Final Project Report for CS 184A/284A, Fall 2019

Project Title: Skin Cancer Prediction Based on MNIST: HAM10000

Project Number: 32

Student Name(s)
Yushan Zhang, 90042930, yushanz5@uci.edu
Ziyi Yang, 90545904, zyang18@uci.edu

1. Introduction and Problem Statement

This project develops a CNN model that is able to classify seven different types of skin cancer. We aimed to construct and refine a model to predict skin cancer types with high precision. The data we used is called "MNIST: HAM10000", containing over 10,000 labeled images of pigmented skin lesions. Through iterative optimization of model parameters, including using different optimizers, adjustments in learning rates, and varying the number of epochs, we are trying to find a model that predicts the highest accuracy on the test data set.

Our findings indicate that our CNN model can achieve a precision rate of 77.78% in classifying skin cancer types from test images. This level of accuracy demonstrates CNN's potential as a supportive tool for diagnosing skin cancer, potentially leading to earlier and more accurate treatment for patients.

2. Related Work

In the past, CNN has shown significant potential for skin cancer classification. Some classific models like VGG-16, googleNet, and DRN-50, [1, 2] and even self-made easy models, [3] have outstanding performance on accuracy and sensitivity. For VGG-16, it can achieve 0.903 accuracy and 0.794 sensitivity, while increasing layers increases the accuracy and sensitivity, until around 50 layers in so-called FCRN-50 models. Its accuracy can achieve 0.949 accuracy and 0.897 sensitivity with C++ and Matlab-based Caffe libraries. [1] However, other research groups used another method that maintains higher accuracy with fewer layers for a model in CNN. In that self-made model from Yunendah Nur Fuadah's group, they narrowed the image into only 3 hidden layers, but still achieved 99% accuracy on melanoma from ISID 2016 datasets with 0.0346 loss. [3] However, the classic CNN model can't handle the MNIST: HAM10000 datasets. As the states show, all attempts of classic CNN can achieve around 75 percent accuracy[4] for the datasets and more accuracy by combining other methods like transfer learning [5], Swish Activation Function [6], or Resnet model.[7]

3. Data Sets

The dataset we used is called "MNIST: HAM10000", an open-source dataset about pigmented skin lesions [1]. This data set contains 10015 images. Each image has 2352 pixel values and 1 label number. The diagnostic classes in the dataset have 7 distinct values and are provided as follows[8].

- 0: 'akiec' – Actinic keratoses and intraepithelial carcinoma

- 1: ‘bcc’ – basal cell carcinoma
- 2: ‘bkl’ – benign keratosis-like lesions
- 3: ‘df’ – dermatofibroma
- 4: ‘nv’ – melanocytic nevi
- 5: ‘vasc’ – vascular lesions
- 6: ‘mel’ – melanoma

We reshape each image into (28 pixels in height, 28 pixels in width, and 3 layers) for normalization. This helps to decrease the computational load during the training process. The figures below show different cancer examples after reshaping.

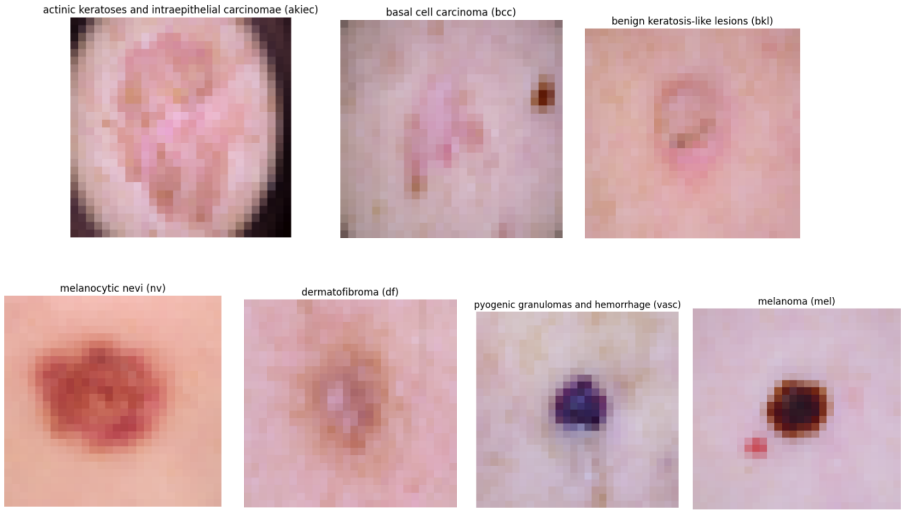

Figure 3.1: Random images selected from the dataset

The overall data is set into 64% training, 16% validation, and 20% testing respectively.

