UNIVERSITY OF SCIENCE AND TECHNOLOGY OF HANOI

ICT DEPARTMENT

MASTER THESIS

by

Viet Kieu Quoc

ICT.M8.003

Information and Communication Technology

Title:

Skin Cancer Classification using Deep Learning

Supervisors: Dr. TRAN Giang Son

Lab name: ICT Lab

August 21, 2024

ATTESTATION

I hereby, Viet Kieu Quoc, certify that my report doesn’t contain plagiarism (copy/paste) from other sources.

In case of plagiarism in my report, I know the consequences and I understand that my report won’t be evaluated. In this case, my M2 internship will be noted as "fail".

August 21, 2024

Signature

Viet Kieu Quoc

Acknowledgements

First and foremost, I would like to express my deepest gratitude to Dr. TRAN Giang Son, my research supervisor for his patience, helpful theoretical and practical advice and enthusiastic encouragement during the period of my internship. Words cannot express my feelings, nor my thanks for all his help. It was an honor that he became my supervisor for the last few years.

I would also like to show gratitude to Dr. Nghiem Thi Phuong for her devoted instructions, indicators and continuous encouragement. Thank you very much for introducing me this interesting research topic and helping me over the project.

I would like to thank the University of Science and Technology (USTH) in general and ICTLab in particular for teaching me a lot of valuable knowledge and experiences, and also giving the opportunity to work in a professional environment as one of the temporary member of the lab.

Thanks should also go to the StackOverFlow community for answering many questions of mine.

Last but not least, I would like to appreciate all the help and support from my family and friends during my internship.

Contents

List of Acronyms

AK Actinic keratoses. 6, 8, 9, 31–33

BCC Basal Cell Carcinoma. ii, 1, 2, 6, 8, 31, 33

BKL Benign keratosis-like lesions. 6, 8, 31, 33

CAD Computer Aided Diagnosis. iv, 4, 36

CNN Convolutional Neural Network. 12, 15, 22, 23, 34, 35

DCNN Deep Convolution Neural Network. 5

DF Dermatofibroma. 6, 8, 31–33

DL Deep Learning. 5

ISIC International Skin Imaging Collaboration. iv, 4–6, 8, 9, 24, 34–36

MEL Melanoma. ii, 1, 3, 6, 8, 31, 33, 34

NV Melanocytic nevi. ii, 6, 8, 31, 33, 34

PPV Positive Predictive Value. 26

ReLU Rectified Linear Unit. 13, 14

SCC Squamous Cell Carcinoma. ii, 1, 2, 6, 8, 9, 31–33

SVM Support Vector Machine. 5, 12

UNK Unknown. 9, 31, 33

UV Ultraviolet. 1

VASC Vascular lesions. 6, 8, 31, 33

List of Figures

List of Tables

Abstract

Melanoma is one of the most malignant, metastatic and dangerous types of skin cancer that causes a majority of deaths related to skin cancer. It was estimated that in 2018 there were about 91,270 new cases of skin cancer from melanoma with 9,320 deaths. Scientists indicated that melanoma is a curable disease if it is diagnosed early and correctly. Due to this, it is necessary to examine and observe melanoma closely when it is still at the early stage.

In order to detect skin cancer from melanoma, besides clinical tests, dermatologists often use their eyes to examine characteristics of skin lesions such as color, texture and shape to diagnose if the lesion is a benign or malignant tumor. Nowadays, advances in technologies allow the widely use of dermoscopy images in examining and diagnosing melanoma skin cancer. To support this task, many Computer Aided Diagnosis (CAD) systems are designed to detect melanoma from dermoscopy images.

One important step of the CAD system for melanoma skin cancer is to classify if the melanoma skin lesion is benign or malignant. Due to this, many methods are proposed in the literature to detect malignant melanoma from skin lesions. Therefore, Deep learning has been proved as a popular and powerful method in many medical imaging diagnosis areas. In the context of this project, we focus on the study of machine learning and deep learning methods for the classification of multiclass skin cancer such as malignant melanoma or nevi from dermoscopy images. We test on ISIC dataset with 2 models: InceptionV3 and Xception. The evaluation shows that there is a great disparity between 2 models and the best one is the Xception model with 95% accuracy.

Keywords: *Skin cancer, Melanoma, Neural Network, ISIC, Deep Learning, Computer-aided Diagnosis*

Chapter 1

Introduction

1.1 Context and Motivation

Definition. Cancer is a common name for a set of related diseases. In any cancer, there are abnormal cells, which are normal cells of the body, but now divide, multiply uncontrollably, invade surrounding tissues and spread away (called metastasis).

It can occur almost anywhere in the body's one trillion cells [28]. Normally, cells grow and divide to form new cells, the way the human body grows and develops. Of course, old cells will eventually age or be damaged, die, and be replaced by new cells.

When cancer appears, the natural process is disrupted. Cells become more and more abnormal. Old cells do not die but continue to grow, constantly producing new cells. They just multiply uncontrollably, and eventually form an abnormal mass that we call a tumor.

Currently, more than 100 types of cancer have been identified, of which the prevalence of each disease varies with age, sex and race [17]. For example, most breast cancer occurs in women and prostate cancer occurs only in men. But in this topic we will focus on the skin cancer type.

Skin cancer is the abnormal growth of skin cells in the epidermis due to untreated DNA damage. It is also the main cause that triggers mutation and leads the skin cells to multiply quickly and then form malignant tumours. Skin cancer develops primarily on areas of sun-exposed skin, e.g., the face, ears, neck, chest, arms and hands, and legs. The sun's unhealthy Ultraviolet (UV) rays and the utilization of UV tanning machines are the two most leading reasons that lead to skin cancer [12]. Since UV radiation is a human carcinogen, having *5 or more sunburns* will increase the risk of melanoma. It can affect anyone regardless of skin color. When comes to later stages, patients that have skin cancer with dark skin color are often misdiagnosed and it will be difficult to cure. Particularly, every single day nearly 10,000 people in the U.S are diagnosed with skin cancer, approximately 3 millions of American get affected of non-melanoma skin cancer, and more than 5.4 million skin cancer cases are diagnosed in more than 3.3 million people in the US each year according to the research. According to estimation, around 200,000 new cases of melanoma that includes half of noninvasive and half of invasive were diagnosed in 2019 [13].

Categories. The most common types of skin cancer are Basal Cell Carcinoma (BCC) (approximately 80%), Squamous Cell Carcinoma (SCC) (around 16%), Melanoma (MEL) (4%). Basal Cell Carcinoma (BCC) is more common in men and can be anywhere on the body, most commonly on the head, neck and face. Initially, the keratin will appear, then it will spread and infiltrate. The rock surface of the horn has a slit. BCC grows slowly and does not metastasize, however, there

are also cases where extensive damage destroys skin organs, which can spread deeply, destroying blood vessels, pinching nerves, etc. (fig. 1.1)

Figure 1.1 – Basal Cell Carcinoma Symptom

The 2nd one, Squamous Cell Carcinoma (SCC), occurs based on existing lesions. After the wound spreads, the horny layer will thicken, the surface is ulcerative and penetrates deep under the skin, on the edge of the wound, there are red pants and meat buds emerge. The disease tends to develop and invade lymphocytes and surrounding organs and causes destruction of large blood vessels, nerves. After a while it may appear lymph node metastases, often seen metastasis of lungs, liver, less common in bones and internal organs. (fig. 1.2).

Figure 1.2 – Squamous Cell Carcinoma Symptom

Both BCC and SCC belong to the group of non-melanoma skin cancer. They are diagnosed by clinical images and histopathology images, together with the differential diagnosis to make the most accurate conclusions about the condition. Depending on the condition of the patient and the shape of the disease, appropriate treatment methods should be applied, as follows:

- Cut lesions in rhombohedral or elliptical shapes by surgical methods.
- Burning laser.
- Radioactivity, radiotherapy (X-rays, radium rays).

Finally, Melanoma belongs to malignant form, developed from melanin-producing cells, is common in middle-aged women. The disease can appear anywhere on the body, usually at places frequently exposed to the sun. The common symptom of the disease is the appearance of abnormalities on the skin such as hyperpigmentation or old moles with a change in shape, size, etc. (fig. 1.3).

Figure 1.3 – Melanoma Symptom

Currently, the diagnosis of melanoma is based on biopsy of the lesion, as follows:

- Full lesion biopsy.
- Biopsy of part of the injury.
- Drill biopsy with a press.

Diagnosis. Early diagnosis of a particular strain of cancer is a urgent essentials for the patient's successful treatment. Dermoscopy is a skin imaging methodology that has shown improvement for conclusion of skin malignant growth contrasted with independent visual assessment. In any case, clinicians ought to get sufficient preparing for those enhancements to be figured it out. According to an estimation of the Skin Cancer Foundation, the number of people in United States that was diagnosed and cured from non-melanoma skin cancers grown by 77% from 1994 to 2014, and for melanoma group, it will increase by at least 2% in 2020[13]. That will result in the amount of deaths will decrease by 5.3%. Additionally, nearly 200,000 cases of melanoma will be diagnosed in the U.S in 2020. Half of them will be in noninvasive that surrounded to the epidermis (the top layer of skin), and the rest will be invasive, entering the epidermis into the skin's subsequent layer (the dermis)[13].

For early diagnosis skin cancer, the doctor can:

- Skin examination: The doctor may look at the skin to determine if the changed skin is more likely to have skin cancer. Further testing may be needed to confirm the diagnosis.

- Discard a suspect skin sample for test (skin biopsy): The doctor may remove a small sample of suspicious-looking skin for laboratory testing. A biopsy can determine if there is skin cancer, and if so, what type of skin cancer there is.

If the doctor determines to have skin cancer, he or she may recommend further tests to determine the extent, or stage, of the skin cancer. Because superficial skin cancer such as basal or squamous cell carcinoma rarely spreads, a biopsy is often the only test required to determine the stage of the cancer. But if there is a large growth or one that is persistent for a while, the doctor may recommend further tests to determine the extent of the cancer.

Despite the fact that an expert can be trained to achieve an accuracy in symptom detection and treatment in types of skin malignant growth up to roughly 80% [21], the medical sector is currently lacking dermatologists for skin caner. Yet, another problem arises is that, to analyze these million images of skin cancer is a huge workload for the doctors due to the high variance of size, shape, texture, location between the healthy skin and the damaged skin. This is an error-prone task.

In order to make expertise more generally accessible, International Skin Imaging Collaboration (ISIC) has developed the their own Archive, an international repository of dermoscopic images, for the purposes of clinical training and also for boosting the development and effectiveness of Computer Aided Diagnosis by hosting the International Skin Imaging Collaboration Challenge. By using computer to classify the skin cancer, we can minimize the workload of dermatologists during the clinical stages. Recent years, machine learning, especially deep learning based algorithms have become a brilliant choice for analyzing medical images. A deep convolutional algorithm can be more effective, accurate and reproducible when it has been well trained.

1.2 State-of-the-art

In the last several years, some methods have been proposed to solve the skin cancer problems such as skin cancer detection and skin cancer classification, or even using preprocessing techniques like segmentation to increase the accuracy of detecting and classifying skin lesions. The ISIC challenge in 2016 introduced different methods as they were submitted for skin cancer segmentation, feature extraction and classification tasks. The best report published a comparative study and showed remarkable accuracy of segmentation and classification results, 95.3% and 91.6% respectively. Another study proposed by Simon Kalouche when he did his final project for the Stanford CS 229: Machine Learning course which was taught by Prof. Andrew Ng [19]. His team was focus on utilizing computer vision-based deep learning methods to detect skin cancer, typically melanoma. They trained on 3 separate models including a logistic regression model, fine-tuned VGG-16 and multi-layer perceptron deep neural network to obtain an outstanding classification accuracy. Their outcome shows that their algorithm was able to segment moles and classify skin lesions is from 70% to 80%. At that time, the classification tasks had 2 types of label only, benign and malignant [9]. In the next year 2017, the dataset was bigger and that time, it had 3 main tasks, segmentation, detection and classification and named the challenge ISIC 2017. There were some unique approaches gave better performance with different DCNNs. The Fully Convolutional Network (FCN) ensemble gave the best segmentation performance in term of accuracy and dice coefficient which are 93.4% and 84.9% respectively.

