MELANOMA DETECTION USING DEEP LEARNING

A Non-Thesis Project
presented in partial fulfillment of requirements
for the degree of Master of Science
in the Department of Computer and Information Science
The University of Mississippi

by

Deependra Phuyal

December 2020

Copyright Deependra Phuyal 2020
ALL RIGHTS RESERVED

ABSTRACT

Convolutional Neural Networks (CNNs) have had great success in image classification over the years. The purpose of this report is to discuss the solution to SIIM-ISIC Melanoma Classification Challenge. Two CNNs (EfficientNet-B2, B3, B4 and B6, and ResNet-50 model) were fine-tuned with images and patient-level contextual data (metadata) provided by SIIM-ISIC. The best model scored 0.8688 on private leaderboard and 0.8819 on public leaderboard. My findings suggests: 1) with a good validation strategy, a very good performing classification model can be build and 2) EfficientNet models outperformed ResNet-50 model despite being trained on lesser number of parameters.

ACKNOWLEDGMENTS

I would like to thank Dr. Ana Pavel for constant help and guidance throughout the project and especially for providing powerful GPUs to train my models. Without her generosity this project would not be possible. I would also like to thank my project advisor Dr. Yixin Chen for his help and constructive suggestions. I would like to thank my friends and family for their unwavering love and support. Lastly, I want to thank Sweta Adhikari for her support and encouragement.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Deep learning is a branch of machine learning inspired by the structure and function of human brain called artificial neural networks or simply neural networks. It has dramatically improved the state-of-the-art in many tasks like object detection, speech recognition, drug design and delivery, medical image analysis, machine translation, and many more [7]. Although deep learning was first theorized in the 1980s, it has only found its success in the last decade due to increase in the large amount of labeled data and computational power. Neural networks have three components: an input later, a hidden layer, and an output layer. A deep neural network gets its name from the fact that it is made out of many regular neural networks joined together. Unlike conventional machine learning techniques that were limited in their ability to process data in their raw form, deep learning models can learn to perform classification task directly from images, text, or sound without the need for manual feature extraction [7].

One of the most popular types of deep neural networks architecture is called convolutional neural networks. CNNs excels at object recognition in image data. CNNs essentially takes images and apply filters to extract information and train the model's weights to classify an image. The two CNNs used on this project is described in Chapter 3.

Melanoma is a common skin cancer which is responsible for 75% of skin cancer deaths. The American Cancer Society has predicted over hundred thousand new melanoma cases and

seven thousand deaths from the disease [8]. However, if caught early and accurately, it can be cured with minor surgery. Fast and accurate diagnosis could tremendously benefit doctors and patients. Recent deep learning based computer vision has advanced detection algorithms that has pushed model performance to be close to human expert level in many medical fields. The detection algorithms can be used to diagnose more accurately if contextual images of same patient is used to characterize images with melanoma. This project is a part of SIIM-ISIC Melanoma Classification challenge. Challenge is to build models to identify melanoma using images of skin lesions and patient-level contextual data (metadata) to assist dermatologists.

CHAPTER 2

DATASET OVERVIEW

The dataset for this experiment is selected from Kaggle, an online community of data scientists where datasets are accessible to the users to build models in a web-based data science environment and to solve data science challenges. The official dataset for this classification challenge was generated by the Society for Imaging Informatics in Medicine (SIIM) and International Skin Imaging Collaboration (ISIC). The images in the dataset are from the following sources: Hospital Clínic de Barcelona, Medical University of Vienna, Memorial Sloan Kettering Cancer Center, Melanoma Institute Australia, University of Queensland, and the University of Athens Medical School [1]. The goal of this project is to use images of skin lesions within the same patient and correctly identify benign (non-cancerous) and malignant (cancerous) cases. The patient-level contextual information provided in the metadata files could help build a better classification model to aid clinical dermatologists.

SIIM-ISIC dataset is split into train and test folders along with metadata in the DICOM and CSV format as shown in the figure 2.1. The training set contains 33,126 dermatoscopic images of unique benign and malignant skin lesions from over 2,000 patients and eight features; image_name, patient_id, sex, age_approx, anatom_site_general_challenge, diagnosis, benign_malignant, and target. The test set contains 10,982 medical images and five features; image_name, patient_id, sex, age_approx, and anatom_site_general_challenge. All the images are

provided in three formats; DICOM (Digital Imaging and Communications in Medicine) format, JPEG format, and TFRecord format. For this experiment, JPEG images were taken and resized to (256, 256) to make computation faster.

