates 22
V. Experiments and results 23
5.1. QUT Fish Dataset 23
5.2. Results 23
VI. Conclusion 26
Reference 27
Acknowledgment 30

List of figures

Figure 1-Relationship between different subset of AI 3
Figure 2-Artificial Neural Network 6
Figure 3-A simple feed-forward neural network 6
Figure 4-Illustration of how the learning rate affects the convergence. 9
Figure 5-Division of dataset for training 10
Figure 6-Half of the neurons in the hidden layer are disabled 10
Figure 7-Activation function ReLU as a function of x. 12
Figure 8-Detailed workflow for training and testing a deep learning model. 15
Figure 9-CNN architecture 16
Figure 10-Convolutional layer 17
Figure 11-pooling layer 18
Figure 12-Flattened layer 19
Figure 13-Fully connected layer 20
Figure 14-recurrent neural network, where A is a cell of the network. 21
Figure 15-Before and after applying dropout method 22
Figure 16-Some samples from QUT 23
Figure 17-Illustration on how test loss decreases 25

List of tables

Table 1….....2

Table 2….....24

Table 3….....25

List of abbreviations

AI: Artificial Intelligence

ML: Machine Learning

ANN: Artificial Neural Network

DL: Deep Learning

DNN: Deep Neural Network

CNN: Convolutional Neural Network

RNN: Recurrent Neural Network

LSTM: Long Short-Term Memory

lr : Learning rate

NN: Neural network

SGD: Stochastic Gradient Descent

DN: Deep Network

CPU: Central Processing Unit

GPU: Graphics processing unit

ReLu: Rectified Linear Unit

콘볼루션 신경망에 기초한 물고기 분류

Nepomscene Nduwarugira

부 경 대 학 교 대 학 원 IT 융 합 응 용 공 학 과

요 약

자동 어류 이미지 분류 시스템은 어류를 소비하는 과정이나 다른 일을 위해 분류하는 과정에서 매우 중요한 역할을 한다. 어류 이미지를 수동으로 다른 등급으로 분류하는 것은 어렵고 지치고 지루하다. 이 논문은 어류 이미지를 분류할 수 있는 빠르고 정확한 시스템을 제안한다. 시스템은 이미지 전처리, 특징 추출 및 분류 방법으로 구성된다. 제안된 어류 분류 모델을 훈련하고 테스트하기 위해, 심층 신경망의 두 가지 대표적인 유형인 Convolution Neural Network (CNN)과 Long Short Term Memory(LSTM)을 사용했다. 이 중 CNN은 텍스트 분류에서 인기가 높은 LSTM보다 이미지 분류에 대해 더 높은 성능을 보여주었다. 게다가, 훈련의 어려움 때문에, 정규화 기법인 Dropout과 N-best 후보법을 적용하여, 오버피팅을 피하고, 단 하나의 최적해가 아닌 복수의 답안을 반환했다.

Abstract

Automatic fish images classification system plays a very important role in the process of dividing fishes into categories for human consumption or other tasks. To manually classify fishes into different classes is difficult, tiring and boring. This thesis proposes a fast and accurate system capable of classifying fish images into different categories. The system comprises image processing, feature extraction and classification method. To train and test the proposed fish classification model, we used the convolutional neural network (CNN) and the long short-term memory (LSTM) which are two representative types of deep neural network. Among them, CNN has been, proved more for images classification, than LSTM, which is popular for text classification. Furthermore, due to difficulties in training, regularization technique of dropout and N-best candidates methods were applied to avoid overfitting and return multiple answers instead of a single best.

I. Introduction

Nowadays, people are thirsty to easily classify fishes into their species. Manually separating fishes into different types is difficult, tiring and boring.

In this research, automatic fish classification model was proposed using Deep Neural Network (DNN), specially, Convolutional Neural Network (CNN) and long short-term memory (LSTM) but it is found that CNN is very preferred method recording 81% against 47% from LSTM. The system will consider the similarity of fishes based on their features. Due to overfitting appeared during training model, dropout method has been applied to overcome this issue and top k candidates method were applied in testing to return multiple answers instead of a single best

This dissertation is organized as follow:

Chapter 2 introduces relationship between my used methods and others, chapter 3 describes artificial intelligence (AI) and theirs subsets, chapter 4 develops the procedure of different models applied such as: convolutional neural network(CNN), dropout model and top k candidates ,chapter 5 explains the experiments and results after applying several models to classify fishes into their species. Finally, chapter 6 concludes the thesis.

II. Related work

Some researches on fish images classification and recognition have been published by applying convolutional neural network. In [1], they proposed a CNN architecture to recognize fish images and the dataset size is 22 476 fish images. The dataset is divided just into four classes

The accuracy reached 96%.

In [2], we used the same dataset to classify fish images and the test accuracy is different because the author did not apply dropout method to remove some useless nodes and edges, then the top k candidates method was not applied to return multiple answers instead of a single best. My work focused on three important methods in fish images classification according to the difficulties behind in image classification.

Table 1: My work compared with the previous work

#	Conv layer	Pooling layer	Dropout	Top k candidates	Performance
My work	5x5 4x4 3x3	2x2 2x2 2x2	0.5	1	59%
The previous work	5x5 5x5 5x5	3x2 3x2 3x3	0	1	52.59%

III. Artificial intelligence

Humans can automatically perform the intellectual tasks. After these tasks, it can be described as artificial intelligence [3]. Machine learning is a subset of artificial intelligence where the relationship between the input and output data of a system is modelled by extracting attributes features from the input data. Artificial intelligence has a several subset, one of them is machine learning. Artificial neural networks are a subset of machine learning, which are inspired by the structure within the human brain, using layers of neurons. Deep learning is also a subset of neural networks where it consists of many layers. The figure 1 illustrates the relationship between artificial intelligence, machine learning, neural networks and deep learning. Deep learning has a wide range of successful applications such as computer vision, speech recognition and image classification [4].