Nevertheless, in 2015, there was a new architecture called **U-Net**, introduced specially for medical image segmentation tasks. And until now, it is still popular and prove the efficiency in various modalities of medical imaging and computational pathology. Even though it performs with small dataset, U-net still gives precise segmentation's result [26]. In 2018, an improved version of U-net called R2U-Net, which stands for Recurrent Residual Convolutional (RRN) U-Net, was announced [24]. It was tested on different datasets relate to medical image segmentation tasks on ISIC 2017 Challenge. The results were compared against other models including SegNet[30], Residual U-Net (ResU-Net) and it appeared to be improved considerably[24]. Also in the same year, there was another model proposed called LadderNet[31]. This network is a chain of multiple U-nets combined of multiple encoding, decoding units and is tested for a retinal blood vessel segmentation task [31].

In term of classification task, the lesions are usually extracted with classical methods like *Color Feature, Edge Histogram* and non-linear SVM for classification. DCNN based methods (e.g. ResNet, ImageNet [18]) are applied to automatically extract feature from input samples. The methods were tested on ISIC 2016 Challenge Dataset and the results were compared against 8 expert dermatologists on 100 subsets of test images. It surpassed all of them [8]. In 2018, ISIC 2018 Challenge was hold with the title *The skin lesion analysis towards melanoma detection with deep learning methods*[1]. Some of the Deep Learning (DL) methods were evaluated by using ISIC 2017 Dataset and, the best outcomes were 0.718 dice coefficient for segmentation, 0.833 scores for feature extraction, and 0.823 scores for classification tasks [22].

Typically, researchers attempt to preprocess the data before feeding them into a network to train. One popular technique is called data augmentation, a method to increase the number of data samples. Another preprocessing technique is proposed by Farooq et al. from National University of Ireland. Their proposed study includes 2 main phases, the 1st phase is to enhance the image quality by removing the clutters and generating a refined version of training images. This is done by applying a sharpening filter followed by a hair removal algorithm. They use some

image quality measurement metrics, i.e. Peak Signal to Noise (PSNR), Mean Square Error (MSE), Maximum Absolute Squared Deviation and Energy Ratio / Ratio of Squared Norms (MXERR) to analyze the image quality before and after applying preprocessing function. The 2nd phase is to train the upgraded dataset version by using the deep learning models including InceptionV3 [5] and MobileNet[2]. During the process, both models show a remarkable development in train and validation accuracy after using the refined version of images. Nevertheless, InceptionV3 outperforms MobileNet in term of better validation accuracy, so it was chosen to evaluate on test data, and the final result was 86% of accuracy. [11]

1.3 Internship Objectives

The ISIC 2019 Challenge contains 2 different tasks which are 1) classify dermoscopic images without meta-data, and 2) classify images with additional available meta-data. During 6 months of this internship, we focus on the 2nd task and conduct research in multi-class *skin cancer classification*, specifically 9 classes including Melanoma (MEL), Melanocytic nevi (NV), Benign keratosis-like lesions (BKL), Basal Cell Carcinoma (BCC), Actinic keratoses (AK), Vascular lesions (VASC), Dermatofibroma (DF), Squamous Cell Carcinoma (SCC) an additional outlier class not represented using different deep learning models. We also study about the skin cancer images, and based on these available image data to classify the label among all of them.

1.4 Thesis organization

The content of this thesis includes five main chapters:

- Chapter 1: introduces the concepts of the internship including background knowledge related to Cancer, Skin Cancer and State-of-the-art.
- Chapter 2: reviews materials used in this internship.
- Chapter 3: briefly describes different methods carried out during the internship period.
- Chapter 4: evaluates our proposed method, discusses the obtained results and shows limitation of the work.
- Chapter 5: concludes the report and presents future works.

Chapter 2

Materials

2.1 Dermoscopy Image Data

Dermoscopy (also known as Dermatoscopy), is a study of skin lesions using a dermatoscope. The dermatoscope is a handheld device using visible light that helps clinician, dermatologists to watch and analyze skin lesions without the impediment of skin surface reflections, and to discriminate benign from malignant tumors, particularly in the diagnosis of melanoma. Basically, the principle of dermoscopy is transillumination of a lesion, magnify from 10 to hundreds of times, break down refraction, enabling the doctors to see sub-macroscopic structures in the epidermis and top 1 or 2 millimetres of dermis [16].

Figure 2.1 – Performing dermoscopy using Dermatoscope.

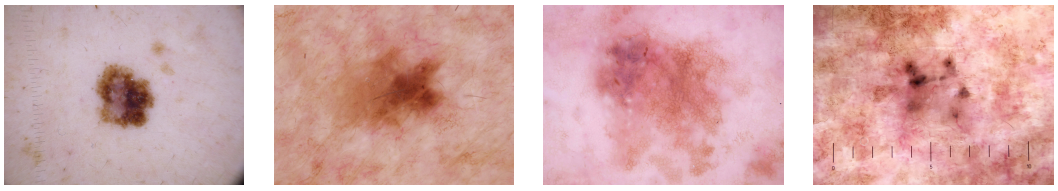
In 1920, a German Dermatologist Johann Saphier introduced the term "dermoscopy" to the world. The first person to use it for pigmentary lesions was Goldman in 1950s. Since then, it has been widely studied and applied to aid in the treatment of disease symptoms. Officially, in the early 1990s, Dermoscopy first began to gain widespread acceptance and use among dermatologists. This marked the advance in clinical diagnosis because it improves the recognition of a growing number of skin symptoms in general dermatology.

2.2 Dataset

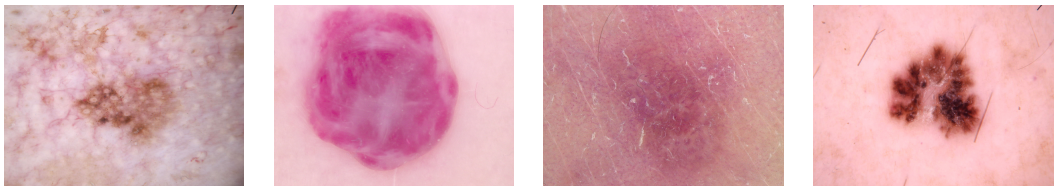
All the data is downloaded from the ISIC 2019 Challenge[1]. The images are collected from the publicly available reference archive for skin lesions: International Skin Imaging Collaboration (ISIC) and also from reputed hospitals. It is an open challenge for everyone to join, test their own models on the same database with a standardized evaluation protocol. It is also to present problems in lesion segmentation, detection of clinical diagnostic patterns, and lesion classification. It was organized from 2016 and still continue until now, plus it attracts many people globally, hundreds up to thousands of registrations of submissions, making them the biggest normalized and comparative study in this field. ISIC 2019 has 2 main tasks: classifying dermoscopic images without meta-data and classifying images with additional available meta-data. ISIC 2019 data is provided courtesy of the following sources: BCN_20000 Dataset © Department of Dermatology [23], Hospital Clínic de Barcelona; HAM10000 Dataset © ViDIR Group, Department of Dermatology, Medical University of Vienna [29]; MSK Dataset © Anonymous [7, 6]. The following table describes different characteristics of the *ISIC 2019* dataset:

Table 2.1 – ISIC2019 Training Dataset

Label	Number of images	Ratio
Melanocytic nevi (NV)	12875	50.83%
Melanoma (MEL)	4522	17.85%
Benign keratosis-like lesions (BKL)	2624	10.36%
Basal Cell Carcinoma (BCC)	3323	13.12%
Actinic keratoses (AK)	867	3.42%
Vascular lesions (VASC)	253	1.00%
Dermatofibroma (DF)	239	0.94%
Squamous Cell Carcinoma (SCC)	628	2.248%



(a) Melanoma (MEL) (b) Melanocytic nevi (NV) (c) Benign keratosis-like lesions (BKL) (d) Basal Cell Carcinoma (BCC)



(e) Actinic keratoses (AK) (f) Vascular lesions (VASC) (g) Dermatofibroma (DF) (h) Squamous Cell Carcinoma (SCC)

Figure 2.2 – Dataset Image Examples

[1] https://challenge2019.isic-archive.com/data.html

Ensuing the previous tendency, the ISIC 2019 dataset is not only bigger but also much more difficulty for classification task. The same with last year, it still have unbalanced parts, a major issue while training any machine learning models in general. On the other hand, the class *AKIEC* in 2018 was divided into 2 separate parts, specifically, *AK* and *SCC*. Participants were provided with 25,331 images for training across eight different categories. Moreover, no images were provided for class 'None of the others' and it only be available during the test time. The image-only test dataset includes 8,238 images across eight diagnostic categories and one 'out-of-distribution' class not represented in training dataset. This will make the whole challenge much harder to deal with. Fig. 2.2 shows one example image for each of the eight skin diseases in the training dataset.

The size of the images inside each part of the training dataset is divergent: The *HAM10000* dataset contains 12,413 images of size 600x450 that were centered and cropped around the lesions; while the other dataset *BCN_20000* has 10,015 images of size 1024x1024, these images are challenging since they are uncropped and also skin lesions are in different locations, scale and angle to spot out; and the last one *MSK* with 819 images accommodates various sizes.

Figure 2.3 – Examples from datasets. From left to right: MSK, HAM10000, BCN_20000 Example

The metadata provided for the training set consists of the approximate age, the location of where the skin lesion was found and a unique identifier for each lesion of each patient. With the provided metadata, *there are duplicate entries per image* and thus patient meaning that *it may be possible to have more than one class per patient*. Due to the nature of the challenge where it is guaranteed that there is only one disease that exists per patient, we have taken the liberty of isolating out the images and thus patients that have only one identified lesion when consulting the metadata. Therefore, this decreases the amount of image data we are dealing with which will thus make training easier.

In the task description given by the ISIC host, no approach of how to deal with the problem of detecting unknown class was stated out. So we will likely deal with it by adding images from other sources which does not belong to one of the training categories, and theses samples will be treated as a separate dataset. We prepare this dataset which contains around 600 images and will be trained apart from the rest dataset.

Major Difficulties:

- Imbalanced categories: There are only 239 images in the smallest class (least amount of pictures), and up to 12,875 in the largest class.
- Disease areas vary in size: some pictures the area of the disease is small, some pictures the disease fills the entire picture
- No images belong to the UNK class in the training set, but our model must classify this class as well since the test set contains these images. This will more closely test the model in real use cases.

2.3 Hardware Infrastructure

With the advancement of machine learning, it is indispensable to mention graphics cards as they make great contributions to fast, efficient and high-performance processing. GPU's processing speed is developing every single day to help us solve the problem with high complexity much faster. During this internship, our experiments (training, testing) are conducted using ICTLab's infrastructure with high-performance computers, the detail can be described as follow:

- CPU: Intel(R) Xeon(R) CPU E5-2620 v3 @ 2.40GHz
- GPU: GeForce RTX 2080 Ti
- RAM: 32GB

Chapter 3

Methods

3.1 Data Pre-processing

Deep learning generally requires a large amount of input data and also they need to be balanced to achieve considerable accuracy. The best way to improve our model is to get more data. One common technique to deal with the problem of small and imbalance data is to generate the image by performing extensive data augmentation. It helps us to increase the size of the data set quickly while changing and adding new properties to the existing data to make the data abundance more abundant. Also it is the key aspect in improving the prediction scores since it acts as a regularizer and therefore prevents overfitting. Thanks to **ImageDataGenerator**[2] class, we can execute numerous methods to generate a bunch of tensor image data with real-time data augmentation. Some of the techniques to augment training data are described as below:

- Randomly flipping horizontally and vertically
- Zoom in the inputs: 10%
- Random rotation in range from 0° to 180°
- Shift horizontally and vertically: 10%

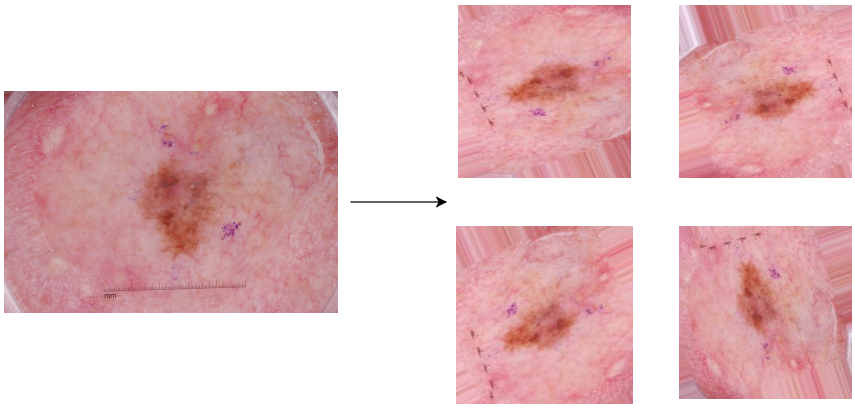


Figure 3.1 – Example of Preprocessing

Because of the unbalanced dataset that was mentioned in section 2.2, we decide to increase the number of image to the number that every category equally has the same size.

