

---

# BME 590L: Optimizing Gray-scale Image Formation Parameters in CNN Skin Cancer Detection

---

**Yangpei Liu, Changxiang Sui, Sumin Lan**

Department of Biomedical Engineering

Duke University

Durham, NC 27708

y1642@duke.edu, cs557@duke.edu, sl597@duke.edu

## Abstract

Training process, which demands tons of time, is gradually becoming a major concern in machine learning field. Thus, some directions have been turned into designing physical layers that then can be adopted to help build real-world imaging systems. This kind of imaging systems is aimed at generating images for computer vision, but for human appreciation. Therefore, images acquired to feed into neural networks performing different work of tasks, ranging from classification to segmentation, make networks much faster to train and generalize for newly acquired images. In this paper, we examine imaging processes regarding illumination phase, aperture phase, and gray-scale image formation for human skin cancer detection (HAM10000). Results show that these physical layers appear to have only a slight improvement on classification. However, what is most interesting to us is to compare between each of these physical parameters and their final trained weights, which are supposed to be highly related to the image configuration we feed into our CNN.

## 1 Instructions

### 1.1 Problem Description

A significant number of people throughout the world are affected with skin cancers. Globally, it is estimated that more than 1 million new cases of nonmelanoma skin cancer were diagnosed in 2018 with 65,000 associated deaths, and more than 280,000 new cases of malignant melanoma were diagnosed, with 60,000 associated deaths [1]. Besides, skin cancer is epidemic in the United States. An estimated 5.4 million total keratinocyte cancers in the United States occur in three million persons annually. Melanoma ranks in the top five most common cancers in the United States [2]. These data show a rapidly rising incidence of skin cancers over the past 30 years. Thus, it is important that the skin cancer has to be detected as early as possible before the cells start invading and spreading.

### 1.2 Data Description

The data we are using is the HAM10000 ("Human Against Machine with 10000 training images") dataset [3]. This dataset collected dermatoscopic images from different populations, acquired and stored by different modalities which can serve as a training set for academic machine learning purposes. Cases include a representative collection of all important diagnostic categories including melanocytic nevi, dermatofibroma, melanoma, Actinic keratoses and intraepithelial carcinoma/Bowen's disease, benign keratosis-like lesions, basal cell carcinoma, and vascular lesions. However, accounting for the fact that this dataset consist of a large collection of multi-source dermatoscopic images of common pigmented skin lesions, huge differences between image configurations and formats are perceived, which predictably will undermine classification accuracy.

### 1.3 Tentative Approach

The hypothesis is based on human eyes are born to be more sensitive to green light rather than red and blue, however, machine should not be constrained by human vision, so expected trained weights should hold different values for RGB channel weights, which in return relates to imaging patterns of cameras in the real-world optimizing over spectrum.

There are three main components in this project: 1) Image database exploration by augmentation in order to reduce the degree of the label distributed unevenness. 2) Creating and identifying a convolutional neural network for image classification 3) Optimizing physical layer weights like RGB color channels weights. The process may reduce the weight of green color and correspondingly increase the weights of red and blue colors. Later, we will create confusion matrix for classification accuracy analysis.

## 2 Related Works

There have been several competitions held since last few years for training neural networks classifying HAM10000 dataset, but nearly few of them attempted to design physical layers before feeding them into latter classification layers. Although so, they do offer a great deal of insights to us when designing our model.

Ge, Li et al. mentioned about their concerns of dealing with imbalanced dataset, where each class does not make up an equal portion of the dataset [4], which brought us a caveat when considering our own dataset when processing. Our case here is that in fact HAM10000 is extremely imbalanced (described in details below in section 3. Method), therefore, we are in need of dataset augmentation for underrepresented labels. Also, we have noticed that it is hard and even naive to simply assess performance of models trained by unevenly distributed dataset by accuracy. Instead, we examine by confusion matrix (part of codes adopted from Siddhartha [5]).