The histogram of the data (Figure 3.2) indicates that melanocytic nevi have significantly larger amounts of data than the others. This means the distribution of the data is not equal. If we use the data directly to train the model, this will make the model learn more about the melanocytic nevi rather than another type of cancer.

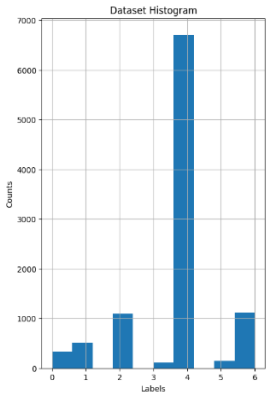

Figure 3.2

Data Before

Labels	Frequency
4	5364
6	907
2	875
1	406
0	265
5	106
3	89

Figure 3.3.1

Data After

Labels	Frequency
4	5264
6	5264
2	5264
1	5264
0	5264
5	5264
3	5264

Figure 3.3.2

To overcome the imbalance in the data set, an oversampling method is introduced to training and validation sets, which randomly duplicates the images in the labels with small data amounts. After processing, the training data and the validation data contain 37548 images in total. The result of the final data amount for each label is shown in Figure 3.3.2.

4. Description of Technical Approach

To classify the different skin cancers, CNN has been chosen. In our project, we use TensorFlow, an advanced open-source software library for machine learning, to build and train our Convolutional Neural Network (CNN) for image classification tasks. TensorFlow offers a comprehensive, flexible ecosystem of tools and libraries that support the development of complex machine-learning models.

A key component of TensorFlow that we utilized is Keras, an open-source software library that provides a Python interface for artificial neural networks. Keras is integrated into TensorFlow as tf. keras, making it TensorFlow's high-level API for building and training deep learning models.

Before fitting images into the model, we resize each image from 2532 pixels value into 3 layers, 28 pixels times 28 pixels, which makes each layer a square image. Then, these resized images are given to the Convolution Neural Network to train and test the model.

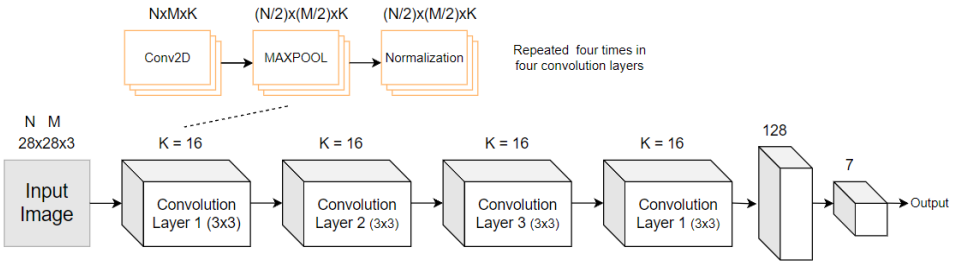

Figure 4.1

Based on the model in the proposal, we build our first model shown in Figure 4.1. It has repeated four convolution layers and a fully connected layer, which contains one dropout layer, one flattened layer, and two dense layers. This model includes 11255 parameters in total.

After testing serval times, we figured out that the proposed model was not precise enough to predict the test data. The best prediction accuracy we get on the test data is 63.2%. From examining the first model, the main problem of the low accuracy could come from the same size of filters, which cannot extract enough information from the image. Therefore, our second model is introduced by changing the filters from ‘16-16-16-16’ to ‘16-32-64-128’. In addition, one more dense layer and one more normalization layer are added into the fully connected layer for learning more complex relations in the data. This new model now contains 428263 parameters and the visualization structure is shown below:

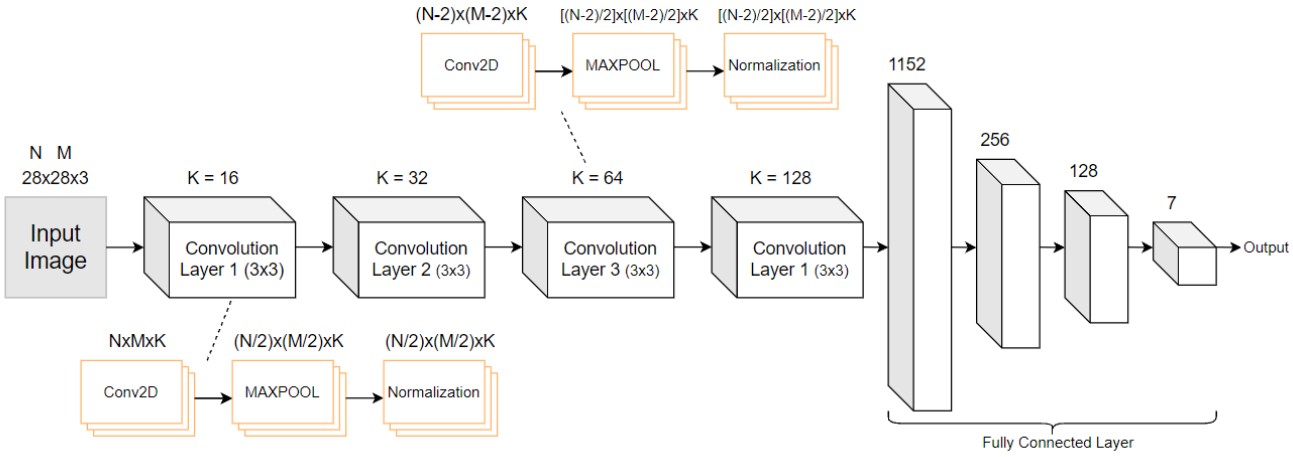

Figure 4.2

4. Software

(a) **Code or Scripts Written by Our Team**

Software/Script	Functionality	Input	Output
CNN Model Building	Train the CNN Model for image classification	Resized Image	Trained CNN Model
Model Tuning	Use different parameters to find the best-predicted accuracy.	Different parameters(optimizers, learning rates, and epochs.)	Different model
Model Evaluation	Evaluate the performance of the trained model	Test data, Trained model	Model performance

(b) **Code or Scripts Written by others**

Software/Script	Source	Functionality	Input	Output
TensorFlow	TensorFlow	Machine learning library	Depend on need	Depend on need
Keras API	Integrated in TensorFlow	Simplified building and training of deep learning model	Model Parameters	Model
RandomOverSampler	Imblearn. oversampling	Randomly duplicate the data that has fewer amounts	Raw Train Features, Raw Train Labels.	Train data Contains the Same Amount of Data for Each Label.
Image Distribution histogram	https://www.kaggle.com/code/raniaioan/starter-skin-cancer-mnist-ham10000-6a5a3b01-0	Draw the histogram of the dataset. (x,y) is (amount of images in the label, label)	Dataset, Number of Graphs Shown, Number of Graphs per Row	A Histogram of the Dataset
Confusion Matrix	Sklearn	Measuring the Performance of a Classification Model	Prediction Labels, True Labels	Test Data Analyzing Matrix

5. Experiments and Evaluation

After deciding on the model we would use for cancer classification, we researched how to improve the accuracy by testing the learning rate, optimizer, and epochs to decide which sets of optimizers would fit the most. For evaluation, we used accuracy and loss as our evaluation mechanism. The Accuracy is calculated by all True events, including correct true and false judgments, divided into all tests, including all judgments the machine made. Loss is how serious the machine made for a bad prediction. According to the data from research [1] more epochs means possible higher accuracy and lower loss until around 100 epochs for their model. At the same time, more epochs means more training time spent during evaluation. In order to balance the spending time and evaluation efficiency, we tested the learning rate with Adam optimizer and 16 epochs. In many research groups[1][3][4], Adam is considered the best optimizer among all other optimizers on accuracy in general skin cancer research among multiple researchers. The choice of 16 epochs is chosen due to research that shows that similar model training performs the best around 3, 6, 9, and 16 epochs, while 16 epochs maintain a good enough loss rate. [4] Table 5.1 represents the result of our tests. As the table shows, when the learning rate is 0.0008, the accuracy achieved peak.

Then we test the model with different optimizers to see the appearance of the model. In previous research, SGD, RMSprop, and Adagrad are popular and have the potential to overcome the performance of the model. So we tested the accuracy with a set learning rate and epochs 16. Figure 5.1 shows that Adam maintains the best performance at a 0.0008 learning rate, and the RMS prop has a tendency for a smaller learning rate will get higher accuracy.