[2]https://keras.io/api/preprocessing/image/

3.2 Convolutional Neural Network

Machine learning algorithms such as Support Vector Machine (SVM) are often used to detect and classify images. However they are frequently constrained by the suspicions we make when we characterize features, it also costs lower precision. Nevertheless, neural networks, could be ideal solution since these algorithms can learn features from image data and increase the precision of the computer and the performance of image detection and classification.

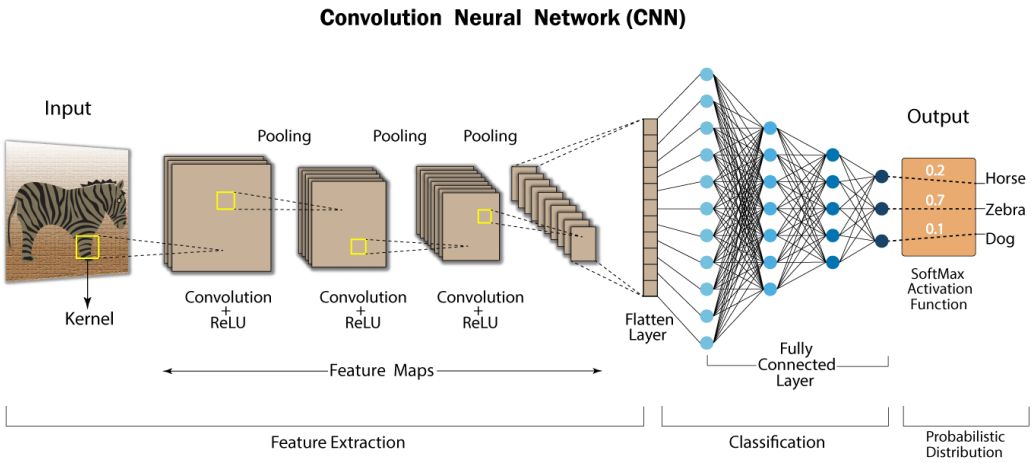


Figure 3.2 – General CNNs architecture.[3]

A Convolutional Neural Network (CNN) is a multilayer neural network (fig. 3.2), comprised of an input and output layer, as well as multiple hidden layers. The hidden layers of a CNN consists of one or more convolution layers, pooling layers, activation layers to extract the features and then followed by one or more fully connected layers to classify. These are the basic layers of every CNN architectures.

CNN is one of the advanced Deep Learning models that helps to build many intelligent systems with high accuracy nowadays. It is also used in many fields from recommender system, natural language processing, computer vision to image and video recognition, etc. Many companies like Facebook, Google used CNN in their products for certain tasks, such as fingerprint and face recognition, self-driving car, automatic flying drone, etc.

Convolutional Layer. Among all the layers in a network, convolutional layers are the most important one. It is the core building block of a CNN that does most of the computational heavy lifting. Convolution is a mathematical operation to merge informations. An example of an input and a convolutional kernel is illustrated on fig. 3.3.

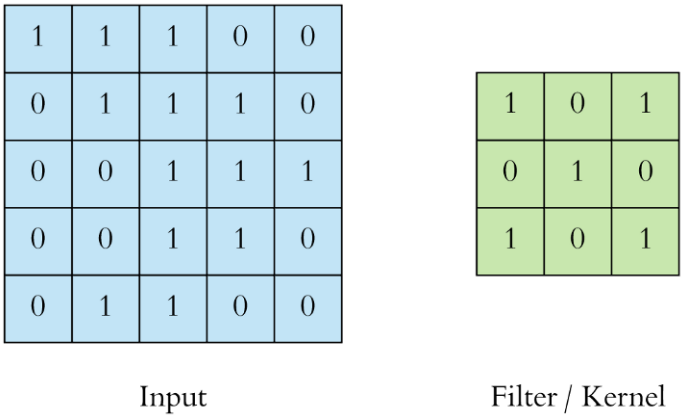


Figure 3.3 – Example of Input and convolution kernel.

The matrix will go over the input matrix from left to right, from up to down, after that multiply every corresponding values of input matrix and kernel matrix, then sum the result to get a specific number. The set of these numbers is another matrix called feature map. The green area where the convolution operation takes place is called the receptive field. The deeper the convolutional layers are, the larger the receptive fields become and the more complex the features are learned [10]. Output of the sample input from fig. 3.3 is illustrated on fig. 3.4.

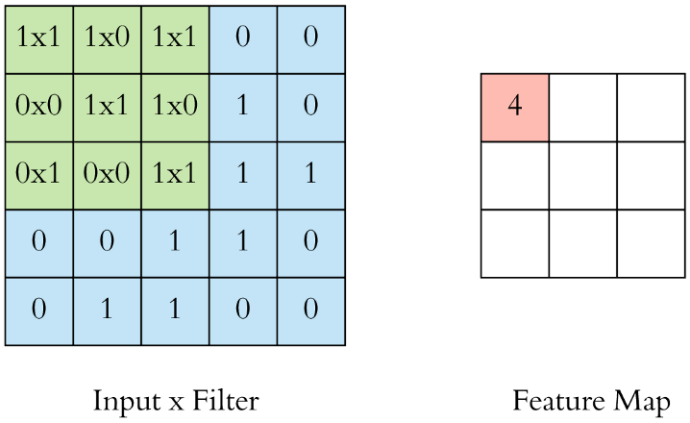


Figure 3.4 – Output of Convolution Operation

Activation Function. Activation functions are nonlinear functions that are applied to the output of neurons in the hidden layer of a network model, and are used as input data for the next layer. Activation function is generated for the purpose of *breaking the linearity* of the neural network. Without the nonlinear activation functions, our neural network, even with many layers, would be as effective as a linear layer and cannot achieve its true potential.

One of the most famous activation function and being used, nearly, in all kind of models and also pretrained models, is Rectified Linear Unit (ReLU). ReLU layers are also playing an important role beside the convolution layers since they can transform the value of every neuron from linear to non-linear. Compared to other function like logistic sigmoid or hyperbolic tangent, ReLU is the better choice since it's easier to implement and significantly faster (several time faster than their equivalents with `tanh` units). Faster learning is the key importance of ReLU. The function is defined as follows:

$$f(x) = \begin{cases} x & \text{for } x \geq 0, \\ 0 & \text{for } x < 0. \end{cases} \tag{3.1}$$

or simply expressed as: $f(x) = max(0, x)$. (fig. 3.5)

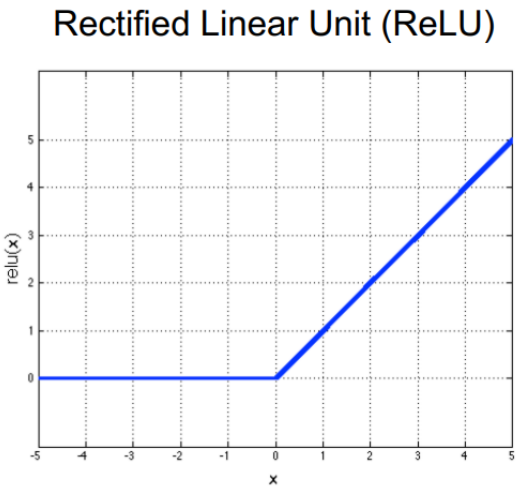


Figure 3.5 – ReLU function

Softmax. Another special case of activation functions that being used mostly other than ReLU is Softmax. Softmax is a way of constraining the output of neural networks to sum to 1. Therefore, the output values of the softmax function can be considered as a probability distribution of the output variables. It is very useful in multi-class classification problems. To do this, Softmax function converts the neural network output value by dividing the total value. The output can now be considered as a vector of the classes's predicted probabilities. We can see more clearly in the following formula:

$$Softmax(x)_i = \frac{exp(x_i)}{\sum_j exp(x_j))} \tag{3.2}$$

where x_i is the input vector and x_j is the output vector.

We can use Euclidean distance to compare the distance between one-hot encoding and softmax to serve the loss function construction and optimization of neural network parameters. However, the cross entropy function discussed below is one of the most commonly used loss functions and has proven to be effective in multilayer classification problems.

Binary Cross-Entropy loss. Binary Cross-entropy is used to compare the distance between the output values of softmax and one-hot encoding. Cross-entropy is a loss function and its value can be minimized. This helps neural networks evaluate the probability (the certainty) of predicting a data sample corresponding to a class. The probability will be maximum for our target variable. Cross entropy is the sum of the negative logarithmic probabilities. We can define it with the following formula:

$$Loss_{BCE} = -(p(x) * logq(x) + (1 - p(x))log(1 - q(x))) \tag{3.3}$$

Where $p(x)$ is the probability of class X in TARGET and $q(x)$ is the probability of class X in PREDICTION. It is applying the natural logarithm to the difference between the prediction and the target. The logarithmic function with the negative value was used to minimize the loss function (maximize the logarithmic function in the same formula, while minimizing its negative value).

Categorical Cross-Entropy loss. If Binary Cross-Entropy loss was applied for 2 classes then we have Categorical Cross-Entropy loss for Multi-class Classification. It also refers as Softmax Loss since it is a Softmax activation *plus* a Cross-Entropy loss. When training the CNN model, to output the probability of each image over the C classes, we will use this loss.

In typical (usually most of it) situation of Multi-Class classification, the labels in ground truth data are specified as one-hot vector, meaning it has 1 on a single position and 0's everywhere else, so the positive class C_p will keep in term of the loss and only one element of the Target vector t is not zero, $t_i = t_p$. When we remove all the components that equal to zero because of target labels, the final formula of Categorical Cross-Entropy can be described as below:

$$E_R(\theta) = -log(\frac{e^{\theta_p}}{\sum_j^C e^{\theta_j}}) \tag{3.4}$$

Where θ_j is the CNN score for the positive class, j refers as the iterator number, C is number of classes.

Stride and Padding. Stride is the distance between the two kernels when scanned. By default stride $= 1$, the kernel will scan two adjacent cells, but with stride $= 2$, the kernel will scan cell number 1 and cell number 3. Ignore the middle box. This is to avoid duplicating the values in the scanned cells. (fig. 3.7)

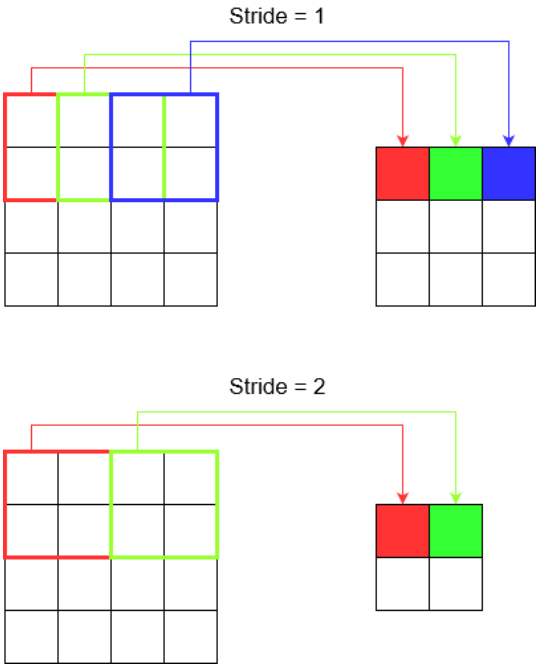


Figure 3.6 – Stride Example

We choose the stride and the larger the size of the kernel, the smaller the size of the feature map, partly because the kernel must be entirely in the input. There is a way to keep the size of the feature map from its original size called **Padding**. When we adjust padding $= 1$, it means we

have added a wrap around the edges of the input, if we want this wrap to be thicker, we need to increase the padding.

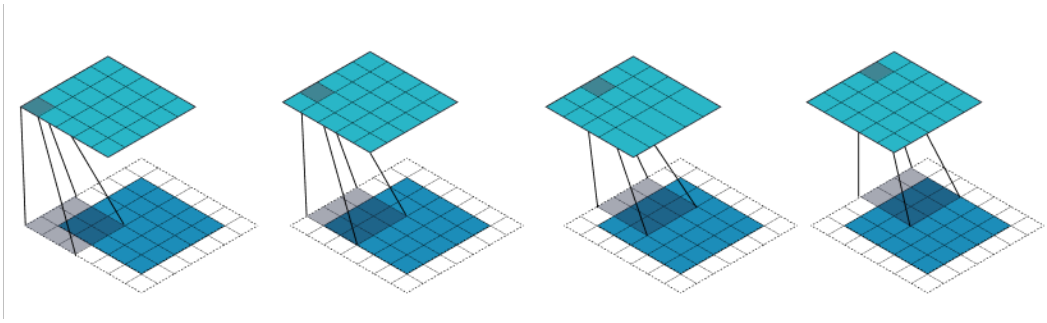


Figure 3.7 – Padding Example

With stride $= 1$ and padding $= 0$, from the original input image, we will scan the kernel through and form the following cells to map to feature map as Fig.3.8 as below.