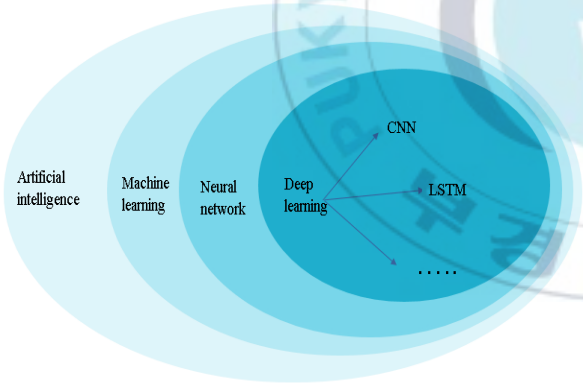


Figure 1-Relationship between different subset of AI

3.1. Machine learning

Nowadays, machine learning is needed almost everywhere and has become an important tool in our daily lives. Sometimes, people cannot realize whether the machine learning has been using. But it is used in many applications such as:

- Email accounts;
- Banks;
- Social media ;
- And so on.

Many problems can be solved by applying machine learning.

Some are solved by applying supervised learning and unsupervised learning.

- Supervised learning aims to teach machine features of the data labeled to make the wanted results with a test data. Supervised learning can be called classification model
- Unsupervised learning is applied when the input data is not labeled or unknown. The algorithm has a goal to group together the data, which are the same features. Unsupervised learning can be called clustering model

  Unsupervised learning is used, for example, to discover which products have been frequently purchased together from a store's transaction data [5].

The focus of this thesis is a branch of machine learning called deep learning, specifically convolutional neural network, which borrow concepts from the functionality of our own brains. Conventional computer programs are very good at quickly performing arithmetic computations and accurately following a list of instructions, both of which the human brain can stumble over. Conversely, some problems that humans can solve in microseconds are utterly impossible for traditional programs. For example, a young child can instantly recognize whether a photograph is a picture of a dog or a cat, but the task is very hard to define in a set of instructions for a computer to follow. We could start by finding edges from the image and devise various rules about what kind of a shape constitutes a dog versus a cat. That approach could conceivably come up with a satisfactory model for recognizing cats and dogs through careful observation and countless rounds

of trial and error. The problem arises when a new requirement turns up that we must also recognize rabbits and crocodiles. We must reconstruct the whole model to accommodate. Tackling these kinds of problems requires an entirely different way of programming a computer to mimic the human brain [6].

3.2. Artificial neural network

Artificial neural networks first showed interest after the publication of a paper by McCulloch and Pitts in 1943. They introduced a model of simplified artificial neurons as conceptual components for circuits capable of computational tasks [7]. Since then, the study of artificial neural networks has advanced in strides and modern neural networks are often capable of matching or even outperforming humans in tasks that were once considered near impossible for computers. The basic building block of artificial neural networks, the neuron, is based on the biological neurons found in the human brain. A simplified explanation of the current understanding of how we learn is that the neuron receives inputs through structures called dendrites, and the connections dynamically strengthen or weaken based on how often they are used. Each input connection's strength determines its weight when the inputs are summed and transformed in the cell body into a new signal, which is the output of the neuron. This output is then transmitted to other neurons and the process continues. This practical understanding of biological neurons can be translated into a computational model. [8]

The artificial neuron also gets some number of inputs, $x_1, x_2, \ldots, x_n$, which are multiplied by some weights, $w_1, w_2, \ldots, w_n$. The sum of those weighted inputs is called the logit of the neuron,

$$z=\sum_{i=0}^{n} wixi \quad (1)$$

This logit is used as the input for a function f to produce the output $y = f(z)$, as depicted in Figure 1. The inputs and weights are usually expressed in vector form $x = [x1\ x2\ \ldots\ xn]$ and $w = [w_1\ w_2\ \ldots\ wn]$ so the output can be computed as the dot product of the input and weight vectors $y = f(x.w + b)$, where b is a constant bias

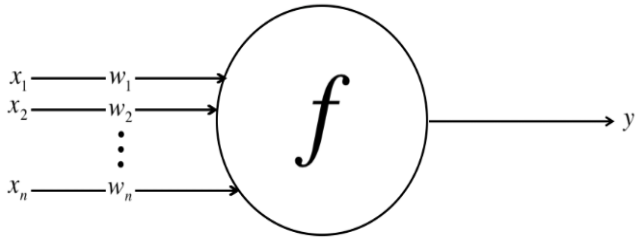


Figure 2-Artificial Neural Network

Creating an artificial neural network is connecting these neurons together to form an approximate model of how biological neurons connect to each other in the brain.

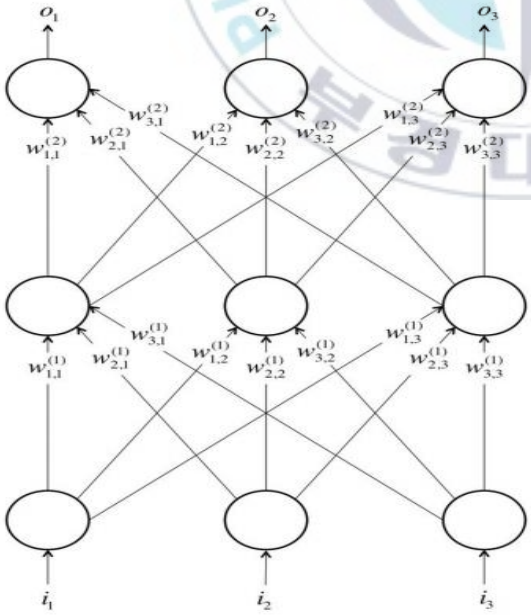


Figure 3-A simple feed-forward neural network

Figure 3 depicts an example of a very simple network. The bottom layer of the network brings in the input data and the top layer computes the result. This kind of network is called a feed-forward network, as there are no connections between neurons in the same layer or from a higher layer to a lower one. The layer in the middle, called the hidden layer, is what enables the network to learn and where most of the interesting work happens. Finding the optimal weights for the hidden layer is the answer to solving problems with neural networks. In more complicated neural networks, there are usually multiple hidden layers with different numbers of neurons and each with their own weight vectors, connected to each other before feeding into the output layer. Often the hidden layers have far fewer neurons than the input layer, which facilitates the learning of compressed representations of the input. As a biological analogy, our eyes receive a huge amount of raw input data through the photoreceptor cells, but our brain perceives it in terms of edges and contours. The hidden layers of the brain network come up with simplified representations of our surroundings.