Learning Rate	Test Accuracy	Loss
0.01	0.7249	1.0945
0.005	0.7559	1.0298
0.002	0.7599	1.3471
0.001	0.7619	1.4389
0.0009	0.7504	1.1048
0.0008	0.7778	1.4412
0.0007	0.7259	1.5426
0.0005	0.7529	1.1546
0.0001	0.7429	0.8016

Table 5.1 Adam's performance on different learning rates

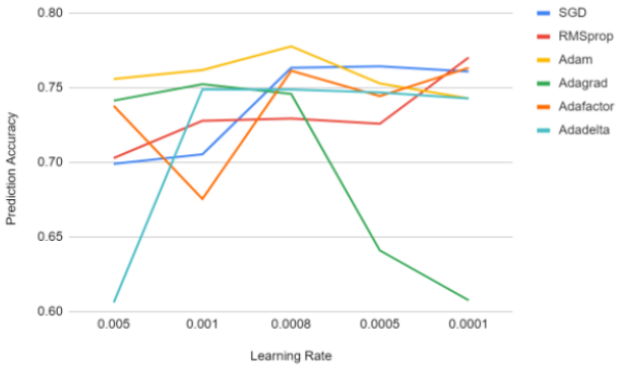

Figure 5.1 Different optimizers performance on learning raes

Finally, the effect of the number of epochs is tested from 5 to 50 with Adam optimizer and 0.0008 learning rate. In Table 5.2, from epochs 5 to 16, there are general increases in accuracy and loss. From 20 epochs, the accuracy stoked around 76% and got lower when epochs got over 50 and achieved the highest loss.

Epoch number	Accuracy	Loss
5	0.7039	0.962
10	0.7209	1.1365
16	0.7778	1.4412
20	0.7524	1.246
25	0.7509	1.7894
30	0.7634	1.7497
35	0.7564	1.6538
40	0.7708	1.6626
45	0.7599	1.6067
50	0.7384	1.8038

Table 5.2 Adam's performance on different epochs with learning rate 0.008

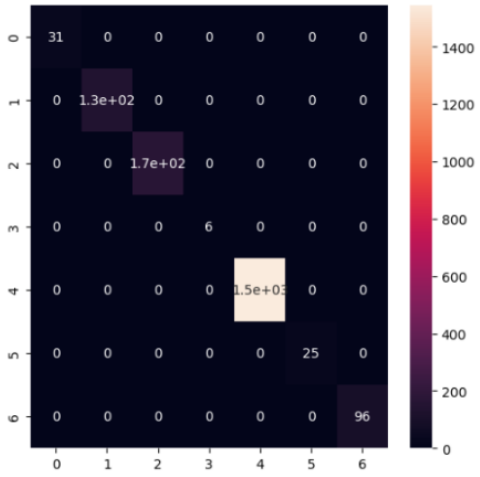

Figure 5.3 Heat map

In order to achieve the highest accuracy, the final set of optimizers is set as 0.0008 learning rate in Adam optimizer and running 16 epochs. For the final model performance, the confusion matrix is improved and judging the specific loss the model made. Confusion matrices are made by presenting how the model performed in each category. Each cell represents how many tasks the model made. As Figure 5.3 shows, which shows that our model has successfully classified 1500 figures in class 4, 170 in class 2, and 130 in class 1. While only 6 and 25 are classified correctly in class 3, and class 5.

6. Discussion and Conclusion

This project provides the opportunity for us to dig deeper into the CNN model during the process of finding the optimal test accuracy. By adding appropriate layers and using proper parameters, the model's performance can vary a lot. From the first model we built to the current model we use, the test accuracy improved from 62.3% to 77%, which is a good result from our perspective. However, this accuracy for CNN is lower than most current CNN models for predicting skin cancers, which can go up to 95%. There are some limitations to our idea that may cause the accuracy to stay lower than that:

1. **Image Resizing**: We use the resized image rather than the whole image. In the beginning, we thought CNN should go through a square image to make the model better. But when we see the result, we think it might be better to use the original image size since this won't lose so many details in the image and thus give better results. The disadvantage of this change might be the model training time goes up longer than usual, which makes each change from tuning the parameters harder.
2. **Model Architecture:** The model we build can still be improved to reach a better result. By adding more convolution layers and changing the fully connected layer, the model could capture more accurate details.