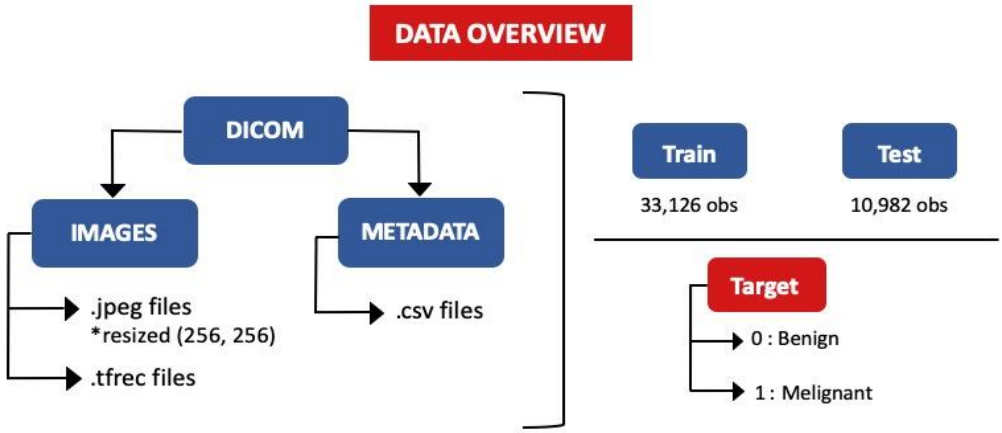

Figure 2. 1 Overview of SIIM-ISIC Dataset

One challenge this dataset present is the unbalanced nature of benign and malignant cases. In figure 2.2, the bar chart displays the number of benign and malignant cases in the training set. Approximately, 98 percent of the images belong to the benign class and malignant class make up for less than 2 percent of the entire dataset. It is difficult to build an accurate classifier with an imbalanced dataset. However, one of the Kagglers, Roman created a dataset with an additional five thousand malignant cases to the original dataset. It increased the percentage of malignant cases to approximately 14 percent, which made the dataset slightly balanced. Apart from an imbalanced dataset, there were few missing values in the train set and test set. The train set has 65 missing values (approx. 0.2% of total data) for sex, 68 missing values for age, and 527 missing values (approx. 1.6% of total data) anatomy features. Similarly, the test set has 351 missing values

(approx. 3% of total data) for anatomy features.

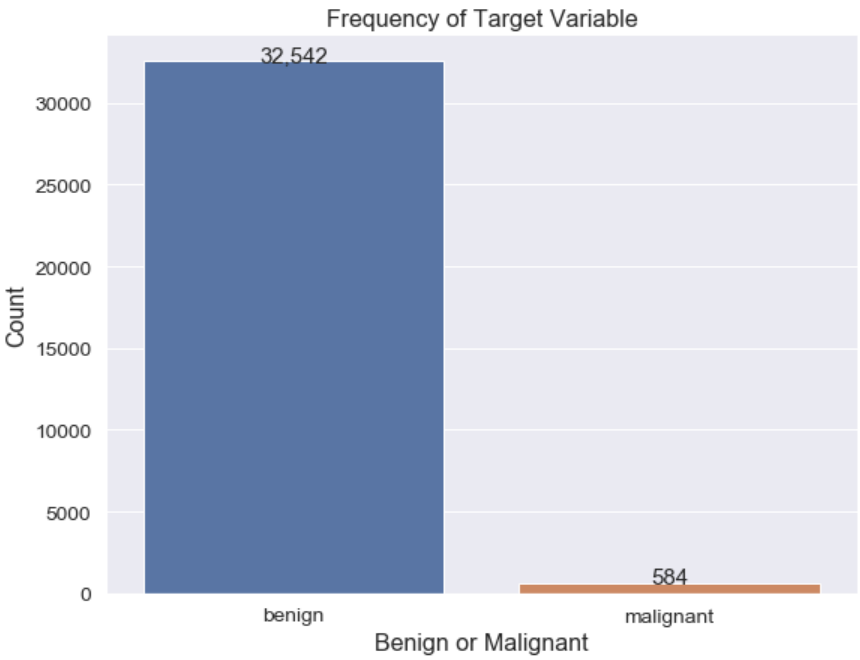

Figure 2. 2 Number of Benign and Malignant Cases

2.1 Data Preparation

The first step done for the data preparation was to impute missing values. As shown in the fig 2.3, the most frequent anatomy value in the train set is torso. Therefore, the *torso* value is imputed for 527 missing values of anatomy feature. Similarly, *male* value is imputed for 65 missing values of sex feature because it had occurrence than female values. Lastly, 68 missing values for age feature is imputed with 0. Likewise, 351 missing values of anatomy feature in the test set is imputed with the *torso* value.

Since neural networks can only operate on vectors of real numbers, the second step was to transform categorical features to numerical values using sklearn LabelEncoder class [2]. LabelEncoder helps normalize features such that they contain values between 0 and n_features-1.