As the neuron is mathematically expressed as a vector, we can express a neural network as a sequence of matrix operations. With an input layer

$x = [x_1\, x_2 \ldots x_n]$, we are trying to find the output vector

$y = [y_1\, y_2 \ldots y_m]$. This can be expressed as a matrix multiplication by constructing a weight matrix $W$of size $n \times m$ and a bias vector of length n. Each column of the matrix represents a neuron and the jth element of the column represents the weight of the connection. The transformation function

$y = f\,(Wx^T + b)$, is applied element-wise. (2)

3.2.1. Training the model

To achieve a more accurate parametric model the parameters must be optimized through a training process. To evaluate the performance of the model the deviation of the prediction must be quantified. A loss function can be used to compute a numerical value, where a larger loss value corresponds to a larger deviation [3]. Cross-entropy is a loss function, which is suitable for classification problems [9]. The reason behind this is that it takes into consideration "how wrong" the prediction was. A high probability for an incorrect prediction will give a high loss whilst a high probability for a correct prediction will give a loss close to zero. The cross-entropy is given by

$$\mathbf{L}\,(\boldsymbol{x_i}, \boldsymbol{y_{i,}}\boldsymbol{\theta}) = -\sum_{k=1}^{k} \boldsymbol{yik}\ \ \mathbf{log\ q_k\ (xi;\ \theta)\ ,\ (3)}$$

x_i: input data
y_i: Prediction
θ : Parameters
k : Number of classes
q_k: Estimated probability

Where xi is the i: th input data point, yi the corresponding prediction, θ the parameters in the model, K the number of classes and qk the estimated probability for the i:th data point belonging to the k:th class [10]. From the loss function a cost function can be defined as the average of the losses for all data points as stated below

$$\boldsymbol{J(\theta)}\frac{\mathbf{1}}{\boldsymbol{n}}\sum_{\boldsymbol{i=1}}^{\boldsymbol{n}}\boldsymbol{L(xi, yi, \theta)}\ \ \mathbf{(4)},$$

where $\boldsymbol{J(\theta)}$ is the cost function dependent on the parameters of the model, *n* the number of data points and $\boldsymbol{L(xi, yi, \theta)}$ the loss function. The parameters of the model are optimized by minimizing a loss function using an optimization algorithm. The aim of the algorithm is to find the minimum, often through an iterative process. The global minimum of a multi-dimensional

function is difficult to find and therefore, a local minimum is considered sufficient. In each step of the training, a loss is calculated, and the parameters are then adjusted to minimize the loss [10]. Stochastic Gradient Descent (SGD) is an optimization algorithm, which computes an approximation of the gradient of the cost function. A local minimum of the cost function can be found by taking a step in the direction of the negative gradient. SGD divides the data into so-called mini-batches and computes the gradient for each mini-batch, in order to reduce the computational time [10]. The size of the step is given by the so-called learning rate (lr). A higher learning rate corresponds to a larger step. If the learning rate is too low the model might not reach the minimum while a too high learning rate might not converge, as illustrated in figure 2 [10].

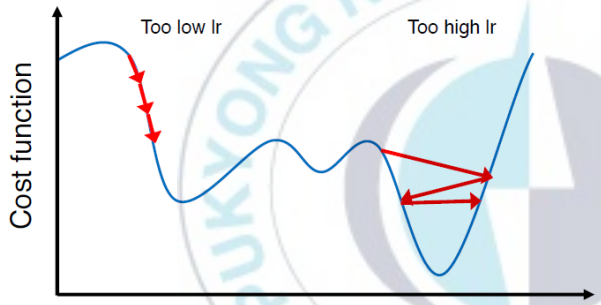


Figure 4-Illustration of how the learning rate affects the convergence.

The Adam optimization algorithm is a version of the SGD where the learning rate is adjusted for each step. The algorithm is computationally efficient and requires little memory [5].

3.2.2. Overfitting

As is true for many machine learning approaches, building a very complex model can easily lead to perfectly fitting the training data. When such a model is evaluated on new data, it performs poorly. Overfitting is a huge challenge in any machine learning task, and especially so in deep learning as neural networks usually have many layers with a large number of neurons, producing a very complex model. One approach to prevent overfitting is dividing the training process into

epochs, single iterations over the full training set. After each epoch, the model is evaluated on a set of validation data to see how well it is generalizing. If the accuracy keeps increasing on the training data, but decreases on the validation data, it is a sign of overfitting and the training should be stopped. Figure 5 shows an example division of the full dataset. [6]

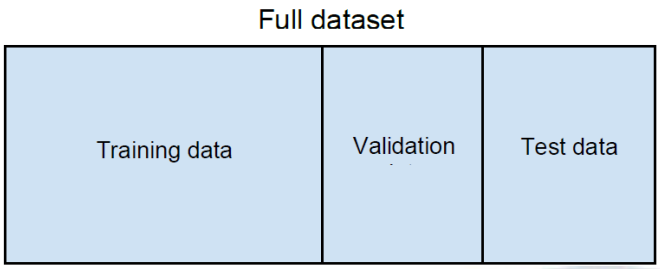


Figure 5-Division of dataset for training

3.2.3. Dropout

Dropout has recently become the preferred method for preventing overfitting [6]. It works very differently compared to others regularization technique and does not rely on modifying the cost function. Instead, dropout modifies the network itself, by temporarily disabling a random subset of the hidden neurons, as shown in Figure 6. [12] Dropout is particularly effective at reducing overfitting in very large and deep networks, where overfitting is often severe.

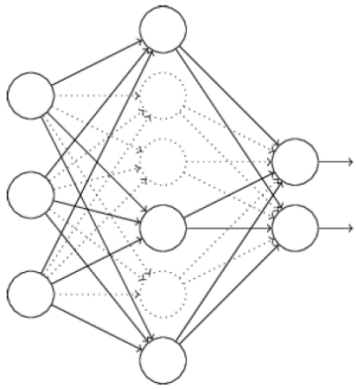


Figure 6-Half of the neurons in the hidden layer are disabled

The input is forward propagated and the result backpropagated through the reshaped network. For the next iteration, the disabled neurons are restored, a new random subset is disabled, and the cycle repeats itself. [12] This results in a procedure that rather resembles training several different networks and averaging their results, without actually having to expend the resources to train a large number of complete networks. The neurons in the network cannot rely on any other specific neurons existing, so they must learn features that are beneficial with many random groups of neurons, instead of forming relationships with some particular set of neurons .Because the weights and biases of the network are trained with only half of the neurons active during each iteration, the weights have to be halved when the full network is finally run.