3. **Data Splitting Strategy:** The dataset split could be reconsidered. We currently have 64% in the training data set, 16% in the fixed validation data set, and 20% in the test data set. Serval changes could improve this part:
    a. Use different potions of separation to find a better result. eg. Increasing the training data amount to 70% and decreasing the test data amount to 15% may work better.
    b. Using the cross-validation method while training the model. This could help the model predict better on various data rather than a fixed data set.
    c. Adding higher weights on images that are in labels 0, 3, and 5. Based on the confusion matrix we got, the model cannot properly deal with the labels that have a small amount of samples. This is probably one of the main reasons why our model cannot reach 80% accuracy.
4. **Oversampling:**
    a. Due to the lack of data in categories 0, 3, and 5, oversampling images in order to overcome the small data size may not work properly for the model. We should find a method about how to train the model with little data but perform well.
5. **Segmentation:**
    a. Another possible processing may be using segmentation in the model, which will make a mask for a specific area of skin cancer. This could be helpful for images with hair or other covers on the cancers.

If we were in charge of a research lab, the most important thing is to figure out a better model for predicting images and it should be over 90 % accuracy. After achieving this, we want to find a solution that can use less data in the training set but can predict a high accuracy on the test data.

Reference:

[1] G. S. Jayalakshmi and V. S. Kumar, "Performance analysis of Convolutional Neural Network (CNN) based Cancerous Skin Lesion Detection System," *2019 International Conference on Computational Intelligence in Data Science (ICCIDS)*, Chennai, India, 2019, pp. 1-6, doi: 10.1109/ICCIDS.2019.8862143.

[2] Julia Höhn, Eva Krieghoff-Henning, Tanja B. Jutzi, Christof von Kalle, Jochen S. Utikal, Friedegund Meier, Frank F. Gellrich, Sarah Hobelsberger, Axel Hauschild, Justin G. Schlager, Lars French, Lucie Heinzerling, Max Schlaak, Kamran Ghoreschi, Franz J. Hilke, Gabriela Poch, Heinz Kutzner, Markus V. Heppt, Sebastian Haferkamp, Wiebke Sondermann, Dirk Schadendorf, Bastian Schilling, Matthias Goebeler, Achim Hekler, Stefan Fröhling, Daniel B. Lipka, Jakob N. Kather, Dieter Krahl, Gerardo Ferrara, Sarah Haggenmüller, Titus J. Brinker, Combining CNN-based histologic whole slide image analysis and patient data to improve skin cancer classification, European Journal of Cancer, Volume 149, 2021, Pages 94-101, ISSN 0959-8049

[3] Yunendah Nur Fu'adah1, NK Caecar Pratiwi1, Muhammad Adnan Pramudito1 and Nur Ibrahim, Convolutional Neural Network (CNN) for Automatic Skin Cancer Classification System
al 2020 IOP Conf. Ser.: Mater. Sci. Eng. 982 012005

[4]K. Pai and A. Giridharan, "Convolutional Neural Networks for classifying skin lesions," *TENCON 2019 - 2019 IEEE Region 10 Conference (TENCON)*, Kochi, India, 2019, pp. 1794-1796, doi: 10.1109/TENCON.2019.8929461.

[5]Garg, R., Maheshwari, S., Shukla, A. (2021). Decision Support System for Detection and Classification of Skin Cancer Using CNN. In: Sharma, M.K., Dhaka, V.S., Perumal, T., Dey, N., Tavares, J.M.R.S. (eds) Innovations in Computational Intelligence and Computer Vision. Advances in Intelligent Systems and Computing, vol 1189. Springer, Singapore.

[6]Farheen, M., Manjushree, M., & Pandit, M. K. (2020). Skin Cancer Detection using CNN with Swish Activation Function.

[7] U. Jamil, M. U. Akram, S. Khalid, S. Abbas, and K. Saleem, ‘‘Computer based melanocytic and nevus image enhancement and segmentation,’’ BioMed Res. Int., vol. 2016, pp. 1–13, Jan. 2016.

https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8929461

[8] MNIST: HAM10000 dataset

“https://www.kaggle.com/datasets/kmader/skin-cancer-mnist-ham10000/data”

[9]Jose Luis Garcia-Arroyo, Begonya Garcia-Zapirain, Segmentation of skin lesions in dermoscopy images using fuzzy classification of pixels and histogram thresholding, Computer Methods and Programs in Biomedicine, 168, 2019, 11-19.

Individual Contributions:

Yushan Zhang:

- Collect the data set from the website.
- Write the base code for the project. Modify the model based on the model's outcome.
- Write the final report's part 3, part 4, and part 6. Process the tables and figures. Organize the report's format.
- Build the Github and upload the code.

Ziyi Yang:

- Running evaluation and making states graph for part 5
- Writing the evaluation code and tests optimizers
- Write part 2 , 5, and 6 in the final report and look for resources of the background information.

Together: Tune the model and find the best among the results.