The categorical features encoded were sex, age, and anatomy. After encoding three features, they were normalized using the normalize function of scikit-learn preprocessing package.

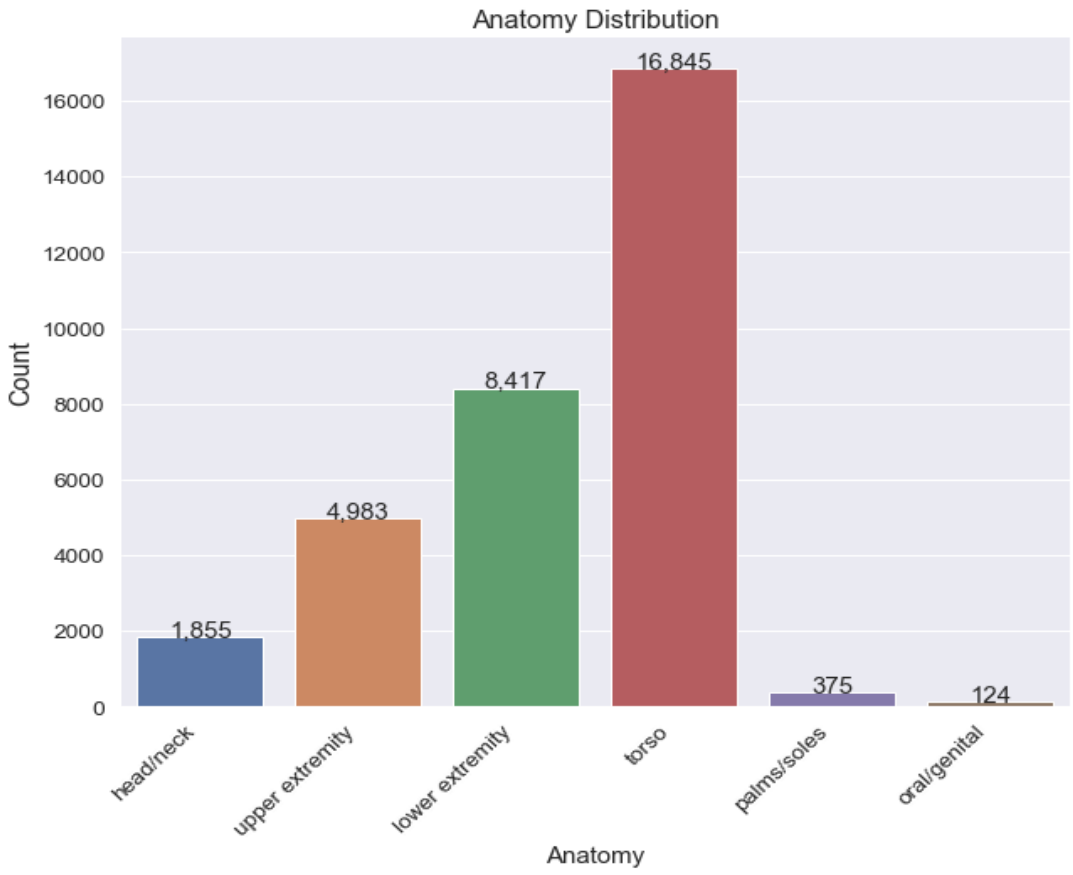

Figure 2. 3 Distribution of Anatomy Feature in Training Set

2.2 Data Augmentation

Data augmentation is one of the techniques used in the field of computer vision to prevent data overfitting in deep learning models and to improve the performance [3]. In this project, Albumentations is used for image augmentations. Albumentations is a fast, popular, and powerful open source Pytorch library for image augmentation with various image transform operations. It

also provides an easier way to apply wrapper around other augmentation libraries. Some of the image augmentations applied for this project are; Transpose, VerticalFlip, HorizontalFlip, Rotate, RandomBrightness, RandomContrast, MotionBlur, MedianBlur, GaussianBlur, GaussNoise OpticalDistortion, GridDistortion, ElasticTransform, CLAHE, HueSaturationValue, ShiftScaleRotate, Cutout resulting on augmented images shown in Fig 2.4. To read more about Albumentations please refer to paper [3]. In addition to Albumentations, Microscope augmentation performed by one of Kagglers is also used because some of the images were taken through microscope which caused black areas around the center of those images. Microscope augmentation helped improve the performance of models discussed in chapter 3.