3.2.4. Batch Normalization

We already discussed the importance of the number of input example or training sizes. The size of the training data can be hundreds of gigabytes of data. These data cannot be processed at once with the processors. The deep networks process training/input data in batches of appropriate sizes the CPU and/or GPU can handle the computations. The distribution of the input data is different for each layer. Hence, the input data changes on each layers of every iteration of the training, which adjusts parameters accordingly. One approach to get around this issue is to initialize the parameters with well-crafted weights and have a very small learning rate at the same time. This causes the computing resources to be used for longer durations. This phenomenon is known as covariate shift. Another way to deal with the problem is to the normalize the input data. In batch normalization the input into each layer is normalized before further processing. In some cases, the batch normalization method eliminates the need to use dropout as the regularizer.

3.2.5. Activation Functions

An activation function scales the input and thereby decides when a neuron is activated, that is passing information to the next layer. The rectified linear unit (ReLU) is an activation function, which has become increasingly popular due to its simplicity and high performance. The function is zero for all negative inputs and equal to the input if it is positive, as expressed below

f (x) = max (0, x) (6),

where f (x) is the activation function dependent on the input [10]. A graphical representation of this function can be seen in figure 7.

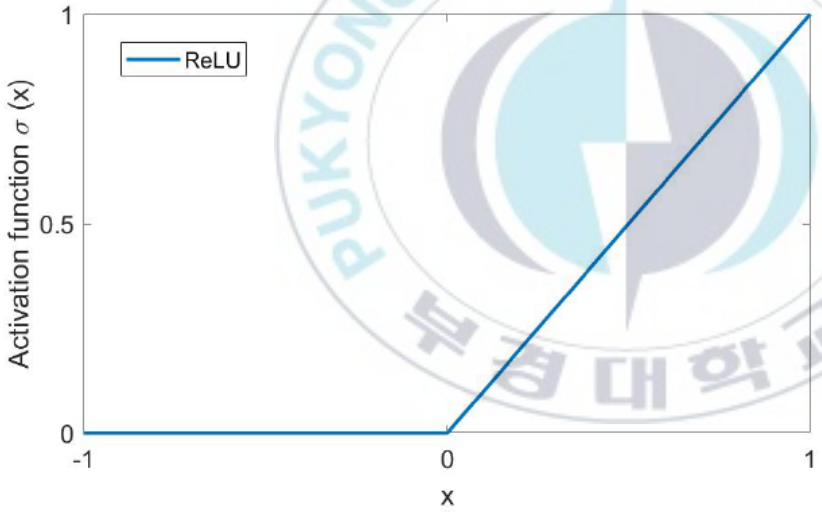


Figure 7-Activation function ReLU as a function of x.

Softmax (z) = $\frac{z_k}{\sum_{i=1}^{k} z_i}$ (7),

where Z is a vector containing the input to the function and k 2 [1,K] where K is the number of classes [13]. Softmax gives an output between 0 and 1 which can be interpreted as a class probability [10].

3.2.6. Optimizer

The Adam optimizer was selected because it has been shown to converge quickly and perform well at image classification tasks. Adam is particularly effective on deep CNNs where the gradients on different types of layers vary greatly because it is able to adapt the learning rates for individual weights. The initial learning rate was set to a relatively small value at 0.0001. The authors recommend a value of 0.001 which is also the default value intensorflow but after testing, the lower value seemed to provide more reliable results while still converging in an acceptable time.

3.2.7.Softmax

For classification tasks, it is helpful to have the output vector be a probability distribution over a set of mutually exclusive classes. For example, in the task of recognizing handwritten digits, the labels 0-9 are mutually exclusive. We can use a softmax layer to produce a probability distribution vector of the form $[p1\ p2\ \ldots\ pn]$ to provide a notion of how confident we are in the predictions. An output vector with one value close to 1 and the other values close to zero signifies a strong prediction, while an output vector with multiple values being close to equal signifies a weak prediction.

The softmax function is defined as:

$$\text{Softmax } (z)_j = \frac{e^{z_j}}{\sum_{i=1}^{k} e^{z_k}} \quad (8),$$

for $j = 1,2,\ldots, K$. As the output of the softmax layer is a probability distribution, with the sum of the outputs being 1, the output of one neuron depends on the outputs of all the other neurons in the layer.

IV. Deep learning

Deep learning is a subset of machine learning and as other applications appear in machine learning, to solve any problem, we begin first to define in details the kind of problem we need to overcome. This consists of determining the vector forms of the inputs and potential outputs. For example, to classify fish images from QUT dataset, the input would be the pixel intensity values of the 112x112x3 pixel images, and softmax should be used the one which to apply for classifying fish images into different classes from 1up to 76.

The internal structure of the network is defined next. The preferred architecture depends on the type of problem and includes deciding the number of hidden layers, the types of neurons, connections and so on. Different kinds of neural network architectures have been found to work well on different types of problems, for example, feed-forward convolutional neural networks are widely used for image recognition tasks while recurrent neural networks work well for tasks involving natural language processing [14]. Figure 8 depicts a general workflow for creating any kind of neural network.

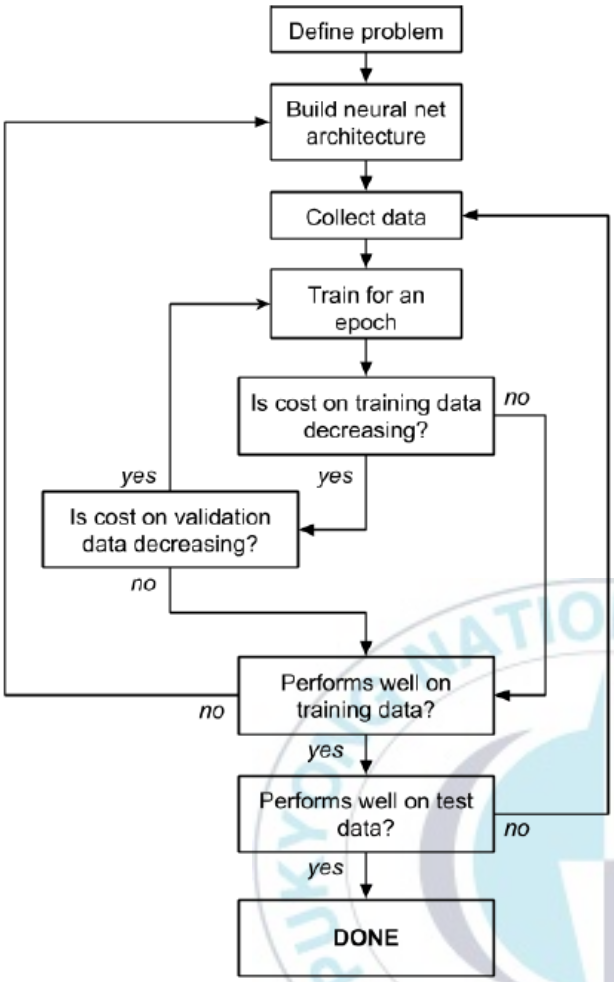


Figure 8-Detailed workflow for training and testing a deep learning model.