Beforehand, we established our primitive CNN model configuration resembling what we continually used in coursework (two conv2D layers, one pooling layer, two conv2D layers, one pooling layer, and two dense layers), which wasn't deep enough and did not work well to our benefits. Therefore, for general deep CNN configuration we adopt the Caffe network [6], which is a fourteen-layer CNN excluding max-pooling. Yet, simply using this model does not provide good generalization to out-sample data; thus, in addition, we adopt drop layer configuration from Siddhartha [5].

Some previous work also played with hyper-parameters like batch size and learning rate. A typical number of batch size is always set to 32, but for our case, in order to incorporate each label an appropriate times in each batch, we need to account for labels that only make up 7% of the dataset (after selective augmentation); thus, we pick the number 128. In this case, after randomization, each label should occupy at least 8 samples in every batch. For learning rate, a default value in keras is 0.001, so we started at this value, but we also tried 0.0001 after seeing some researchers choose 0.0001 when applying for their training process.

## 3 Method

### 3.1 Pre-processing

The HAM10000 dataset containing a large collection of multi-source dermatoscopic images of common pigmented skin lesions, which are then sorted into labels 0 to 6 according to 7-string categories. Due to the limitation of RAM and for the sake of simplicity, we resize the 10015 skin lesion images from 450x600x3 to 90x120x3 using tensorflow.image.resize method.

### 3.2 Augmentation

After inspecting the number of images with different labels in the HAM10000 dataset, we ascertain the problem of imbalance image data where images of label 1, 2, 3, 6 is obviously fewer than label 0, 4 and 5 (examined from Table 1). In order to solve this issue, there are basically three approaches. The first is that we can use weighted crosstropy loss function, giving more weight to underrepresented samples. Although it is the easiest to setup, it could take much time to play with the weights and

Table 1: Data Proportion of each label before and after augmentation

Before augmentation							
Label	0	1	2	3	4	5	6
Proportion	10.97%	3.27%	5.13%	1.15%	11.11%	66.95%	1.42%

After augmentation							
Label	0	1	2	3	4	5	6
Proportion	8.03%	7.13%	7.50%	9.33%	8.37%	50.17%	9.47%

practices suggest its inefficiency. The second way points to balanced batching [7], so that each batch also has the same proportion of classes, but we gave up because of the inconsistency with our established tensorflow training function. We finally choose the third approach which is to selectively augment the underrepresented classes more than others to create a balanced set. By using python `skimage.transform.rotate` module to rotate the images of label 1, 2, 3, 6 with 2, 1, 10, 8 times of  $18^\circ$  respectively, we append the images size to 2, 1, 10, 8 times of formal size. See Table 1 for proportion of images of each label before and after augmentation.

### 3.3 Parameters for iteration strategy

To show the effects of the parameters for iteration strategy (i.e., learning rate and epoch number), we ran the model with physical layers disabled with different iteration values. We decide to use 0.0001 for learning rate after trying 0.001, 0.0002, 0.0005 and 25 for epoch number for all the models after trying 8, 10, 25, 30, 50. Although, 25 epochs lead to a some overfitting when training the model with illumination phase, aperture phase and ‘colorweighing’ layer, we have to control these variables for further discussion.

### 3.4 Physical Layer

Human eyes are born to be more sensitive to green light rather than red and blue; thus, traditional cameras are made to put more weights on green light to form images. To this aim, built-in python algorithm ‘`rgb2gray`’, which performs transformation from RGB images to grayscale images, emphasizes more on green color as well. However, machine learning should not be constrained by human vision. Therefore, we explore weights of image color channels (RGB) that contribute to form grayscale images (optimizing color filter for a grayscale camera, in our model defined as ‘colorweighing’ layer), which then are fed into following forward model combining convolutional neural network (CNN) for classification. We also explore whether different combinations of other physical layers, like illumination phase and aperture phase interplay with color channel weights (‘colorweighing’ layer) alone or together, as well as different initialization of ‘colorweighing’ layer will affect classification accuracy and final trained weights of image color channels. Thus, we develop 7 training process in total, including CNN model with physical layers disabled for training, CNN model with illumination phase and ‘colorweighing’ layer, CNN model with aperture phase and ‘colorweighing’ layer, CNN model with illumination phase, aperture phase and ‘colorweighing’ layer and CNN model with only ‘colorweighing’ layer. The last one are divided into 3 sub-training processes according to different RGB initial value containing python ‘`rgb2gray`’ RGB weights ( $Y = 0.299 R + 0.587 G + 0.114 B$ ) which is also used in the three models mentioned earlier, equal RGB weights ( $Y = 0.333 R + 0.333 G + 0.333 B$ ) and random RGB weights using `tensorflow.random.truncated_normal`. To justify effectiveness of our physical layers, we perform confusion matrix for classification accuracy analysis.