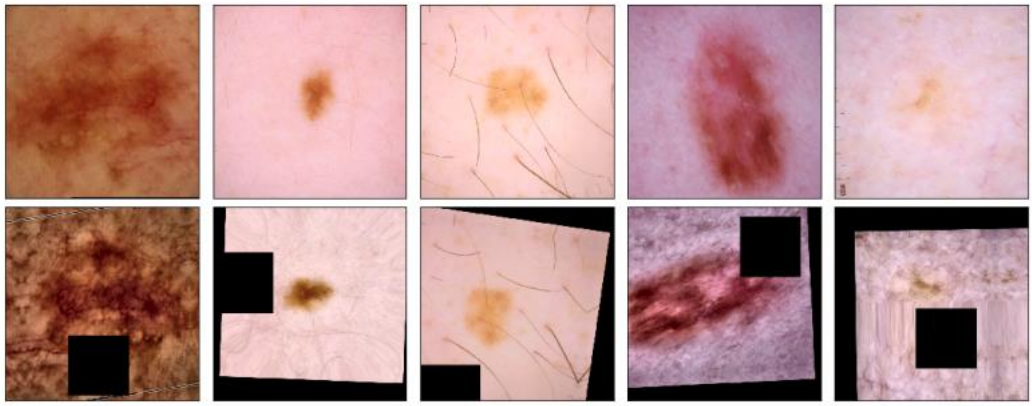

Figure 2. 4 Top row: original images, bottom row: augmented images

CHAPTER 3

MODEL ARCHITECTURE AND EXPERIMENT SETUP

A standard approach for a typical image classification problem is to take a deep CNN model trained on the ImageNet dataset, replace the last layer such that the output dimension equals the target's dimension, and fine tune it on the specific dataset. ImageNet dataset consists of over 15 millions labeled high-resolution images in over 22,000 categories like dogs, cats, keyboard, foods, pencil, etc. [4]. For this project, the pretrained CNN models used are EfficientNet B2, B3, B4, B6, and ResNet-50. A brief description of architecture of EfficientNet models and ResNet-50 is presented below.

3.1 ResNet-50 Architecture

Over the years, there has been a common trend in the research community that a feedforward network architecture needs to go deeper to prevent data overfitting. Moreover, deeper networks can represent more complex features, thus the model robustness and performance can be increased. However, adding more layers to the networks made deep networks hard to train because it made the accuracy value to either saturate or decrease abruptly. The reason for accuracy degradation was vanishing gradient problem. In an attempt to solve this problem, Residual Networks or ResNets introduced a "identity shortcut connection" that skips one or more layers as shown in the figure 3.1. The authors of [5] argue we could simply stack identity mappings (layer

that does not do anything) upon the current network, and the resulting architecture would perform the same. That is the deeper model should not produce a training error higher than its shallower counterparts.

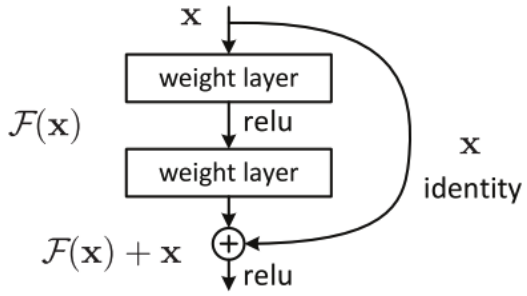

Figure 3. 1 A Residual block

ResNets stack residual blocks (shown in Fig 3.1.) on top of each other to form a network. For instance, a ResNet-50 has fifty layers using residual blocks or is 50 layers deep. ResNet-50 has 48 convolution layers along with one MaxPool layer and one Average Pool layer. The network ends with an average pool layer and a fully connected layer containing 1000 nodes with softmax function (which outputs one layer).

3.1.1 ResNet-50 Architecture Schema

Fig 3.2 illustrates ResNet-50 model schema used in this project. The metadata (age, sex, anatomy) goes through three fully connected layers before being concatenated with the CNN features, which then go to the final fully connected layer to output probability of target variable (benign or malignant case). The last layer of ResNet-50 model which originally contains softmax function is replaced with BCEWithLogitsLoss function. For training schedule, AdamW optimizer is used as oppose to traditional Adam optimizer because it yields better training loss and the models

generalize much better than models trained with Adam. The initial learning rate is set to 0.005