Proposed methods

4.1. Convolutional neural network

In proposed system, we are building a simple convolutional neural network (CNN) of three layers that can identify and classify the images into 76 classes. In general, most of the machine learning applications requires GPU (Graphics processing unit) because of high number of computations on large amount of data. Since GPUs have almost 200 times more processors than CPU, it improves the performance of neural networks. As the number of layers in the network increases, the number of computations increases and hence the need for GPUs increases.

Therefore, in order to build a neural network that can work on CPU as well a very small network is build. There are various software libraries that have focused on machine learning. Few of them are Theano, Scikit-learn and Tensorflow. In proposed system, a convolutional neural network based fish classifier is built using Tensorflow which an open source software library focused on machine is learning. It is implemented using python language and windows 10 system. Python is used because of its simplicity and ease to learn. It provides various tools that are helpful in making machine learning applications. In this work, a QUT fish dataset is used. However, the same network can be used to train other datasets also. The QUT fish dataset consists of 1140 labelled images. In this experiment, 855 fishes are used as training data and 285 fishes are used as testing data. Convolutional neural network has input layers, output layers and hidden layers. The hidden layers are consists of convolutional layer, flattened layer and a fully connected layer. The Figure 9 shows the architecture of the proposed convolutional neural network.

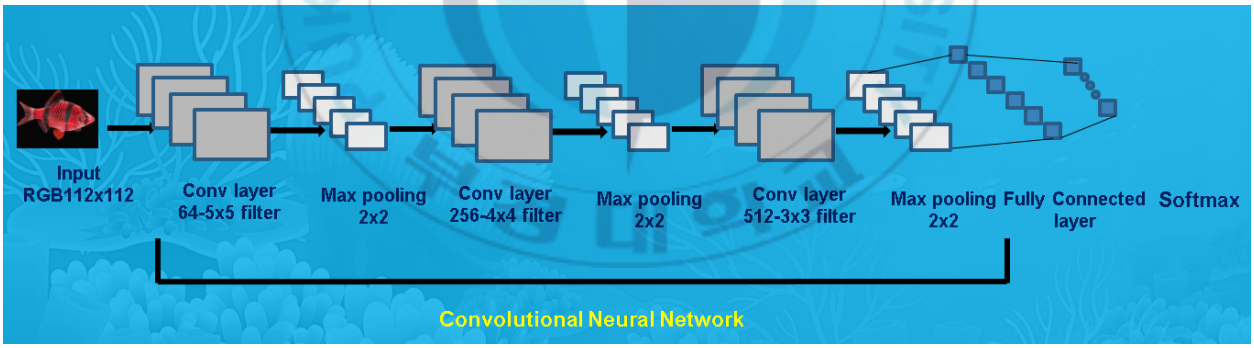


Figure 9-CNN architecture

4.1.1. Input layer

The input layer works directly on the original input data, and for the input fish, the input data is the pixel value of the fish. The input image is resized to 112 by 112 and has tree channels, which is very useful in image classification.

4.1.2. Convolutional Layer

Convolutional layer is the building block of convolutional neural network. The principal goal of convolutional layer is to extract features. It is built of one or more convolutional layers followed by max or average pooling layer to reduce the dimensionality. My design structure supports 3D structure of input such as fishes. This is built by applying local connections. The input to the convolutional layer is an n*n*p fish where ‘n’ is height and width of fish and ‘p’ is the number of channels. The convolutional layer will have filters of size z*z*p where ‘z’ is smaller than the size of fish. A filter sized from the fish is selected and convolution is calculated with the filter. This will result in a single number as output to which a bias is added. Here convolution (element-wise product) is nothing but matrix multiplication of z*z*p sized of image and z*z*p sized filter. Filter is slided over the whole input image to calculate the output across the fish. The number of pixels through which the sliding takes place is called stride. All these outputs are concatenated to have an activation map or feature map. After each convolution, the size of the fish decreases. Therefore, it is a standard practice to add zeros on the boundary of input layer such that the output is same as input layer. This is called as padding.

$$\begin{bmatrix} O_{11} & O_{12} \\ O_{21} & O_{22} \end{bmatrix} = \text{Correlation}\left(\begin{bmatrix} X_{11} & X_{12} & X_{13} \\ X_{21} & X_{22} & X_{23} \\ X_{31} & X_{32} & X_{33} \end{bmatrix}, \begin{bmatrix} F_{11} & F_{12} \\ F_{21} & F_{22} \end{bmatrix} \right)$$

$$O_{11} = F_{11}X_{11} + F_{12}X_{12} + F_{21}X_{21} + F_{22}X_{22}$$
$$O_{12} = F_{11}X_{12} + F_{12}X_{13} + F_{21}X_{22} + F_{22}X_{23}$$
$$O_{21} = F_{11}X_{21} + F_{12}X_{22} + F_{21}X_{31} + F_{22}X_{32}$$
$$O_{22} = F_{11}X_{22} + F_{12}X_{23} + F_{21}X_{32} + F_{22}X_{33}$$

Figure 10-Convolutional layer

4.1.3. Pooling layer

Pooling layer is used to reduce the dimensionality of the feature map in order to reduce the processing time. The main purpose of pooling layer is to reduce the spatial size (height, width). This reduces the number of parameters; hence, the number of computations is also reduced. There are three different types of pooling. They are max pooling, average pooling, min pooling. The most commonly used pooling is max pooling where we take a filter of size f*f and apply the maximum operation over the f*f sized part of image. Mostly pooling is done with a filter of size 2*2 with a stride of 2.This method reduces the image size into two parts .