## 4 Results

### 4.1 Classification results of original images with different models

We first show the classification performance, which means how well our models with or without physical layer. Table 2 and table 3 illustrate the results of our classification experiment. In the

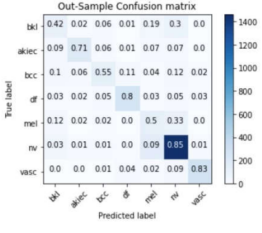
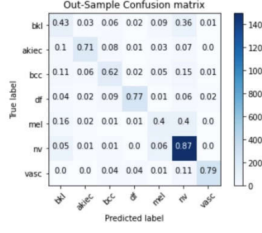
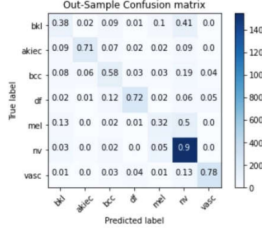
classification of original images from the HAM10000 dataset, we've created 7 types of models which we have mentioned above. Considering the accuracy, all models performed not quite well in the test data classification with accuracy between 70% and 80%. But we still could observe the relationship from their comparison. Particularly, model 3(model with aperture phase and 'colorweighing' layer) has done the best job with an accuracy of 76.3%, while model1(with physical layers disabled for training) has done the worst, because of no physical layer conduction. When considering different RGB ratio in 'colorweighing' layer, it seems there is not too many differences while it will increase rapidly when adding aperture phase or illumination phase. However, it is quite different when we evaluate the classification performance of our classifier using the confusion matrix over our validation set. For the darkest unit shown in the matrix, the label 5(nv) has the highest ratio of True value and predicted label among all the models. This reflects the high-accuracy prediction. Based on the previous content, we can see the label 5(nv) has large amounts of observations(up to 66.94%), which has definitely great deal of influence on accuracy. Besides, benefiting from the augmentation, label 3(df) and label 6(vasc) tend to have high ratio and even closed to label 5. But there are some other decisive influence. Label 1(akiec) and label 3(df) only accounts for about 7% 8% after augmentation, though, they have a high prediction ratio. For comparison between models, interestingly, the last 3 models display a tendency that the lower the ratio of df and higher ratio of mel(although their accuracy is still very small). Furthermore, the label 5(nv) is also the most frequently wrong prediction, which means many other labels will be recognized as label 5(nv).

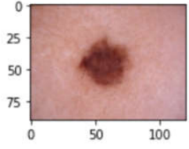
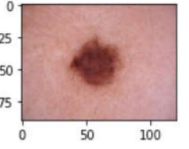
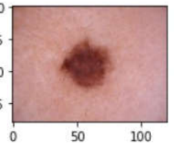
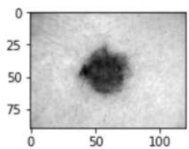
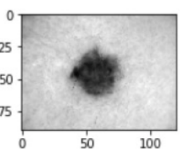
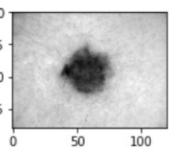
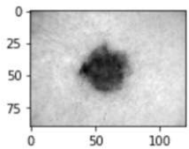
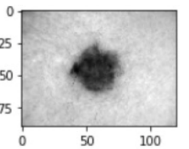
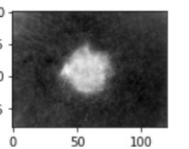
#### 4.2 Final trained weights of three color channels