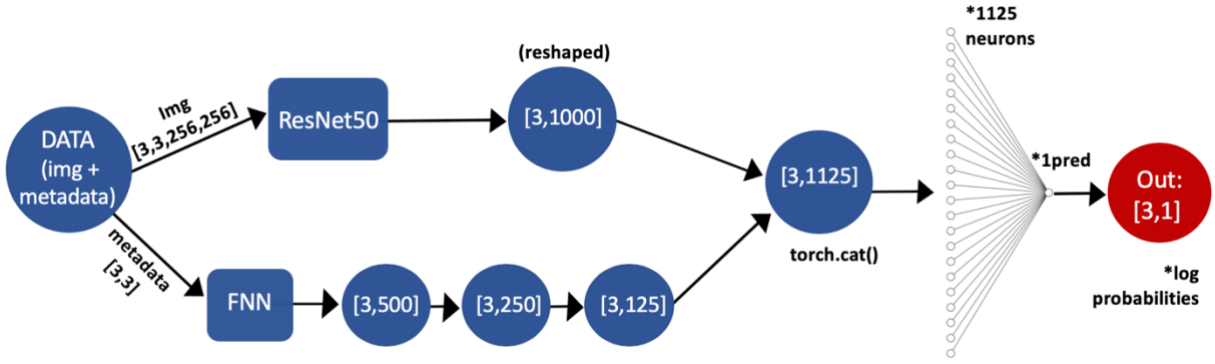

Figure 3. 2 ResNet-50 Schema

with 64 batch size. The total number of epochs used were 24. ReduceLROnPlateau method is used to dynamically reduce the learning rate based on measures like factor and early stopping patience. Early stopping patience is set to three and the factor to 0.2, i.e., if a model does not show improvements after three epochs, it stops and decreases the learning rate by a factor of 0.2 in the next training and validation loop.

3.2 EfficientNets Architecture

Generally, convolutional neural network models are made too deep, wide, or with a very high resolution to improve the accuracy. However, doing so quickly saturates the model making it inefficient. To resolve this, in 2019 two engineers from Google brain team named Mingxing Tan and Quoc V. Le [6] introduced EfficientNet and their strategic way to scale the deep neural networks to achieve higher accuracy. In fact, they made EfficientNet computationally efficient and achieved state of art result on ImageNet dataset, 84.4% top-1 accuracy as shown in fig 3.3. The scaling method introduced in EfficientNet is named compound scaling. It suggests instead of

scaling only one model attribute out of depth, width, and resolution; strategically scaling three attributes together delivers better results.

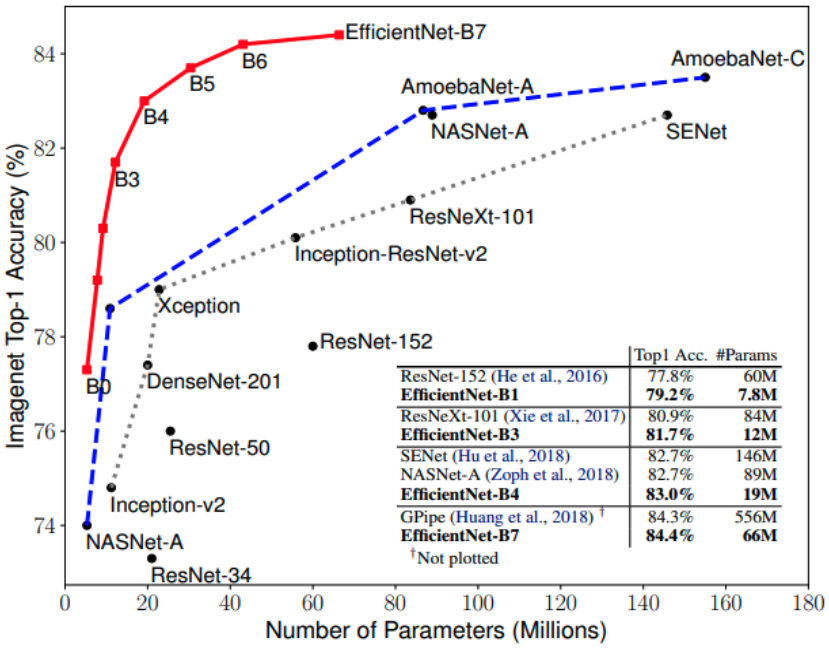

Figure 3. 3 Various Models Size vs ImageNet Accuracy

The base model of EfficientNet family is EfficientNet-B0, developed using a multi-objective neural architecture search that optimizes both accuracy and floating-point operations (FLOPS). Using B0 as a baseline model, a full family of EfficientNets from B1 to B7 were developed. EfficientNet-B0 architecture uses seven inverted residual blocks with different settings for each blocks. These blocks also use squeeze and excitation block along with Swish activation. More details can be found on [6].

3.2.1 EfficientNets Architecture Schema

Fig 3.4 illustrates EfficientNet-B6 model schema used in this project with slight change

(number of neurons on last layer) for EfficientNet B2, B3, B4, and B6 models. The metadata (age, sex, anatomy) goes through two fully connected layers before being concatenated with the CNN features, which then go to the final fully connected layer to output probability of target variable (benign or malignant case). The last layer of all EfficientNet models are replaced with BCEWithLogitsLoss function. Similar to ResNet-50, AdamW optimizer is used for training schedule. The initial learning rate is set to 0.005 with 24 batch size for EfficientNet-B2, 32 for EfficientNet-B3 and B4, and only 16 batch size for EfficientNet-B6 due to limited memory capacity. The total number of epochs used were 24 for EfficientNet-B2 and 15 for EfficientNet-B3, B4, and B6. ReduceLROnPlateau method is used to dynamically reduce the learning rate based on measures like factor and early stopping patience. Early stopping patience is set to three and the factor to 0.2, i.e., if a model does not show improvements after three epochs, it stops and decreases the learning rate by a factor of 0.2 in the next training and validation loop.