Figure 3.8 – Example of Feature Map

Pooling layers. Pooling, a form of non-linear down-sampling, often placed between the convolutional layers. It is used to reduce the spatial dimensions (and consequently the number of trainable parameters to learn) but not depth on a CNN model. Basically, after using pooling layer, the network:

1. Reduces the computational cost.
2. Less spatial information also means less parameters, so more chance to avoid overfitting problem - the networks are too fit to the training dataset but has low performance in testing data.

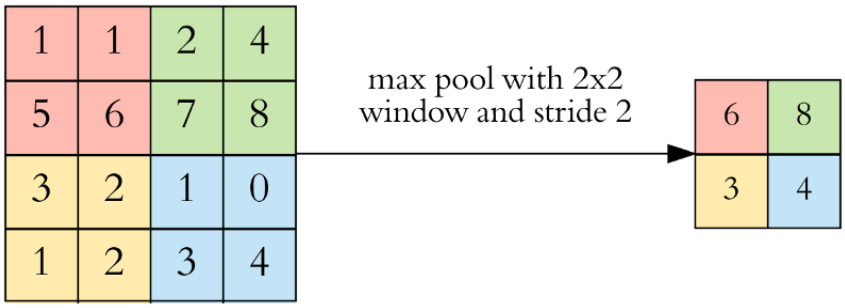


Figure 3.9 – Example of max pooling with filter size 2x2

There exists different pooling techniques, such as minimum, maximum or average. Nowadays, Max-pooling is being used a lot in CNNs, it performs down-sampling by separating the input into same-size rectangular pooling regions, and computing the maximum of each region. Fig. 3.9 illustrates a max pooling example.

Dropout layer. The term *dropout* refers to dropping out units (both hidden and visible) in a neural network.

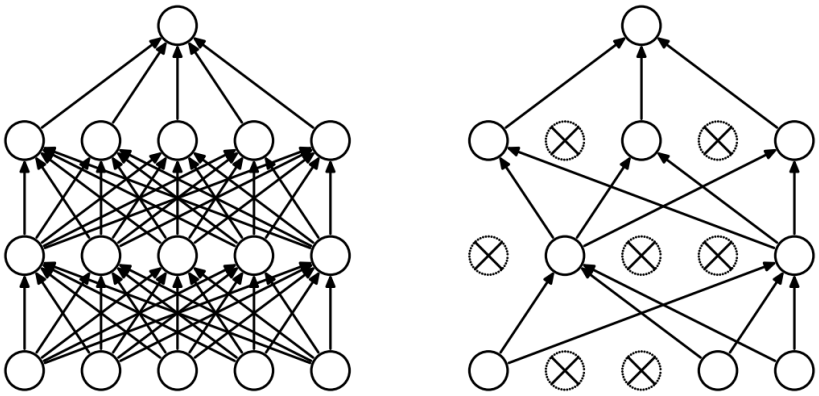


Figure 3.10 – Dropout Neural Net Model. [25]

At each training stage, when performing forward propagation to the layer using drop-out, instead of calculating all the nodes on the layer, at each node we will randomly see if the node is calculated or not based on the probability p (fig. 3.10). This forces the layer to learn with different neurons the same *concept*, improving generalization, and also to preventing overfitting.

Fully-connected layers. Neurons in a fully connected layer have full connections to all neurons in the previous layer, as seen in regular artificial neural networks. Their activations can hence be computed with a matrix multiplication followed by a bias offset.

Normally, after the convolution and pooling layers, we add a couple of fully connected layers to wrap up the CNN architecture. One layer to gather the feature layers, and then convert the data from 3-D, or 2-D into 1-D. The other layer is the output, the neuron's number of this layer depends on the output we want to find. For example, MNIST dataset, we have a set of handwritten numbers from 0 to 9, so the output number is 10.

Pointwise convolution. When we build a deep learning model, one problem we have is that the number of channels of feature maps increases with the depth of the network, which results in too many number of parameters that will lead to overfitting or increase computational complexity. We can have pooling layer like max pooling, avg pooling, etc. helps us to reduce the size of the width and height of a feature map while remaining the required feature. However, such pooling layers cannot help us increase or decrease the number of channels, but we can if using Pointwise convolution. It is a convolution with first two dimensions equal to 1, and the third dimension equals to the number of channels at the input.

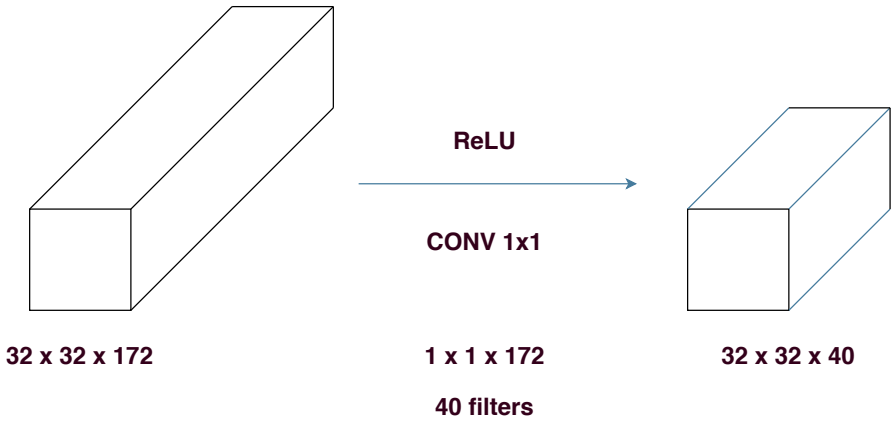


Figure 3.11 – Pointwise Convolution

Take an example with the fig. 3.11 above, we have input is a matrix of size 32x32x172, we also have a pointwise convolution of size 1x1x172. We multiply the input by pointwise convolution and choose the number of filters which is the number of channels at the desired output we want (here is 40), after that we will have an output matrix of size 32x32x40. The width and height do not change. This helps to control the channel size as we like.

Depthwise Separable Convolution. Depthwise Separable Convolution consists of two phases:

1. **Depthwise Convolution.** Depthwise Convolution have the idea that instead of multiplying convolution with the entire channel like traditional convolution, it will divide the input map features into groups with *a fixed number of channels equal to 1*. For example image below, according to traditional convolution, we have input of size 7x7x3 multiplied by a kernel of size 3x3x3 which will give output of size 5x5x3. Depthwise Convolution will divide the kernels into 3 small kernels of size 3x3x1 and multiplying these small kernels together with 3 groups, each small group of 7x7x1 produces 3 outputs of 5x5x1. We can see that when we overlap those three outputs, we get the output with the size of 5x5x3, exactly like the traditional way but we can perform better because we use 3 separate kernels, at the same time the number of parameters in filters is also reduced significantly.

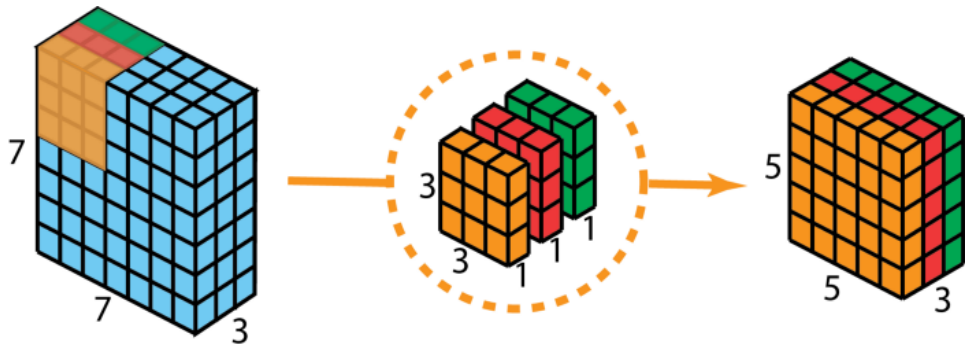


Figure 3.12 – Depthwise Separable Convolution

2. **Pointwise Convolution.** After implementing *Depthwise Convolution*, then we can multiply convolution by *Pointwise Convolution* to customize the number of channels as desired which was mentioned above.

   Last step is to generate the final image with the shape we desire, for instance, produce 128 kernels with the size 1x1x3 to output a 5x5x1 each and acquire the final image of shape 5x5x128 (fig. 3.13).

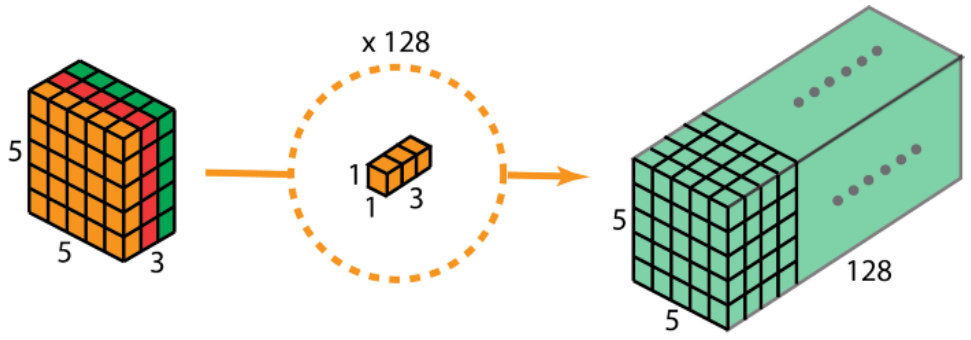


Figure 3.13 – Pointwise convolution with 128 kernels, outputting an image with 128 channels

The main reasons that make a clearly difference between *Traditional convolution* and *Depthwise separable convolution* are *Multiplication* and *Transformation*. Let's take an example with the example above, we have 128 3x3x3 kernels that move 5x5 times. In Traditional convolution case, that would mean 128x3x3x3x5x5=86,400 multiplications which the computer needs to do. For Depthwise separable convolution, in 1st phase, the depthwise convolution has 3 3x3x1 kernels that move 5x5 times which will be considered as 3x3x3x5x5=675 multiplications; 2nd phase, the pointwise convolution, we have 128 1x1x3 kernels that move 5x5 times, that counts 128x1x1x3x5x5=9,600 multiplications. Summing up, the number of calculations the computer has to do for Depthwise separable convolution is 10,275 multiplications only, shorter than Traditional convolution. The lower computations, the faster the process can make.

What happens behind all of this even though both of the convolutions are doing the same thing (proceed the image through a 3x3 kernel, reduce it to 1 channel only, and then multiple it to 128 channels) is *Transformation*. In Traditional case, the image is transformed 128 times, each transformation uses up 3x3x3x5x5=675 multiplications. While in Depthwise separable convolution, we transform the image *only once* - in the 1st phase, Depthwise Convolution; next step after we have the transformed image, we just extend it to 128 channels. Not only do we have to transform the image repeatedly but also we save up a lot of computational power.

3.3 Transfer Learning

Various well-known models, have been trained on large datasets (MNIST, CIFAR-100, ImageNet, ...) [27], and the model's source code and Weights are publicized to the public (mainly on GitHub). We call such Weights Modeling as a Pretrained Model. The new model uses part or all of the pretrained model as part of it to learn a new tasks called the Transfered Model.

Deep learning models contain extensive number of parameters and training data. To retrain a new one will cost intensive computation and take a long time since it requires enormous or massive resources, and they converge slowly in general. By using *transfer learning*, the model converges quickly, and we can get a better model based on our own idea modifications. *Transfer Learning* is also a way for models to communicate the capabilities that each model can do. A model can learn on certain source tasks and then this pretrained model is used for another model so that new model learns on target tasks faster (fig. 3.14).