From Table 2, we can see for the CNN model with illumination phase, aperture phase and 'colorweighing' layer, the final trained weights remain the same from initial RGB weights, while for the CNN model with 'colorweighing' layer and either illumination phase or aperture phase, the weights of R and B decrease, indicating the significance of green channel for diagnosis of skin cancers. This trend can also be proved when we set different initial value of RGB weights except the last column in Table 3. When we set the initial value to be random, the final trained weights appear to be negative. Then we compare the grayscale image formed by python algorithm 'rgb2gray' and the trained weights for different initialization of 'colorweighing' layer in Table 4. The grayscale image looks the same for the first two fixed initial weights, while turns to the opposite color for random RGB initial weights.

	CNN model with physical layers disabled for training	CNN model with illumination phase, aperture phase and 'colorweighing' layer	CNN model with aperture phase and 'colorweighing' layer	CNN model with illumination phase and 'colorweighing' layer
Test accuracy	73.8%	75.4%	76.6%	76.3%
Test loss	0.9985	0.9497	0.9071	0.9252
Confusion Matrix				
Illumination phase	N/A			
Aperture phase	N/A			
Trained weights	N/A	[0.267, 0.621, 0.110]	[0.299, 0.587, 0.114]	[0.273, 0.616, 0.109]

Table 2. Test accuracy, test loss, confusion matrix, phase and final trained weights of CNN model with different combinations of physical layers

	CNN model with only 'colorweighing' layer ( $Y = 0.299 R + 0.587 G + 0.114 B$ )	CNN model with only 'colorweighing' layer ( $Y = 0.333 R + 0.333 G + 0.333 B$ )	CNN model with only 'colorweighing' layer (random RGB weights)
Test accuracy	74.8%	74.8%	74.5%
Test loss	0.9276	0.9348	1.1874
Confusion Matrix			
Trained weights	[0.272, 0.617, 0.109]	[0.293, 0.387, 0.338]	[-0.277, -0.972, -0.274]
Table 3. Test results, confusion matrix and final trained weights of different initialization of 'colorweighing' layer			

	CNN model with only 'colorweighing' layer ( $Y = 0.299 R + 0.587 G + 0.114 B$ )	CNN model with only 'colorweighing' layer ( $Y = 0.333 R + 0.333 G + 0.333 B$ )	CNN model with only 'colorweighing' layer (random RGB weights)
Original Image			
Grayscale image formed by python algorithm 'rgb2gray'			
Grayscale image formed by final trained weights			
Table 4. Original image and Grayscale image of different initialization of 'colorweighing' layer			

## 5 Discussion and Future Directions

As expected, we do see some improvement on classification accuracy of seven labels (see confusion matrices in Table 2 and 3), especially for those that are underrepresented, although their accuracies are still limited after applying physical layers in terms of label 0 (bkl) and 4 (mel). This makes sense because physical layers introduce additional degrees of freedom for networks to be trained. If improving accuracy is necessary in future work, we will need to look into details of those images to determine how to incorporate other features extracted from original images to feed simultaneously to classification networks. Nonetheless, What is more intriguing here is to compare between each model we build with regards to their generalization performance and final trained weights.

## 5.1 Interplay of Different Physical Layers

We can see from Table 2 that combinations of optimization for illumination phase and aperture phase work not as well as optimization for only one of them at a time. What is also worth-noting is that final trained weights of illumination phase under such combination situation fluctuate more drastically, which might be attributed to the fact that too much freedom is introduced. Even around corners of images, where lesions are less likely to appear, there is still much fluctuation in illumination phase map. Such weights in intuition are not that important compared to weights in the map center. We can confirm this by examining illumination map acquired from model 4, where we disable optimization for aperture phase. Trained illumination phase map here fluctuates slightly more in the middle, where lies the skin lesions. Two available trained aperture phase maps only hold trained values in the inner circle, that is due to the aperture shape we apply for these models; thus, there are no gradients outside the circle, which are indispensable for updating weights.