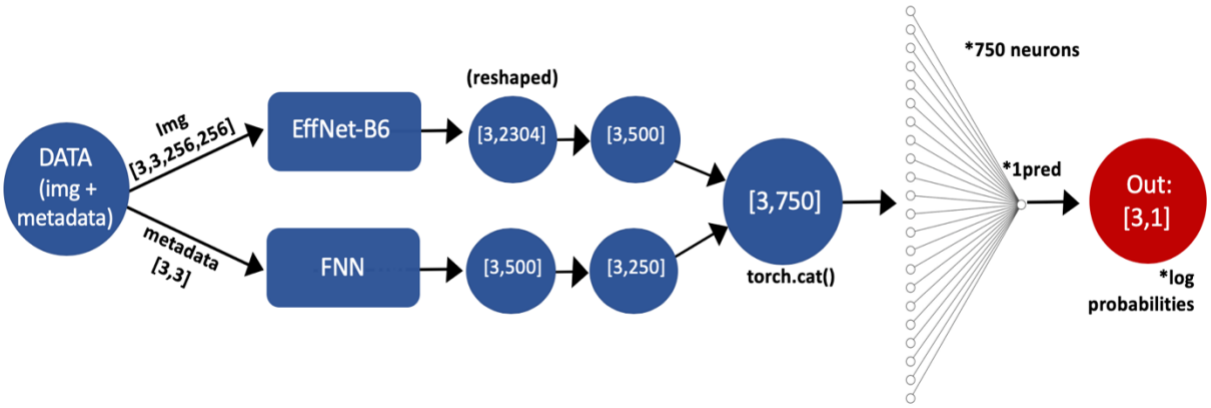

Figure 3. 4 EfficientNet-B6 Schema With Metadata

3.3 Development Setup

The following development setup is used for this project.

- Two AWS g4dn.4xlarge instances with NVIDIA T4 Tensor Core GPU
- Ubuntu 18.04.5 LTS (Bionic Beaver)
- Cuda 10.0.130
- Pytorch 1.6.0
- Python 3.8.5
- Pandas 1.1.2
- Numpy 1.19.1
- Kaggle API 1.5.8
- VS Code 1.51.1

CHAPTER 4

RESULTS

It is crucial to set up a trustworthy validation process in order to properly evaluate and compare models. This is particularly true if the size of the dataset is small to medium, or the evaluation metric is unstable (which is the case of this project). Due to small percentage (1.76%) of positive samples (i.e., malignant) in the original dataset, it caused the Area Under the Curve (AUC) metric to be unstable, even with 5-group fold cross validation. GroupKFold is a K-fold iterator variant with non-overlapping groups. The same group will not appear in two different fold because the number of distinct groups has to be at least equal to the number of folds. Patient id is used for grouping because there are multiple patients with multiple images taken. The solution to this problem is to use both the external dataset and this year's dataset, and perform 5-group fold cross validation on combined data. The external dataset adds nine times more malignant cases than this year's data, which makes the AUC much more stable. However, the above mentioned problem still occurs during model training and validation, and Fig 4.1 perfectly illustrates it.

Since EfficientNet-B6 model achieved the best private leaderboard (LB) score and public leaderboard score, the discussion of training and validation loss as well as training and validation accuracies is about EfficientNet-B6. Fig 4.1 demonstrate EfficientNet-B6 training loss in all five folds, the training loss does not reach the point of stability, i.e., model could not train after a certain point. It is to be expected from a dataset with much smaller positive samples compare to negative

samples.

In Fig 4.2, a line graph of training and validation accuracies of an average (of five folds) fold is plotted. The orange line represents validation accuracy while the blue line represents train accuracy on train and valid sets. For the figure, it can be concluded that higher training and validation accuracies can be achieved from models with pre-trained weights.