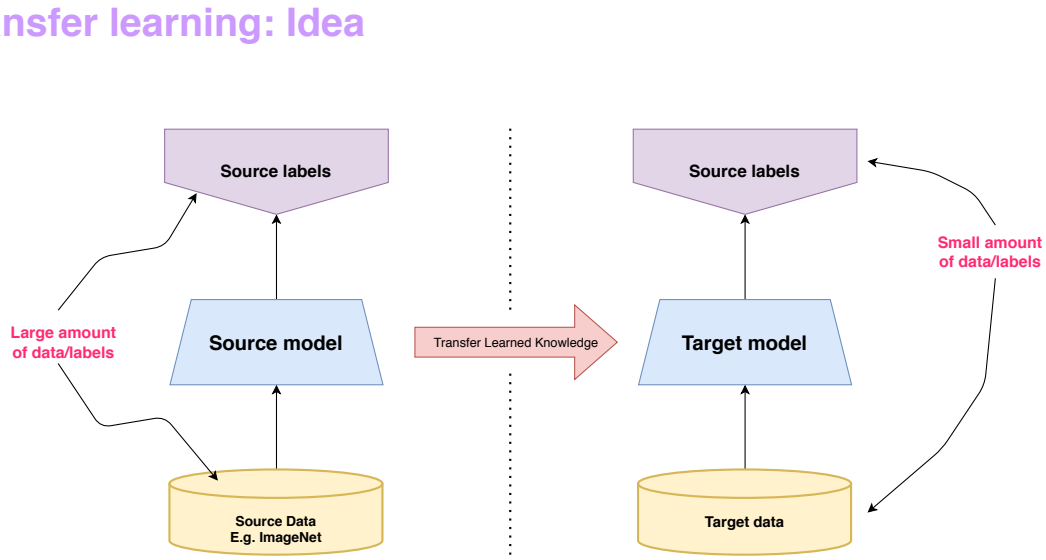


Figure 3.14 – Transfer Learning Strategy.

Simply put, Transfer Learning in Deep Learning:

- is a machine learning technique where a model that is trained on a task is reused on a second related task.
- is an optimization that improves speed and performance when modeling second tasks
- relates to issues like multi-task learning and concept drift.

We can use transfer learning in our own predictive models in two popular approaches:

1. Develop Model
   (a) *Select Source Task.* We to choose a predictive model that is related to the data in which there are some relationships in the input data, the output data, and (or) the concepts learned during the process of mapping input data to output.
   (b) *Develop Source Model.* Develop the model for the 1st task and need to ensure that some learning features have been implemented.

(c) *Reuse Model.* The model that is suitable to the source task can then be used as the starting point for a model on the second task of interest. This may involve all or part of the model, depending on the modeling technique used.

(d) *Fine-tune Model.* The model may need to be fine-tuned on the input-output data available for the task of interest.

2. Pre-trained Model (Most popular one)

(a) *Select Source Model.* A pretrained source model is chosen from the available models. Many research organizations offer models on large datasets we can choose from (E.g. Google's Inception-V3, Oxford University's VGG, Microsoft's ResNet).

(b) *Reuse Model.* The model that is suitable to the source task can then be used as the starting point for a model on the second task of interest. This may involve all or part of the model, depending on the modeling technique used.

(c) *Fine-tune Model.* The model may need to be fine-tuned on the input-output data available for the task of interest.

Even though Transfer Learning is a shortcut to save time or get better performance, we should only apply it in the following cases, because in other cases just build model like other machine learning problems can still give such positive results:

1. *Higher Start.* The initial skill (before fine-tuning model) on source model is higher.
2. *Higher Slope.* There is a large difference in the model improvement rate during training.
3. *Higher asymptote.* Convergence ability of the pretrained model is better.

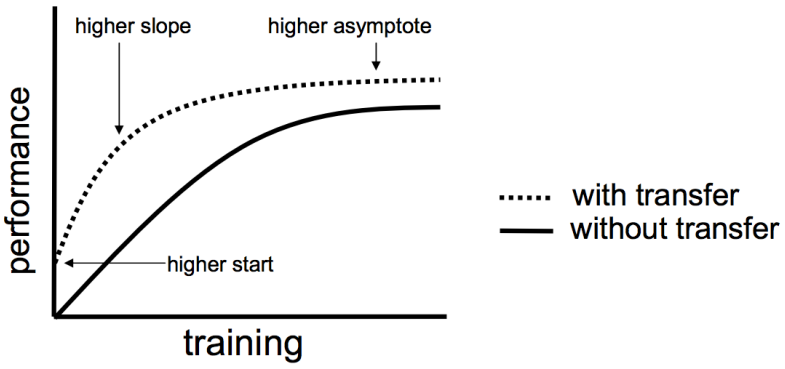


Figure 3.15 – Transfer Learning Benefit.

The use of the pretrained model is a huge step for others to follow in their predecessor's accomplishments, leveraging existing pretrained models to create new models for more specific target tasks and more practical application. It's not a copy of the idea, the creators of the pretrained model themselves publicize their success hoping that others can benefit from the models, or at least use it to to deal with their affairs.

3.4 Fine-tuning

To use the pretrained model effectively, we need the following 2 things:

- Add layers that match our target tasks, remove the layers of the pretrained model we don't use, but make the model more efficient. This is a difficult problem (very difficult) requires in-depth studies of each layer and their purpose.
- Having a very good training strategy is not easy either, because if we train badly, it will lose the effectiveness of the pretrained model and thus reduce the capabilities of the model we are training, even worse than train all over again.

Therefore, fine-tuning was born to help we have an effective training strategy on our transfered model. Fine-tuning not only helps us to adjust the transfered model's weights to suit our target tasks, it also provides an optimal way to train both the pretrained model and the new part in the transfered model to achieve high accuracy on target tasks, making the two parts fit together to complete a new model.

3.5 Model Architecture

3.5.1 InceptionV3

Inception-V3 could be considered as a breakthrough in the development of Deep Learning history and also a method of neural network design derived from Google. It won the 2014 Imagenet competition and was designed to improve the speed and size of the model. Inception's idea was to build big network architectures by incorporating multiple micro-architectures. Each hidden layer incorporates several higher-level representations of the image. In each layer they can use several different types of kernels instead of just using one type of kernels. Average pooling is used with a number of different sizes when concatenating the above layers. A special feature of Inception is that the kernel parameters can be learned during training. Using some kernels, the model can define small features like abstract information by itself. In particular, the 1x1 convolution layer reduces the characteristics and calculation speed, thereby making the inference process faster. We can take a look at the simplest inception module with several convolution classes with different kernel sizes and pooling classes with the fig. 3.17 below:

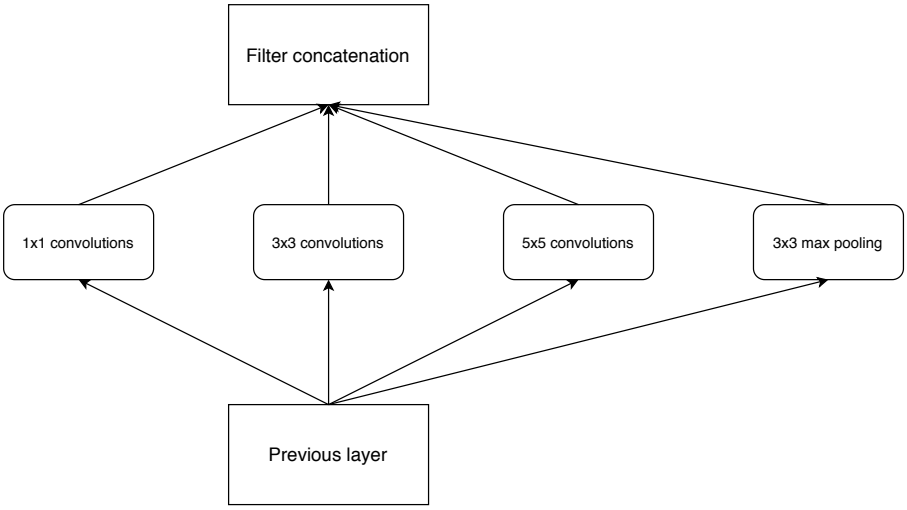


Figure 3.16 – A simple Inception Module

For typical CNN layer, we choose whether we put a stack of 3x3 filters, or a stack of 5x5 filters or a pooling layer. Commonly, they all are useful to the network, so the inception module decide

to use all of them. It means that the module adds all 1x1, 3x3, 5x5 filters and execute convolution on the output from the previous layers rather than picking a particular filter size layer. Because pooling layer is the idea key behind the accomplishment of CNNs, the inception module still leave a place for pooling path.

The computations in the inception modules are performed in parallel. The output of each inception module is very large because it is concatenated, so a 1x1 filter is used to reduce the data dimension. When a 1x1 filter is added, the inception module becomes the following structure:

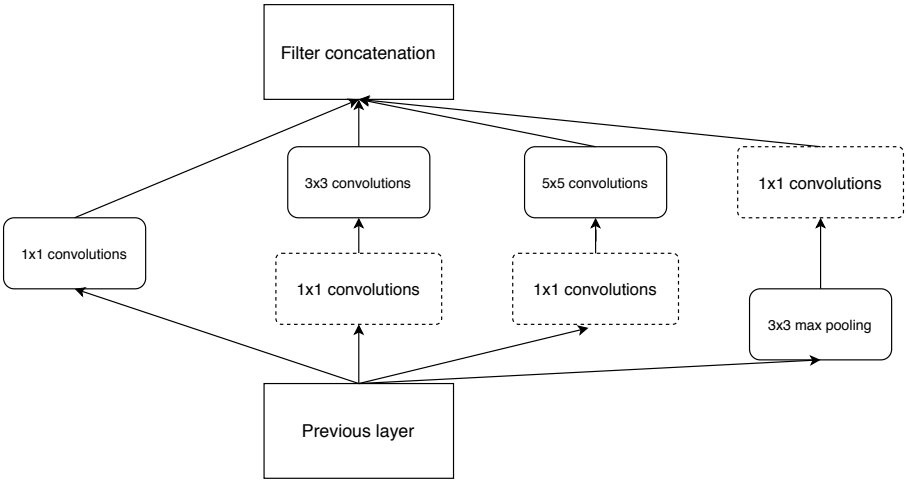


Figure 3.17 – Inception Module with dimension reductions. [5]

After combining all Inception layer, we obtain the following general model:

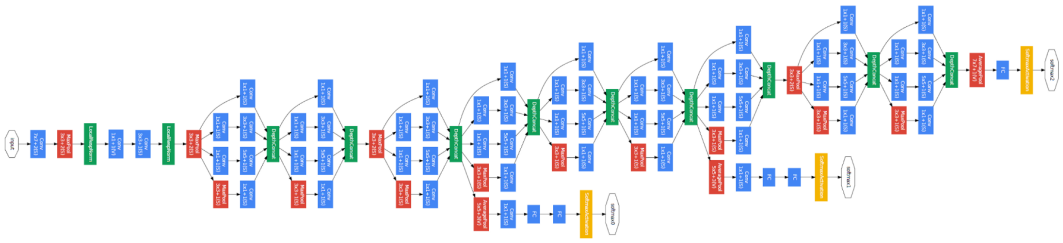


Figure 3.18 – InceptionV3 Architecture.

The architecture in Fig.(3.18) uses 9 inception modules with a total of about 100 hidden layers.

3.5.2 Xception

Chollet, 2016 built a Convolutional Neural Network architecture which depthwise separable convolution layers are used mostly, called *Xception*. The name is short for "Extreme Inception", since its hypothesis is a stronger version than the one of Inception: "the mapping of cross channels correlations and spatial correlations in the feature maps of convolutional neural networks can be entirely decoupled"[4].

The network has 71 layers deep with only 22.9 million parameters and an image input size of 299-by-299, and 36 convolutional layers forming the feature extraction base of the network. These layers are structured into 14 modules, all of them have linear residual connections around, except for the first and last modules. Basically, this architecture is a linear stack of depthwise separable convolution layers with residual connections which make it easy to define and modify

by using a powerful and high-level library like Keras. An open-source implementation of Xception using Keras and TensorFlow is provided as part of the Keras Applications module, under the MIT license. Xception is also one of the latest and most accurate models until now.