What we want to focus the most here is the color channel weights we train for each case. During the training process, we find that if initial values are set randomly, the final trained weights are diverging; however, if we examine the grayscale images formed using those differently initiated trained weights, they look nearly the same in the sense of format. They resemble the inverted version of classical grayscale images formed by `rgb2gray`, just like the lower left image in Table 4. This makes sense in that normally transformed grayscales hold high values (white) on the normal area, but low values on the lesion area (see mid row of Table 4). However, randomly initialized neural networks try to set high values on which area that affects the most for classification. We also note that ascribing from non-linearity introduced by activation functions and etc., there tend to be many local minima when training the network. Therefore, when initial values of 'colorweighing' layer vary, the final trained weights vary, and they optimize to some value close to their corresponding initial values.

## 5.2 Future Directions

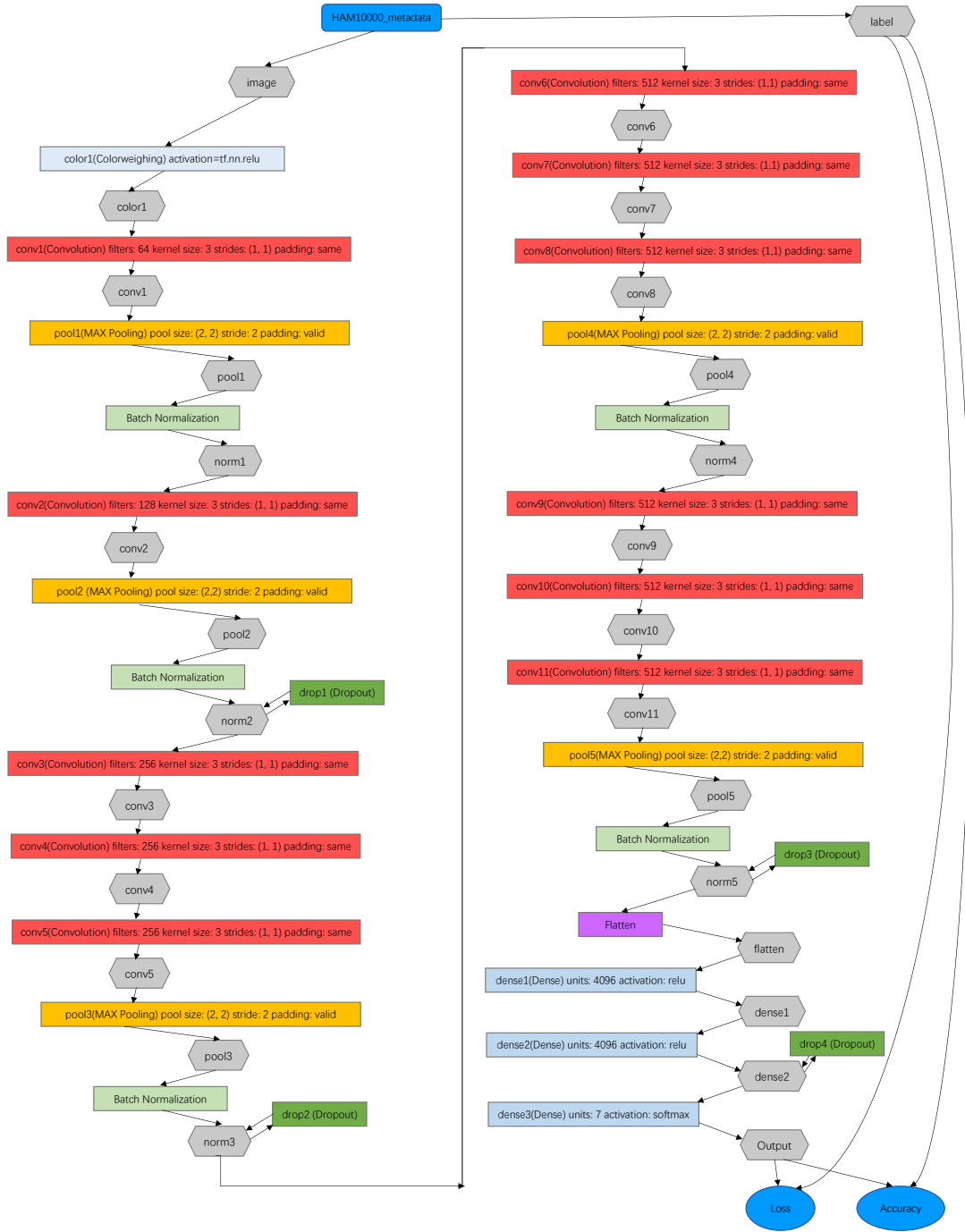
As mentioned above, future directions could be taken on extracting other image features from original images to perform classification tasks. In addition, we are also interested in ensemble models of differently formatted physical layers as they yield different results, which may help neural network understand or analysis these skin lesion images better. Due to limit quantity of time, we do not concentrate much on regularizers for physical layers, we consistently use L1 loss for 'colorweighing' layers trying to restrain trained weights to be non-negative, but they do not work well. Thus, we will do more research on applying regularizers to physical layers, which is needed for conducting physically-reasonable imaging process.