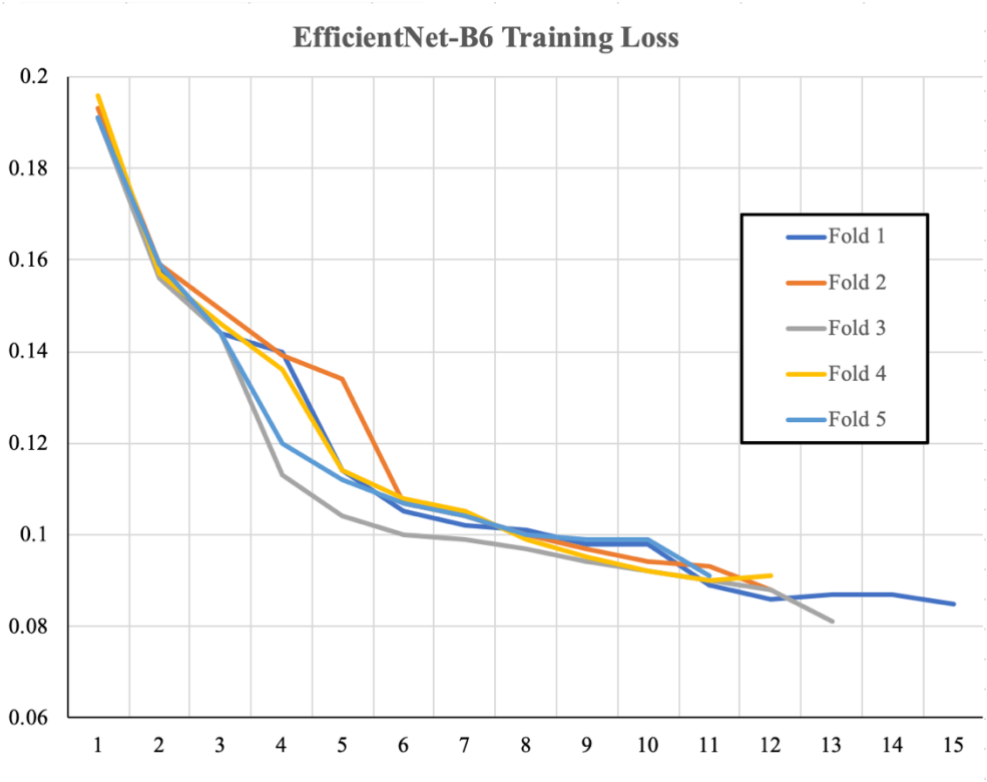

Figure 4. 1 EfficientNet-B6 Training Loss on 5-folds across 15 epochs

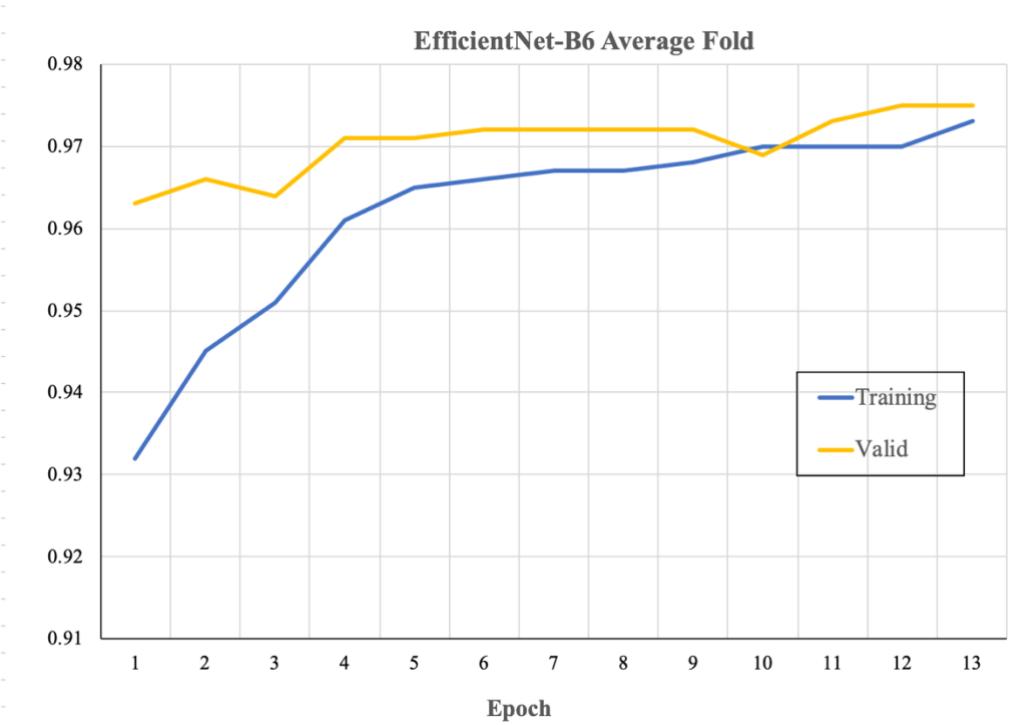

Figure 4. 2 Line graph of EfficientNet-B6 Training and Valid Accuracy

Performance of EfficientNet models (B2, B3, B4, and B6) and ResNet-50 on various folds, batch size, and epochs is listed in Table 4.1. The training time for a single model ranged from 8 hours to 29 hours for all five folds depending on the experiment setup discussed in Chapter 3. The best AUC score of each folds of a model is saved, which is then used for testing on Kaggle's official competition test set. The best private LB score and public LB score was obtained on EfficientNet-B6; **0.8688** and **0.8819** respectively. The private LB score is calculated with approximately 70% of the test data and public LB score is calculated with approximately 30% of the test data. The second best private and public LB score of 0.8662 and 0.8783 was obtained on EfficientNet-B4. One major finding that can be noted from Table 4.1 is that even though EfficientNet-B2, B3, and B4 models are trained on much fewer parameters (9.2M, 12M, and 19M respectively) than ResNet-50 (26M), they consistently outperform ResNet-50. Therefore, it is

better to use EfficientNet models when computational power is limited without compromising the performance.

Model	Fold	Avg. Train Accuracy	Avg. Valid Accuracy	Avg. AUC
ResNet-50	1	0.956	0.943	0.865
	2	0.954	0.973	0.919
	3	0.954	0.966	0.900
	4	0.927	0.951	0.856
	5	0.941	0.951	0.873
EfficientNet-B2	1	0.960	0.958	0.901
	2	0.958	0.977	0.920
	3	0.961	0.973	0.912
	4	0.967	0.975	0.924
	5	0.967	0.971	0.915
EfficientNet-B3	1	0.955	0.971	0.909
	2	0.959	0.970	0.914
	3	0.962	0.970	0.900
	4	0.952	0.966	0.916
	5	0.963	0.977	0.933
EfficientNet-B4	1	0.959	0.973	0.923
	2	0.965	0.974	0.928
	3	0.960	0.968	0.894
	4	0.957	0.973	0.915
	5	0.962	0.974	0.933
EfficientNet-B6	1	0.962	0.973	0.923
	2	0.959	0.968	0.920
	3	0.962	0.970	0.913
	4	0.960	0.970	0.914
	5	0.960	0.973	0.929

Table 4.1. Comparison of performance of EfficientNet models and ResNet-50. Avg Train Accuracy and Avg Valid Accuracy is calculated for each fold across 15 epochs for EfficientNet-B3, B4 and B6, and 24 epochs for ResNet-50 and EfficientNet-B2. Avg AUC is calculated on validation dataset for each fold.

CHAPTER 5

CONCLUSION AND FUTURE PLANS