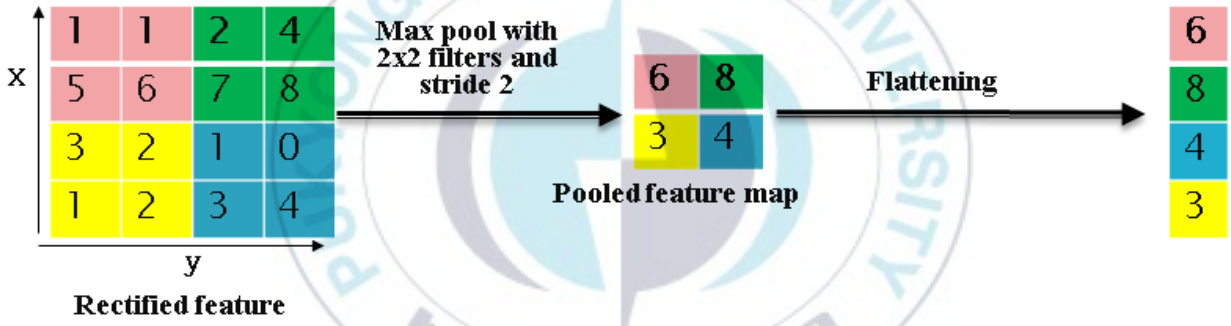


Figure 11-pooling layer

4.1.4. Flattening Layer

The output of a convolutional layer is a multi-dimensional Tensor. A flattening layer is used in order to convert this into a single dimensional tensor. This is done using the reshape operation of tensorflow framework. It gets the output from the previous convolutional layers and flattens its structure to create a single feature vector which can be used by the fully connected layer to perform classification.

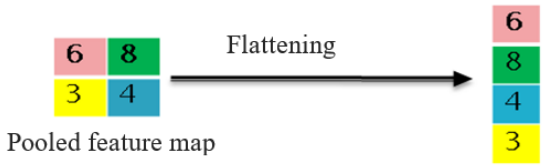


Figure 12-Flattened layer

4.1.5. Fully connected layer

This layer performs the classification of the fish based on the features extracted by the previous convolutional layers. In fully connected layer, every neuron is connected with every neuron of previous layer. A softmax function is used to convert the output of neural network into probability for each class. After deciding the network architecture, focus should be given on parameters of the network. The best set of parameters can be found using back propagation technique. In this technique, random set of parameters are used at first. These values are changed such that for every training image we get correct output. Gradient descent is an optimizer method that is quick in finding the correct parameters. Cost is a single number that indicates the accuracy of the classifier increases as the cost decreases. Therefore, training is done till the cost remains constant. After training is done, the parameters and architecture are saved in a binary file called as model. A new fish is sent as input to the same network and the probability of the new fish is calculated. This is called as prediction.

Instead of feeding the whole training data to the network, it is divided into batches of fishes called as epochs. Each epoch may contain 16 or 32 fishes and it takes more than 50 iterations to train the whole dataset. A convolutional neural network is built and trained with the fishes from the chosen QUT fish dataset. The process of training takes place with the help of CPU only. The

output of this phase would be a classifier which can be used to predict the class of the given input fish in prediction phase.

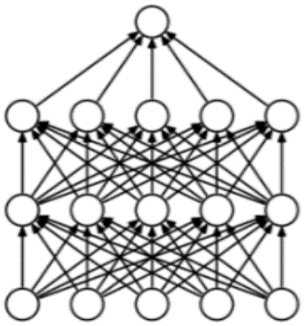


Figure 13-Fully connected layer

4.1.6. Output layer

The number of output neurons is set according to the specific application task. If the task is for classification and recognition, the output layer is usually a classifier. A softmax function is applied to transform the output of layer into the probability for each class.

4.2. Recurrent neural networks (RNN)

RNN is one class of deep learning and is applied on data with a sequential structure. The method has a loop which is helpful the information to be kept and to flow in different steps of the system. The recurrent neural network figure is illustrated below:

Recurrent neural networks (RNN) are a type of neural networks which are mostly applied on data with a sequential structure. The networks contain loops which allows information to persist and be passed between the different steps of the network. RNNs can be viewed as copies of the same network and a common visualization can be seen in figure 14.

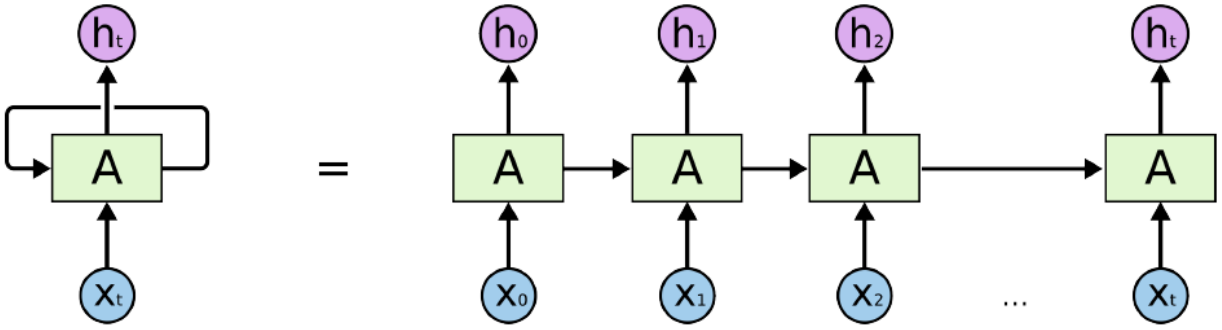


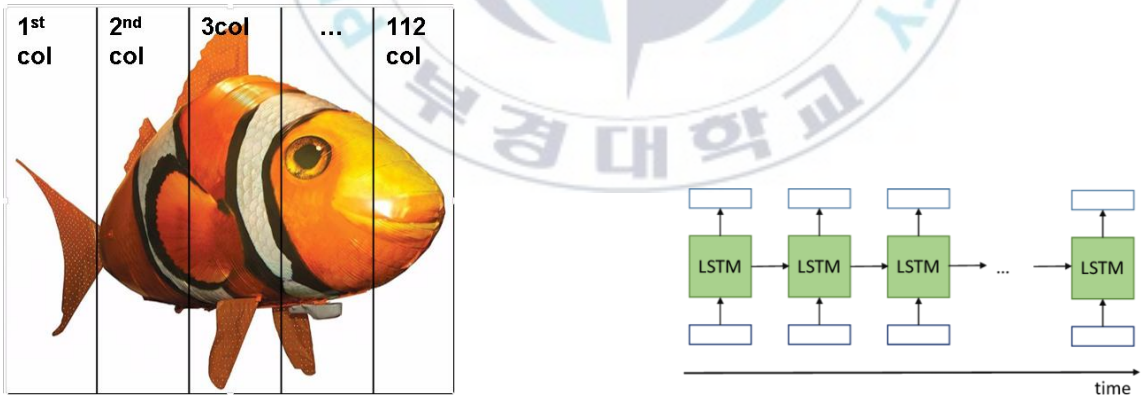
Figure 14-recurrent neural network, where A is a cell of the network.

x_t: input
h_t: output
A: $cell$

4.3. Long Short-Term Memory (LSTM)

For comparison, in this research, LSTM has been applied to compare with CNN.