## Acknowledgments

We appreciate Dr. Roarke, Amey and Jun's help during this semester. Dr. Roarke provided us with valuable suggestion on project directions and dataset selection; Amey supported us a lot during the ongoing of our project, even though he was away in India right then; Jun brought us with constructive insights in TA sessions.

## References

- [1] Strowd, L., 2019. Shibboleth Authentication Request. [online] [www.clinicalkey.com.proxy.lib.duke.edu](http://www.clinicalkey.com.proxy.lib.duke.edu). Available at: <<https://www-clinicalkey-com.proxy.lib.duke.edu/#!/content/playContent/1-s2.0-S073386351930052X?returnurl=null&referrer=null>> [Accessed 19 April 2020].
- [2] Loescher LJ, Stratton D, Slebodnik M, Goodman H Systematic review of advanced practice nurses' skin cancer detection knowledge and attitudes, clinical skin examination, lesion detection, and training. *Journal of the American Association of Nurse Practitioners*. 2018;30(1):43–58. doi: 10.1097/JXX.0000000000000004.
- [3] Philipp, T., 2020. The HAM10000 Dataset, A Large Collection Of Multi-Source Dermatoscopic Images Of Common Pigmented Skin Lesions. [online] Harvard Dataverse. Available at: <<https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/DBW86T>> [Accessed 19 April 2020]
- [4] Ge, Yunhao, et al. "Melanoma segmentation and classification in clinical images using deep learning." *Proceedings of the 2018 10th International Conference on Machine Learning and Computing*. 2018.
- [5] M. Siddhartha, "Step wise Approach : CNN Model (77.0344% Accuracy)", Kaggle.com, 2020. [Online]. Available: <https://www.kaggle.com/sid321axn/step-wise-approach-cnn-model-77-0344-accuracy#Data-Augmentation>. [Accessed: 19- Apr- 2020]
- [6] Nasiri, Sara, et al. "Deep-CLASS at ISIC Machine Learning Challenge 2018." *arXiv preprint arXiv:1807.08993* (2018).
- [7] G. Lemaitre, 2020. Porto Seguro: Balancing Samples In Mini-Batches With Keras [online] [Imbalanced-learn.readthedocs.io](https://imbalanced-learn.readthedocs.io). Available at: <[https://imbalanced-learn.readthedocs.io/en/stable/auto\\_examples/applications/porto\\_seguro\\_keras\\_under\\_sampling](https://imbalanced-learn.readthedocs.io/en/stable/auto_examples/applications/porto_seguro_keras_under_sampling)> [Accessed 21 April 2020].



Appendix: CNN Model Configuration