From this experiment, it can be concluded that a good performing classification model can be built to detect melanoma using deep convolutional neural network models pre-trained on ImageNet and EfficientNet family models outperform ResNet-50 model. Being said that, there are a few things that can be done to improve the performance; use ensemble learning to ensure model makes the most stable and best possible prediction, train on multiple image sizes, use model with higher accuracy on ImageNet (e.g. EfficientNet-B7), and hyperparameter tuning (adjust learning rate, batch size, number of epochs , etc.). Some future plans for this project includes: a website or an user interface with functionality to enter patient-level contextual information, use ensemble learning on predictions from multiple models, and use the method to draw a random number of pseudo artificial hairs on images because a few Kagglers' were able to increase their public and private LB score.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Rotemberg, Veronica, Nicholas Kurtansky, Brigid Betz-Stablein, Liam Caffery, Emmanouil Chousakos, Noel Codella, Marc Combalia, et al. “A Patient-Centric Dataset of Images and Metadata for Identifying Melanomas Using Clinical Context.” arXiv.org, August 7, 2020. https://arxiv.org/abs/2008.07360.

[2] Hancock, J. T. and T. Khoshgoftaar. “Survey on categorical data for neural networks.” Journal of Big Data 7 (2020): 1-41.

[3] Buslaev, Alexander V., A. Parinov, Eugene Khvedchenya, V. Iglovikov and A. Kalinin. “Albumentations: fast and flexible image augmentations.” Inf. 11 (2020): 125.

[4] Krizhevsky, A., Ilya Sutskever and Geoffrey E. Hinton. “ImageNet classification with deep convolutional neural networks.” CACM (2017).

[5] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 770-778, doi: 10.1109/CVPR.2016.90.

[6] Tan, M. and Quoc V. Le. “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks.” ICML (2019).

[7] LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. “Deep Learning.” Nature News. Nature Publishing Group, May 27, 2015. https://www.nature.com/articles/nature14539.

[8] Han, Seung Seog, M. S. Kim, Woohyung Lim, Gyeong Hun Park, I. Park and S. Chang. “Classification of the Clinical Images for Benign and Malignant Cutaneous Tumors Using a Deep Learning Algorithm.” The Journal of investigative dermatology 138 7 (2018): 1529-1538.