LSTM is useful for sequence data, then image can be viewed as sequence of columns. Therefore, LSTM can catch the patterns of sequence of columns.



4.4. Dropout

The output of the last pooling layer acts as an input to the so called fully connected layer. Usually there are many nodes and edges that may lead to overfitting. Fortunately, during training, regularization technique called ‘dropout’ [16] can intervene to overcome this issue by removing nodes and edges which are not very active in classification stage. Let us see the figure 15:

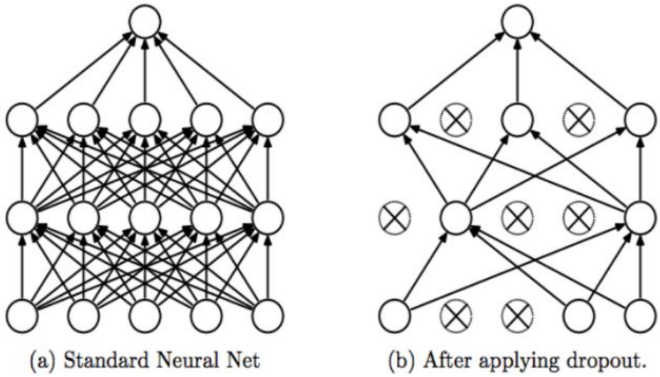


Figure 15-Before and after applying dropout method

4.5. Top k candidates

Top k candidates is one of the methods helpful to classify images.

Due to difficulties in many classification problems, creating or returning top likely candidates is often a preferred method for future post-processing. It is applied during testing to return multiple answers instead of a single best. About my chosen structure, I pick five top candidates to be returned during testing. The following figure is an example returned three candidates.

Top 3 candidates

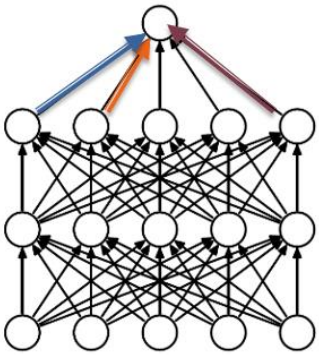


Figure 16-: Top k candidates

V. Experiments and results

5.1. QUT Fish Dataset

In this research we take QUT fish image dataset. This is used to compare two different deep neural networks structures. This dataset has been also used in [16] for object classification. It is a collection of 1140 fish images from 76 different categories. To classify these fish images is difficult because some fishes from the same class can look differently (different color, different size, different orientation, different background,…).The difficulties for fish images classification come from several causes: the fishes from different classes can look similar and the illumination is another challenge.

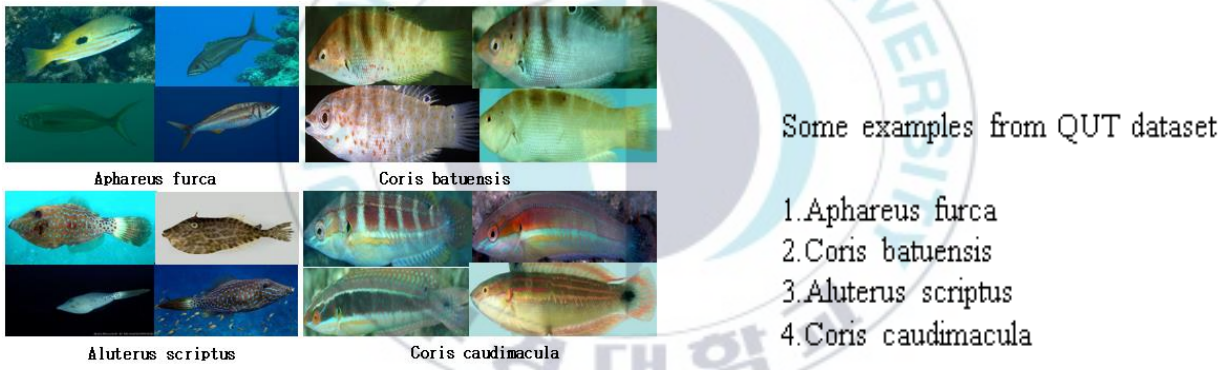


Figure 16-Some samples from QUT

5.2. Results

Deep neural networks with various numbers of layers and filters have been tested for fish classification. From two up to four layers have been applied in the design. However, due too large number of connections, training is often difficult. Several methods such as dropout and top k candidates were adopted to overcome this issue. With three convolutional layers, we observed

a performance of 53 % with single best without dropout but the figure rose to 59% with 0.5 dropout. Indeed, the performance increases gradually with increasing numbers of candidates. From single best to top five, when it reaches 81%.For comparison, the cases of two layers and four layers did not work as well, respectively 80% and 65%.In CNN the number of filters played an important role in the fish classification task. See Table 1, the third row. A comparison of CNN with three layers and the LSTM with 500 layers gives 81% and 47%.

Table2-Convolutional layered structures.

Experiments	Number of layers	Filters sizes	Number of filters	Dropout rate	Top k candidates	Test accuracy
1	2	5x5x3 4x4	128 512	0.5	1	58%
					5	80%
2	3	5x5x3 4x4 3x3	64 128 256	0.5	1	53%
					5	75%
3	3	5x5x3 4x4 3x3	64 256 512	0	1	53%
					5	62%
4	3	5x5x3 4x4 3x3	64 256 512	0.5	1	59%
					5	81%
5	4	5x5x3 4x4 3x3 3x3	64 256 512 512	0.5	1	53%
					5	65%

Table 1: Proposed CNN compared with LSTM

Experiments	Methods	Top k candidates	Number of hidden layers	Test accuracy
1	CNN	5	3	81%
2	LSTM	5	**500**	**47%**