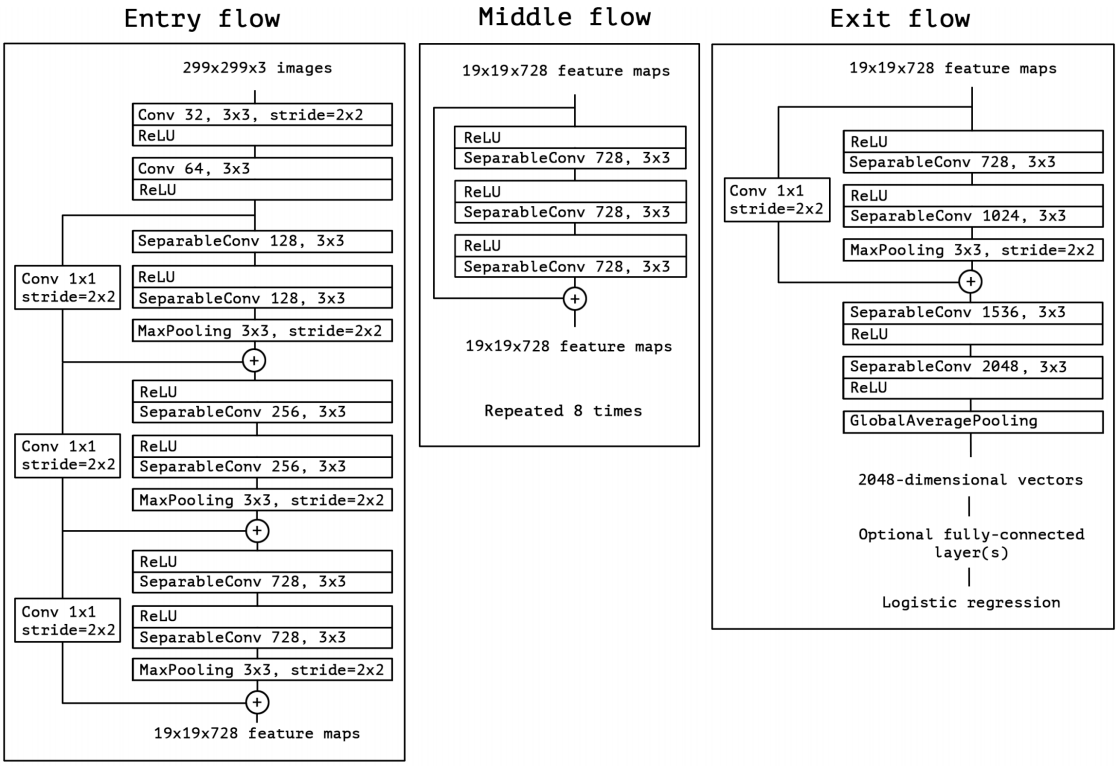


Figure 3.19 – The Xception architecture[4].

The figure 3.19 illustrates the entire architecture of Xception and describe how the data works with it. First, it goes through the entry flow to the middle flow which will be repeated 8 times, and finally through the exit flow. A small reminder is that, all Convolution and Separable Convolution layers are followed by a batch normalization that is not included in the figure above, also these Separable Convolution layers apply a depth multiplier of 1 (no depth expansion).

3.6 Metrics

The *Primary Metric Value* that the Challenge wants to aim is *Balanced Multiclass Accuracy*, we will refer as BMA. According to the ISIC's definition, It is the greatest diagnosis category score determines the category prediction for each image; the mean recall of this multiclass confusion matrix (i.e. the mean of the diagonal element-wise divided by the positive incidences). Participants who get the best one will be ranked and awards granted based on it. To summarize, BMA is the same as the average recall of all skin disease classes which is 9 including the unknown class as well. The formula is explained as below:

$$BMA = \frac{1}{9}\sum_{i=1}^{9}\frac{TP_i}{TP_i + FN_i} \tag{3.5}$$

with weighted by the categoey prevalence, which TP_i and FN_i represent the true positive cases and false negative cases for i^{th} class.

Nevertheless, it is not the only metric they will work on it, to conclude the scientific research, regular measurements are going to be utilized to predict responses (comparing prediction vs. ground truth) for each of the image:

- One of the basic evaluation criteria to check the model is using confusion matrix. In the process of classification, data is separated into 2 typical classes which are Positives (P) and Negatives (N) regarding to has No-nodule. After the operation, the classifier predicts all data values whether they are positive or negative. The outcome produces four unique types of results below:
  - True positive (TP): refers the total of predictions where the classifier predicts the positive category as positive
  - True negative (TN): refers the total of predictions where the classifier predicts the negative category as negative
  - False positive (FP): refers the total of predictions where the classifier predicts the negative category as positive (Type I error).
  - False negative (FN): refers the total of predictions where the classifier predicts the positive category as negative (Type II error).

  True / False indicates whether what we predicted is true or not (true or false). Positive / Negative means what we predict (yes or no).

  Based on these 4 outcomes, performance of the classification model can be evaluated using their 2×2 formulated table called binary confusion matrix as in table 3.1.

Table 3.1 – Confusion matrix of the classifier

		True class	
		Positive	Negative
Predicted class	Positive	TP	FP
	Negative	FN	TN

Nevertheless, our case is about multi-class problem so we cannot apply the same theory of binary classification. In multi-class case, there will be NO positive or negative classes here which will make it hard to find TP, TN, FP, FN but the solution is quite simple, we just need to find the 4 categories (TP, TN, FP, FN) for each single class. An example is illustrated below, suppose we are having a dataset contains 3 groups namely Dog, Cat, Mouse with the following confusion matrix (Table 3.2) and we want to find information relate to class Mouse:

Table 3.2 – Confusion matrix for Multi-Class Classification

		True class		
		Dog	Cat	Mouse
Predicted class	Dog	6	9	7
	Cat	3	5	6
	Mouse	1	4	2

We will have:

  - TP = 2
  - TN = 6+9+3+5 = 23
  - FP = 4+1 = 5
  - FN = 6+7 = 13

- *Sensitivity (Recall)* measures the proportion of actual positives that are truly positive:

$$Sensitivity = \frac{\sum TP}{\sum TP + \sum FN} \tag{3.6}$$

- *Specificity* calculates the percentage of actual negatives that are correctly negative:

$$Specificity = \frac{\sum TN}{\sum TN + \sum FP} \tag{3.7}$$

- *Accuracy* is the number of correct predictions made as a ratio of all predictions made:

$$Accuracy = \frac{\sum TN + \sum TP}{\sum TN + TP + FN + FP} \tag{3.8}$$

- *F1-score* is the harmonic mean of the precision and recall and it is a measure of a test's accuracy with the following formula:

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{3.9}$$

  F1-score is valid between half intervals (0,1]. The higher of F1-score, the better the classifier. When both recall and precision are equal to 1 (best possible scenario), F1 = 1.

- *Positive Predictive Value (PPV) (Precision)*
  With a way of identifying a positive class, *Precision* is defined as the ratio of true positive points among those classified as positive (True Positive + False Positive).

$$PPV = \frac{\sum TP}{\sum TP + \sum FP} \tag{3.10}$$

Chapter 4

Evaluation

4.1 Experiment Setup

Data. To tackle the imbalance problem, during the pre-processing, we pick the number of images per class for all the categories to a random number which is 20000 as the number that each class will have to balance the dataset, for all the classes that having fewer number of samples, we generate by performing various augmentation as mentioned in *Section 2.4*. Finally, we split data as: Training set accounts for 80% of the total dataset, validation set accounts for 20%

Framework. We choose Keras as the deep learning framework. Keras contains a group of different pre-trained models on the ImageNet dataset that helps us build the model we want quickly due to its fast convergence and also, it has an ability to run on top of various deep learning libraries like TensorFlow or Theano running as backend. We did try to experiment some of Keras models like VGG, ResNet, InceptionV3, Xception, but the first two models, they required hardware resources, high-configuration hardware and large memory.

Model. So in the end, we pick InceptionV3 and Xception as our pretrained model for training. Also Xception was the upgraded version of InceptionV3. As for the parameter to setup for the 2 models, we adjust as following:

InceptionV3

- Set of batch size for training and validating is 30, the iterator for each epoch equals to number of training samples divide to number of training/validating batch size.
- Remove the last 2 layers of the model
- Pick Dropout = 0.25 to prevent overfitting.
- Retrain the whole model
- Use the Adam optimizer with a start learning rate of 0.001.
- Choose Softmax as the activation function in the prediction layer.
- Loss function: "Multi-class" Categorical Cross-entropy with metrics is categorical_accuracy and top 3 accuracy.
- Use ModelCheckpoint callback to save the weight with the best accuracy of the model.
- Reduce learning rate when the metric has stopped improving: decrease by a factor of 0.05 after 2 epochs with ReduceLROnPlateau.

- Call Early Stopping to stop the training if no improvement was detected with patience=10.
- Perform 20 epochs.

Xception

- Set of batch size for training and validating is 20, the iterator for each epoch equals to number of training samples divide to number of training/validating batch size.
- Remove the last 2 layers of the model
- Pick Dropout = 0.5 to prevent overfitting.
- Fine-tuning the model by choosing the last 34 layers of the model to be retrained instead of the whole model. Because We need to choose how many layers we actually want to be trained. Here we are freezing the weights of all layers except the last 34 layers in the new model. The last 34 layers of the model will be trained. We do not want to change the weight of ImageNet that Xception had trained before, we only want to readapt the pretrained weights in an incremental way with different dataset.
- Use the Adam optimizer with a start learning rate of 0.01.
- Choose Softmax as the activation function in the prediction layer.
- Loss function: "Multi-class" Categorical Cross-entropy with metrics is categorical_accuracy and top 3 accuracy.
- Use ModelCheckpoint callback to save the weight with the best accuracy of the model.
- Reduce learning rate when the metric has stopped improving: decrease by a factor of 0.5 after 2 epochs with ReduceLROnPlateau.
- Call Early Stopping to stop the training if no improvement was detected with patience=20.
- Perform 30 epochs.

4.2 Result

The fig. 4.1 below illustrates the learning performance of InceptionV3 model throughout experience or time. The training-validation curves is represented to help us diagnose learning problems such as underfitting, overfitting or well-fit model, also to figure out if the training and validation dataset are representative or not.

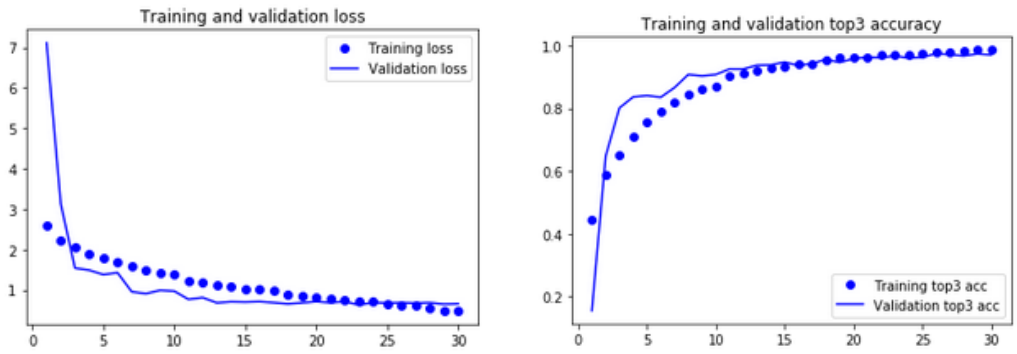
(a) Training and Validation Loss

(b) Training and Validation Top 3 Accuracy

Figure 4.1 – The Learning Curves of InceptionV3.

Looking at the plotting curve and we have mentioned above, we got stuck with the Overfitting issue as the gap between Training and Validation Loss was too huge (fig. 3.2a). In the initial stage of training, the validation loss is lower compare to the training loss and vice versa, the validation accuracy is higher than the training accuracy, this can be explained in several ways. First, during the fine-tuning of the model, we have utilized the Dropout layer in the architecture so we can avoid the Overfitting, these layers was meant to disable some of the neurons during the training in order to reduce the complexity of the system. But in Keras, the purpose of using Dropout is to disable during testing phase, it will give the network full computational power that can both perform prediction and have a training accuracy better and greater just for a few epochs while evaluating the system. Second reason, as the model is developing from time to time, the training loss at the last batches is usually higher than the starting batches of an epoch. In stark contrast to validation loss, it is computed at the end of an epoch that cause lower loss, which is why the validation loss is smaller compare to training loss.

We also plotting the learning curve of the model (fig. 4.2) to diagnose whether the model is going the right direction or not. Based on these curves, the whole process seem to be stability since the gap between training loss and validation loss is minimal, most likely it is the *generalization gap* (nearly zero in an ideal situation). This plot indicates that the results are reproducible, also the algorithm is robust with high confidence for accurately classifying lesions belong to which categories. And these results prove that there is no overfitting or underfitting on the transfer learning model. However, since this is a multiclass classification, so during the prediction for multiple classes, there is a probability of having an incorrect prediction. In addition, the model performance was decreasing with several classification classes. But in the end, the model's performance works pretty well.



(a) Training and Validation Loss

(b) Training and Validation Top 3 Accuracy

Figure 4.2 – The Learning Curves of Xception.

4.2.1 InceptionV3

We have done experimentation completely for the InceptionV3 model by testing the effectiveness of the model over the testing dataset which contains 8,238 images, its outcome were not as we expected due to the fact that we had the Overfitting problem, the reason lies behind the imbalance between every class. The model *Accuracy* outcome is 93% with the Primary Metric Value (*Balanced Multiclass Accuracy*) of 0.41, furthermore the model achieves *Sensitivity*, *Specificity*, *Precision* of 41%, 94%, 47%, respectively. Looking at these results, the Sensitivity and Precision are way too low, if this was acquired from CAD medical system then it sure be not accepted since it determines the people life. The reason lies behind the huge gap in the number of samples between each class. The evaluation result is shown as following:

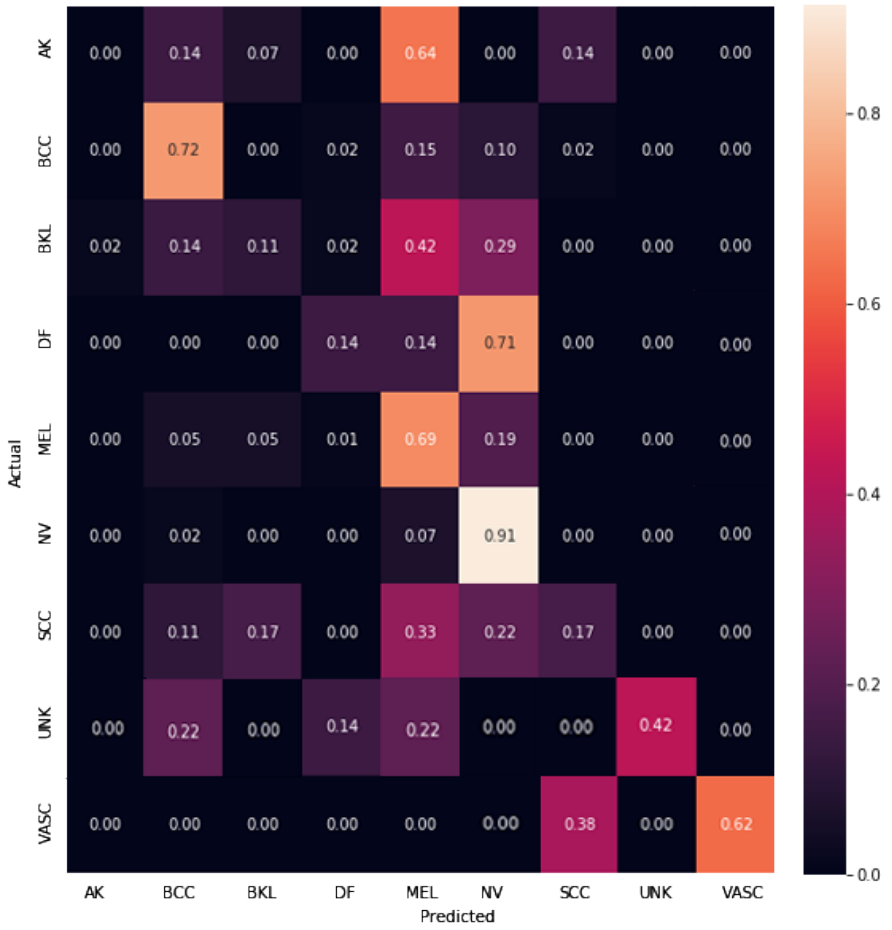


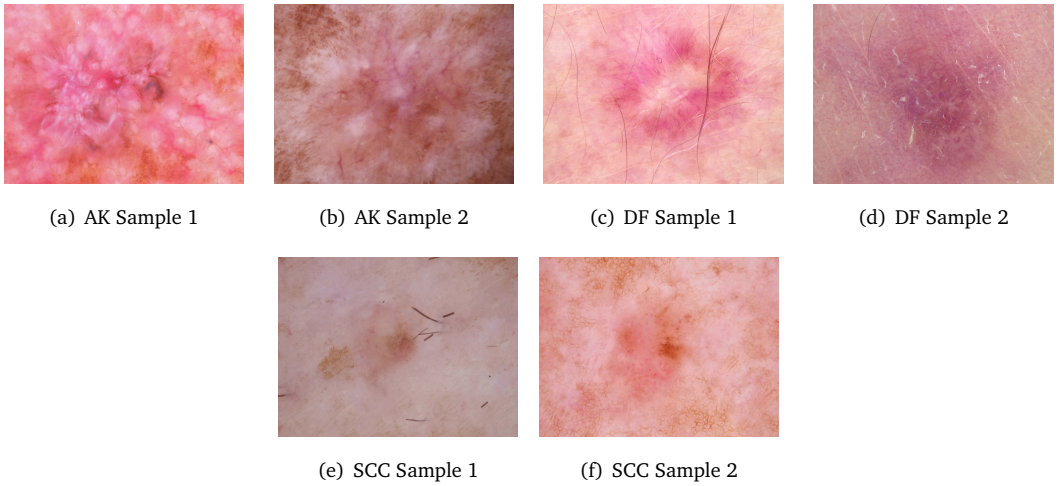
Figure 4.3 – Confusion matrix of Inception

Table 4.1 – Classification Report of experiment with InceptionV3 Model

	Precision	Recall	F1-score
AK	0.00	0.00	0.00
BCC	0.54	0.72	0.62
BKL	0.55	0.11	0.19
DF	0.17	0.14	0.15
MEL	0.31	0.69	0.43
NV	0.90	0.91	0.90
SCC	0.50	0.17	0.25
VASC	0.83	0.62	0.71
UNK	0.49	0.14	0.21

Based on these consequences, even though some of the metrics are quite significant with *Accuracy* and *Specificity* are up to 93-94%, the rest are trivial. Firstly, the model could not predict much of images belong to the AK type so all 3 measurements including Precision, Recall, F1-score equal to 0. Apparently, the model was predicting all the AK images belong to the MEL class. This also happened for other classes as well. Right above the AK class are the classes of DF, BKL and SCC with the results that are not up to the standard, F1-score equal to 15%,19%, 25% respectively. The images of these classes were predicted wrongly, as such, BKL was in the same case with AK, while DF ones were predicted as NV class. Nevertheless, SCC class is classified scatteredly

among other categories, not evenly uniform. We made a conclusion this happened because these categories have a limited number of samples in the dataset and in addition their different shape and color are quite difficult to classify, such as color of the skin, the unrecognizable lesions, etc. (fig. 4.4). On the other hand, for the other classes, their measurement metrics indicate the model is still good and consistent as they are varying between 70% - 90%.



(a) AK Sample 1 (b) AK Sample 2 (c) DF Sample 1 (d) DF Sample 2

(e) SCC Sample 1 (f) SCC Sample 2

Figure 4.4 – Some difficult examples to classify.

4.2.2 Xception

After testing the model on 8,238 images, the work achieves an accuracy of 95%, also obtain *Balanced Multiclass Accuracy* of 0.70, the model has 70% of Sensitivity and 96% of Specificity. When compare with the InceptionV3 model we can see the result of Xception is much better, the other metrics is also much higher but still not really quite well since the dataset is still vastly imbalanced. The metrics used to evaluate the model can be described as below:

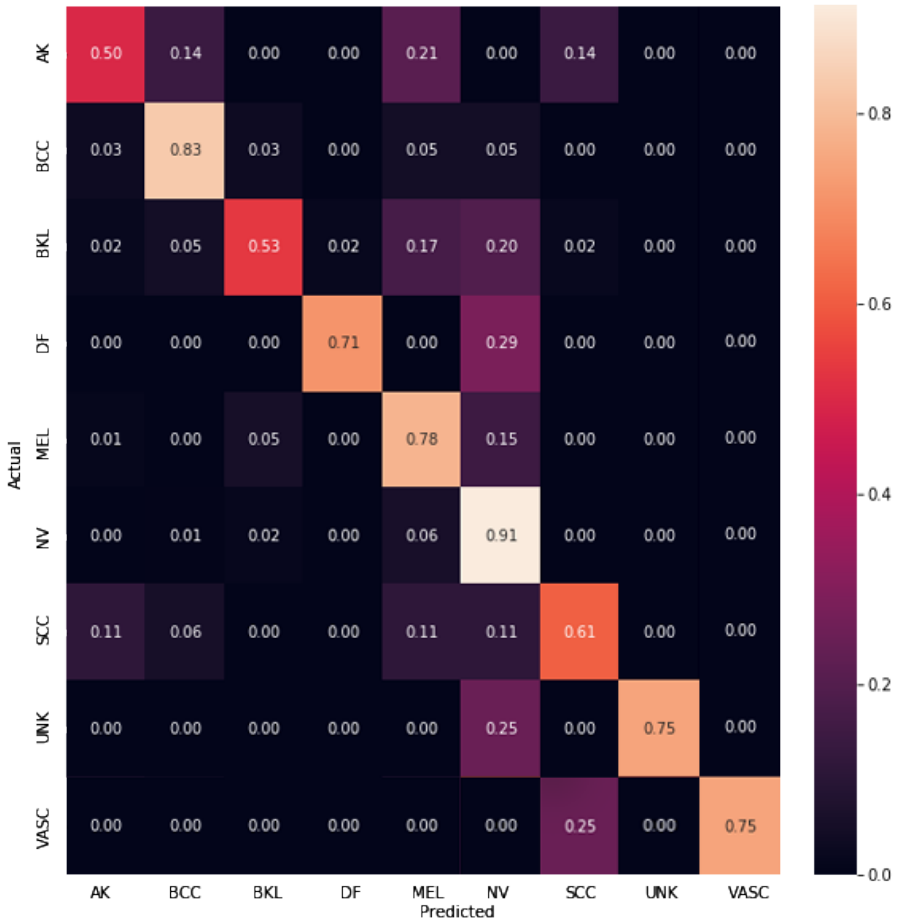


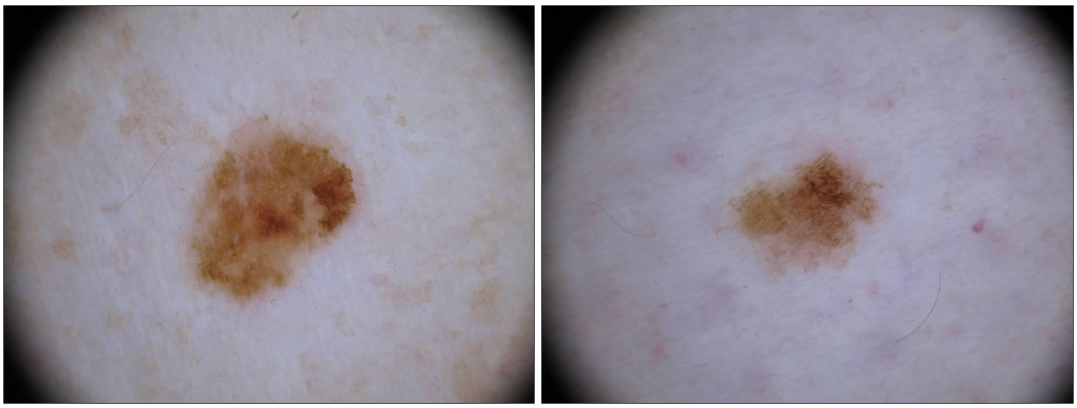
Figure 4.5 – Confusion matrix of Xception

Table 4.2 – Classification Report of experiment with Xception Model

	Precision	Recall	F1-score
AK	0.47	0.57	0.52
BCC	0.80	0.78	0.79
BKL	0.72	0.54	0.62
DF	0.40	0.29	0.33
MEL	0.45	0.81	0.58
NV	0.93	0.89	0.91
SCC	0.62	0.44	0.52
VASC	0.75	0.75	0.75
UNK	0.75	0.75	0.75

Table 4.2 presents the performance result for the skin cancer classification algorithm. Clearly the result has a big difference between every categories since the dataset is vastly imbalance, the best outcome comes to the NV class 91% of F1-score, and the lowest one is DF with only 33% right after AK and SCC classes, this happens since the shape of these classes is difficult to discern (blur, irregular shapes of skin lesions, different types of colors on each skin,). As we can see in the classification report, the recall of MEL was incredibly higher than precision, at first it was lower and equal to precision, the reason was due to the similarity between MEL and NV class (fig. 4.6); at that time more than 25% false negatives belong to NV group. To solve this issue, we decided to

weigh the classes, punish harder on the false negatives for the MEL category, the consequence was the recall was increasing, but also decreased the precision, this should be reasonable since having false positive for a dangerous disease is better than having a false negatives. But in the end, over 8 out 9 classes the model predicted, they exceed over 50% of F1-score, this result was quite accepted. Compare to the previous experiment with InceptionV3 model by looking at the Classification report, the average values for all columns were higher. After many trials for fine tuning and the optimum outcome, this were the best results so far.