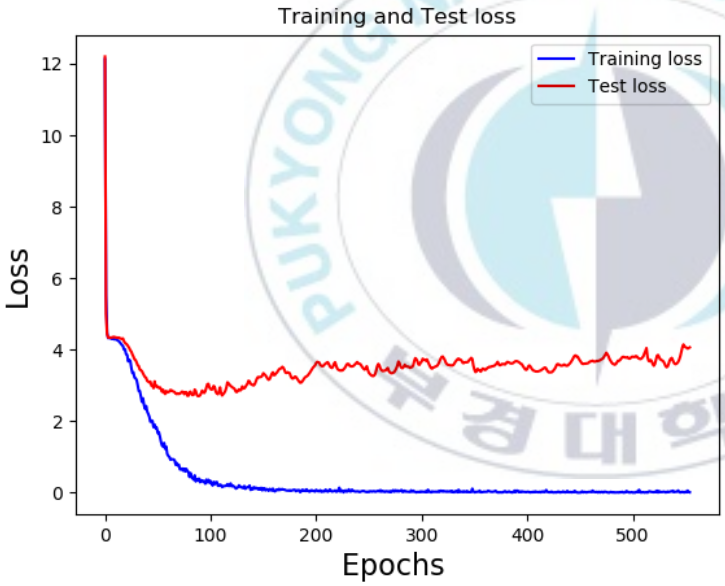


Figure 17-Illustration on how test loss decreases

The red cover tell us how the error decreases .After 60 iterations the training should be stopped but the high accuracy was reached after 550 iterations. So, it would be better to wait several iterations to obtain a high performance.

VI. Conclusion

In this dissertation, we have applied several methods to classify fish images.

Convolutional neural network, dropout and top k candidates method have been proposed as solution for fish images classification. According to our knowledge, LSTM is useful for sequence data and any image is viewed as sequence of columns, so LSTM can catch the patterns of sequence of columns. Indeed, for comparison, LSTM has been tested for fish images classification .After applying many experiments, it is found that CNN is the preferred method recording 81% by applying three convolutional layers [64, 256, 512 filters], with 0.5 dropout and top5. However, by applying two layers [128,512 filters], with 0.5 dropout and top 5, the performance is similar (80%) with three layers. By conclusion, the variety of filters number is very important in classification.

The future work would be comparing this work with Generative Adversarial Network (GANs), because it is one of the methods applied to augment the small dataset.

Neural network model needs many images to train and then be capable to recognize a new image and classify it.

Reference

[1] G. Ding, Yan Song, J. Guo, C. Feng,G. Li, T.Yan, “Fi sh Recognition Using Convolutional Neural Network”,*IEEE*,oceans 2017- Anchorage.
[2] M.Sarigül and M. Avci, “Comparison of Different Deep Structures for Fish Classification”, *International Journal of Computer Theory and Engineering*, Vol.9, no.5, pp.362-366, October 2017.

[3] Chollet F. Deep Learning with Python [Internet], Manning Publications; 2017 [cited 2018 Maj 17] Available from: https://manning-content.s3.amazonaws.com/download/4/ 02b02cf-c1ac-47ae-8ffd-25f85daad877/SampleCh01.pdf

[4] Goodfellow I, Bengio Y, Courville A. Deep Learning [Internet], MIT Press; 2016 [cited 2018 Mar 28] Available from: http://www.deeplearningbook.org

[5] S. Ahmed, S. Ul Azeem, and Q. Hua, “*Machine Learning with TensorFlow 1.x.”* Packt Publishing, 2017.

[6] AI456. 2013. Error surface of a linear neuron with two input weights, *Wikimedia Commons*, checked one October2018. <https://commons.wikimedia.org/wiki/File:Error_surface_of_a_linear_neuron_with_two_input_weights.png>.

[7] Aphex34. 2015. Input volume connected to a convolutional layer, *Wikimedia Commons*, checked 1 October 2018. <https://commons.wikimedia.org/wiki/File:Conv_layer.png>.

[8] Bettina Berendt and Christoph Hanser. 2007. Tags are not metadata, but “just more content” — to some people. In: *Proc. of the International Conference on Weblogs and*

Social Media.

[9] Dahal P. Classification and Loss Evaluation - Softmax and Cross Entropy Loss [Internet]. Deep Notes, 2017 Maj 28 [cited 2018 Apr 01] Available from: https://deepnotes.io/Softmax-crossentropy.

[10] Lindsten F, Svensson A, Wahlström N, Schön T. Lecture Notes [Internet]. Uppsala University Department of Information Technology [updated 2018 Feb 02; cited 2018 Mar. 18] Available from: http://www.it.uu.se/edu/course/homepage/sml/literature/lecture_notes.pdf.

[11] P. Kingma D, Lei Ba J. Adam: A Method for Stochastic Optimization [Internet]. Proceedings of International Conference for Learning Representations, 2015, San Diego [updated 2017 Jan 30; cited 2018 Apr 17] Available from: https://arxiv.org/pdf/1412.6980.pdf.

[12] Alhussein Fawzi, Horst Samulowitz, Deepak Turaga, and Pascal Frossard. 2016. Adaptive data augmentation for image classification. In: *Proc. of the IEEE International Conference on Image Processing (ICIP)*, 3688-3692.

[13] Wahlström N. Statistical Machine Learning 2018, Lecture 8 - Deep learning and neural networks, [unpublished lecture notes]. Uppsala University; notes provided at lecture given 2018 Feb 22. [Cited 2018 Apr 03] Available from: http://www.it.uu.se/edu/course/homepage/sml/lectures/lecture8_handout_updated.pdf.

[14] Olah C. Cola's blog [Internet]. Christopher Olah. 2014-.Understanding LSTM Networks; 2015 Aug 27 [cited 2018 Apr 03] Available from: https://colah.github.io/posts/2015-08-Understanding-LSTMs.

[15] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, “Dropout: A Simple Way to Prevent Neural Networks from over fitting,” *Journal of Machine Learning Research,* pp. 1929-1958, June 2014.

[16] K. Anantharajah, Z. Y. Ge, C. McCool, S. Denman, C. Fookes, P. Corke, etal, “Local Inter-Session Variability Modelling for Object Classification,”*IEEE Winter Conference on Applications of Computer Vision,* pp.309-316, June 2014.

Acknowledgment

First, I would like to give thanks and praise to the Almighty

God for his grace and blessings throughout the entire project. I would also like to express my deepest gratitude to my supervisor:

Professor **Sin Bong-Kee** for guiding me throughout this project and giving me invaluable advice.

I am extremely thankful to my wonderful parents and my brothers and sister for their love and support.

I thank all the Imedia Lab members, past and present for their kind corporation.

I also place on record my gratitude to Nkenyereye Lewis for helping me to get my admission in PKNU.