(a) MEL Sample (b) NV Sample

Figure 4.6 – The similarity between MEL and NV class.

4.3 Discussion

A comparative study is a must if we want to evaluate the performance of our works even further. It was carried out with other existing previous research to compare since they should use the same image dataset of skin lesions, same training and testing set.

Table 4.3 – Comparative Study

Ref.	Method	Accuracy, %	Precision(PPV), %	Sensitivity (Recall), %	Specificity, %	BMA
[14]	EfficientNets	92.47	57.13	45.26	98.31	0.636
[20]	GoogleNet	94.92	80.36	79.8	97	N/A
[15]	Decision Fusion [1]	99.5	N/A	98.3	99.6	N/A
Our	**InceptionV3**	**93.63**	**47.52**	**41.97**	**94.75**	**0.419**
	Xception	**95.96**	**71.98**	**70.50**	**96.64**	**0.705**

[1] Decision fusion method: Through transfer learning, based on multiple pre-trained CNNs, they use the block to combine multiple CNNs and finally make decisions through multiple blocks. The method of decision fusion can solve the generalization capability of an individual CNN model, it is more robust and stable than the traditional fusion strategy.

First, as listed in the ISIC 2019 leaderboard, Gessert et al. was placed 1st in the Challenge in both tasks for achieving the highest performance with BMA of 0.636, his team did try to ensemble a multi-resolution EfficientNets with loss balancing and obtained an accuracy of 93%. In our 1st work, InceptionV3, our results are still far lower than them. However, the Xception model has made great progress and some of the key metrics have surpassed them, including Accuracy, Precision, Sensitivity and even BMA.

Come to the 2nd result, Khalid M. Hosny proposed the method of using transfer learning and pre-trained deep neural network GoogleNet, even with the imbalance dataset, they still acquired significant outcomes with accuracy, sensitivity, specificity, and precision of 94.92%, 79.8%, 97%, and 80.36%, respectively. Hosny and his team also suggested other ideas of solving the imbalance in images between classes problem, including decreased the number of samples, fine-tuned the weights of all architecture layers instead of fine-tuned only the replaced layers, the performance was actually increased.

Another idea comes from GONG et al., their ways of dealing the imbalance matters were quite effective by training a model based on the ISIC 2019 dataset with StyleGANs. In short, StyleGAN was originally an open-source project by NVIDIA, it creates a generative model that could output high-quality images at each convolution layer and select the realistic images to add to the original dataset. By using these upgraded dataset, they decided to train 43 pre-trained CNN models and divided 3 groups of CNNs for comparison to come up the fusion strategies, which CNNs with higher classification results will be an advantage for the effect of fusion. This method can be considered an extraordinary development since it does not only classify dermoscopy images, but also enrich the dataset and lessen the uneven distribution of dermoscopy images. With this, they gained an accuracy of 99.5% and the Sensitivity, Specificity were 98.3%, 99.6%, respectively.

In our case, we choose the InceptionV3 and Xception model to challenge the classification of skin cancer. For the parameters and hyperparameters of 2 models, mostly all of them are the same except we fine-tuned the weights of all layers and the result was not really high or should be say unacceptable. So we tried the reverse way, opposite of what Hosny said, we fine-tuned the some of the last layers of Xception and the result was improved remarkably. The outcomes was mostly 2 times better than our proposed Inception model as shown in Table 3.3. Compare to the top 3 studies above, we still lack of experiences so the results may not be equal to them but can still improve in the future.

The limit of the Challenge is mentioned above, the goal when apply generally machine learning algorithms is to find a model that generalizes well, so it can make good predictions even on unseen data. Having a balanced and large dataset is crucial for a better result. However, the dataset used in this Challenge had a huge gap between every classes, although we did many way to augment data to improve the performance, the result was not still very high. The trouble also increases when the tasks require our model need to classify an out-of-distribution class so that it can deal with real life case.

Chapter 5

Conclusion & Future work

5.1 Conclusion

In general, as the frequency paces of skin cancer have been raising over the previous many years, there is a pressing need to address this worldwide general medical problem. Skin cancer will have a better diagnosis if it is found and treated earlier. Computer Aided Diagnosis is a powerful system to assist doctors in diagnosing cancer. It also raises people's health awareness to increase survival rate. But since the diagnosis is done by human's works so we cannot always sure whether mistakes can happen or not.

In this report, we have introduced about the cancer, generally speaking, and skin cancer in particular, the statistics show how dangerous of skin cancer. Within 6 months of internship, we have studied the medical field which is about skin cancer and have taken on the ISIC 2019 Challenge of building an architecture for skin cancer classification purpose. Also we have addressed how to create a system that is capable of classifying the skin cancer types with high accuracy by applying *Transfer Learning* and *Fine-tuning* techniques, typically with 2 pre-trained models InceptionV3 and Xception. The models were built in Python by using the Keras package with TensorFlow as back-end. The input data was provided from ISIC Archive, an international repository of dermoscopic images, which was developed by International Skin Imaging Collaboration (ISIC), to train and classify. The dermoscopy image is a huge challenge for classification task due to its complexity feature and lacks of sample, so many of pre-processing techniques have been carried out to solve this drawback.

After the process of training and testing with 2 models, Xception outperform InceptionV3 with accuracy of 95%, sensitivity of 70% and specificity of 96% , the performance of the model is analyzed by using several evaluation metrics. Based on the outcomes we acquire and the comparison with the previous researches, its can be relatively accepted, but it can still go further if we have more data with more variance in the examples.

5.2 Future work

Many adaptations, tests, and experiments have been left for the future due to lack of time. We would like to test with some other pre-trained models to find the best architecture that qualifies for these tasks. In the future, when the amount of data is sufficient by collecting from other sources, the result will become better and we will be able to build the application that can save people from cancer.

Bibliography

[1] https://challenge2018.isic-archive.com/. 2018.

[2] Bo Chen Dmitry Kalenichenko Weijun Wang Tobias Weyand Marco Andreetto Hartwig Adam Andrew G. Howard Menglong Zhu. "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications." In: (2017). DOI: arXiv:1704.04861v1.

[3] DENNY BRITZ. *Understanding Convolutional Neural Networks for NLP*. 2015.

[4] Francois Chollet. "Xception: Deep Learning with Depthwise Separable Convolutions." In: (2016).

[5] Sergey Ioffe Jonathon Shlens Zbigniew Wojna Christian Szegedy Vincent Vanhoucke. "Rethinking the Inception Architecture for Computer Vision." In: (2015).

[6] Noel Codella et al. "Skin lesion analysis toward melanoma detection 2018: A challenge hosted by the international skin imaging collaboration (isic)." In: *arXiv preprint arXiv:1902.03368* (2019).

[7] Noel CF Codella et al. "Skin lesion analysis toward melanoma detection: A challenge at the 2017 international symposium on biomedical imaging (isbi), hosted by the international skin imaging collaboration (isic)." In: *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*. IEEE. 2018, pp. 168–172.

[8] Q-B. Nguyen Sharath Pankanti D. A. Gutman Brian Helba A. C. Halpern Codella Noel CF and John R. Smith. "Deep learning ensembles for melanoma recognition in dermoscopy images." In: *IBM Journal of Research and Development 61, no. 4/5* (2017).

[9] Emre Celebi Brian Helba Michael Marchetti Nabin Mishra Allan Halpern David Gutman Noel C. F. Codella. "Skin Lesion Analysis toward Melanoma Detection: A Challenge at the International Symposium on Biomedical Imaging (ISBI) 2016, hosted by the International Skin Imaging Collaboration (ISIC)." In: (2016). DOI: arXiv:1605.01397.

[10] Arden Dertat. *Applied Deep Learning - Part 4: Convolutional Neural Networks*. 2017.

[11] Muhammad Ali Farooq and Viktor Varkarakis Peter Corcoran others Asma Khatoon. "Advanced Deep Learning Methodologies for Skin Cancer Classification in Prodromal Stages." In: (2020).

[12] SKIN CANCER FOUNDATION. *Skin Cancer 101, Knowledge is Your Best Defense*. https://www.skincancer.org/skin-cancer-information/. 2020.

[13] SKIN CANCER FOUNDATION. *Skin Cancer Facts Statistics, What You Need to Know*. https://www.skincancer.org/skin-cancer-information/skin-cancer-facts/. 2020.

[14] Nils Gessert and Mohsin Shaikh Rene Werner Alexander Schlaefer others Maximilian Nielsen. "Skin Lesion Classification Using Loss Balancing, Ensembles of Multi-Resolution EfficientNets and Meta Data." In: (2019).

[15] AN GONG and WEI LIN others XINJIE YAO. "Dermoscopy Image Classification Based onStyleGANs and Decision Fusion." In: (2020). DOI: 10.1109/ACCESS.2020.2986916.

[16] Stephen Hayes, ed. *Dermoscopy: an update and personal view*. 2018.

[17] NATIONAL CANCER INSTITUTE. *Understanding Cancer*. https://www.cancer.gov/about-cancer/understanding/statistics. 2020.

[18] Shaoqing Ren Jian Sun Kaiming He Xiangyu Zhang. "Deep Residual Learning for Image Recognition." In: (2015).

[19] Kalouche, Andrew Ng others Simon, and John Duchi. "Vision-based classification of skin cancer using deep learning." In: (2016). conducted on Stanfords Machine Learning course (CS 229) taught (2016).

[20] Mohamed M. Foaud Khalid M. Hosny Mohamed A. Kassem. "Skin Lesions Classification Into Eight Classes for ISIC 2019 Using Deep Convolutional Neural Network and Transfer Learning." In: (2020). DOI: 10.1109/ACCESS.2020.3003890.

[21] Pehamberger H. Wolff K. Binder M. Kittler H. "Diagnostic accuracy of dermoscopy." In: ().

[22] Yuexiang Li and Linlin Shen. "Skin lesion analysis towards melanoma detection using deep learning network." In: (2018). DOI: 10.3390/s18020556.

[23] Marc Combalia, Noel C. F. Codella, Veronica Rotemberg, Brian Helba, Veronica Vilaplana, Ofer Reiter, Allan C. Halpern, Susana Puig, Josep Malvehy. "BCN20000: Dermoscopic Lesions in the Wild." In: (2019). DOI: arXiv:1908.02288.

[24] Mahmudul Hasan Tarek M. Taha Vijayan K. Asari Md Zahangir Alom Chris Yakopcic. "Recurrent residual U-Net for medical image segmentation." In: (2018). DOI: 10.1117/1.JMI.6.1.014006.

[25] Alex Krizhevsky Ilya Sutskever Ruslan Salakhutdinov Nitish Srivastava Geoffrey Hinton. "Dropout: a simple way to prevent neural networks from overfitting." In: *JMLR 2014* (2014).

[26] Thomas Brox Olaf Ronneberger Philipp Fischer. "U-Net: Convolutional Networks for Biomedical Image Segmentation." In: (2015). DOI: arXiv:1505.04597.

[27] Hao Su Jonathan Krause Sanjeev Satheesh Sean Ma Zhiheng Huang Andrej Karpathy Aditya Khosla Michael Bernstein Alexander C. Berg & Li Fei-Fei Olga Russakovsky Jia Deng. "ImageNet Large Scale Visual Recognition Challenge." In: *International Journal of Computer Vision volume* (2015).

[28] Adleff V Leal A Hruban C White J Anagnostou V Fiksel J Cristiano S Papp E Speir S Reinert T Orntoft MW-Woodward BD Murphy D Parpart-Li S Riley D Nesselbush M Sengamalay N Georgiadis A Li QK Madsen MR Mortensen FV Huiskens J Punt C van Grieken N Fijneman R Meijer G Husain H Scharpf RB Diaz LA Jr Jones S Angiuoli S Ørntoft T Nielsen HJ Andersen CL Velculescu VE. Phallen J Sausen M. "Direct detection of early-stage cancers using circulating tumor DNA." In: (2016). DOI: 10.1126/scitranslmed.aan2415..

[29] Tschandl P., Rosendahl C. Kittler H. "The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions." In: 180161 (2018). DOI: doi.10.1038/sdata.2018.161.

[30] Roberto Cipolla Vijay Badrinarayanan Alex Kendall. "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation." In: (2016).

[31] Juntang Zhuang. “LadderNet: Multi-path networks based on UNet for medical image segmentation.” In: (2018). DOI: arXiv:1810.07810.