**Multiple Convolutional Neural Network for Skin Dermoscopic Image Classification Project**

**Submitted by**

**Shunqi Zheng Junior**

**Dhavani Thummar Graduate**

**Manoj vanga Graduate**

**Computer Science department**

**Gildart Haase School of Computer Sciences and Engineering**

**project report paper submitted in partial fulfillment of the requirements for the course - Data Warehouse & Data Mining**

**Under the supervision of**

**Dr. Avimanyou Vatsa**

**Assistant Professor, Department of Computer Science**

****

**(Spring 2020)**

**Problem Statements**

The manual skin lesions detection system is human-labor intensive, which needs magnifying and illuminated skin images to improve the clarity of spots. Moreover, the classification process of the dermoscopy images is still confusing because of the presence of artifacts, noises, the complexity and changeability of the skin lesion structures that can not meet the requirements of the lesion’s border detection, feature extraction, and classification processes. They need tools, which are able to detect early-stage skin cancers and delineate them properly from healthy tissue. They do spectral distributions of four lesions but distributions are overlapping so melanoma is hard to recognize in a clinical study.

**Background**

Skin abnormalities are the original indicator of many diseases like skin cancer that is one of the main health problems throughout the world. There are two main lesion types (identification and classification) which are the key process in diagnosis and treatment strategy. So, in order to identify skin disease, an important process called melanoma classification is challenging for traditional physicians because of many existing factors that affect its result.

In this study, we have a small data set (n=61) of hyperspectral images. The data set consists of several lesions, which were imaged and diagnosed by histopathology. Lesions consist of malignant melanomas, melanoma in-situ, dysplastic nevi and benign nevi. Spectral data cube has three-dimensional nature, thus, standard 2D convolutional neural networks might not be enough to utilize spectral data.

We use the HAM10000 dataset which contains 10015 dermatoscopic skin lesion images of seven classes. We implement a hugely popular method called dilated convolution for classifying seven skin lesions with transfer learning techniques. The ensemble uses the DCNN (deep convolutional neural network) method, where they combine the result of four different architectures.

**Proposed Work**

Consequently, researchers have created some classification skills using deep learning (DL) in many fields. For example, the DL algorithms, which have the machine learning architecture; for image classification, Convolutional Neural Networks (CNNs) applied most in the medical image classification processes. With transferring learning, most of the multiple models are initialized based on a pre-trained CNN mode, for example, AlexNet, VGG16, GoogleNet, and ResNet. The first network is trained by using all training sets for limited iterations to speed up training. The second stage is also in the training process, identifying the samples whose classification results have poor performance for further use to train the next model repeatedly. All procedures will be finished iteratively when they are all trained. Next, the results from the above stage are going to be tested to evaluate the performance score of different models. Finally, predict the samples by using the satisfied models with the high-performance score.

As a solution, they are using convolutional neural networks that have shown great success in diﬀerent kinds of pattern recognition tasks. They have also been recently used in classifying melanomas and other skin cancers from dermatoscopy and regular color images. They are using an eﬃcient strategy of convolutional neural networks which contains utilization of both spectral and spatial domain. With hyperspectral data containing wavebands from visible to infra-red region, we are able to gather more information from each pixel of the spectral images than using regular imaging systems. Using a sliding window method over captured spectral images, we will have spectral and spatial domain for further analysis.

In this study, we tested three diﬀerent kinds of feature learning structures - 1D, 2D, and 3D convolutions. We will also have basically two diﬀerent types of inputs. Single spectra and small window surrounding this spectra. A 1D convolution input takes a single spectrum. For 2D and 3D convolution, input will be a subset of the spectral cube. For the optimization we used Adam, which is a ﬁrstorder gradient-based optimization method of stochastic objective functions. There were only 61 imaged lesions (15 malignant melanoma, 6 lentigo maligna, 26 dysplastic nevus and 14 benign nevus). Thus, leave-one-out cross-validation was used. In this procedure, a classiﬁer is trained 61 times for each image separately. This will guarantee that the training set does not include data points from the image which is currently under classiﬁcation.

Four popular deep learning models, namely VGG16, VGG19, MobileNet, and InceptionV3 with atrous or dilated convolution instead of traditional convolution, had been used to build an accurate automated model for the dermatologist. To strengthen the accuracy, we utilized the transfer learning technique with a pre-trained ImageNet dataset for these dilated CNN networks. In Dilated VGG16 and Dilated VGG19 Architecture With dilated convolution without increased computational complexities, we can achieve a larger output feature map, which is proved to be appropriate for skin lesion classification in terms of accuracy. In dilated mobileNet architecture mobileNet has three kinds of convolutional operation: standard convolution, pointwise convolution, and depthwise convolution. In dilated inceptionV3 GoogleNet produces InceptionV3 architecture to perform efficiently under the strict limitations of memory and computation. InceptionV3 has three main parts in its networks, like factorizing convolution, auxiliary classifier, and coherent grid size reduction.

**Results and discussion**

Many preceding studies established that using pre-trained deep learning approaches to classify dermoscopic images is a promising domain. This work proposed a novel multiple convolution neural network model (MCNN) to classify the dermoscopic images. Transfer learning is used to train multiple models on different samples. The samples are selected according to their scores’ performance at the previous models. Compared to scalar measures, the receiver operating characteristic (ROC) curve can offer richer measure by calculating the AUC( the area under the ROC curve).

our approach gives us pixel-wise information. We ended situations where one lesion had several diﬀerently classiﬁed pixels. Malignant lesions can have non-malignant parts. If there was even a single pixel, which was classiﬁed as melanoma, the whole lesion was classiﬁed into melanoma. With all classiﬁers, we achieved the same positive predictive value (PPV) as clinicians. It is shown that the utilization of the spectral and spatial domain increases classiﬁcation performance. Our study’s ﬁrst limitation comes from the small data set. Even though we had over a hundred million pixels at our disposal, we eventually had only 61 diﬀerent lesions. This is a quite limited data set and more data is needed to develop and calculate a more robust and accurate neural network model. This would mean that we will need multi-center studies, where patient data is gathered in several countries simultaneously.

To construct our models, we mainly use Keras for the frontend development and TensorFlow for the backend. Next, Pandas and Scikit Learn are utilized respectively for data preprocessing, and to evaluate these proposed models. Training every model for 200 iterations and taking 32 as the mini-batch size. The models executed on Intel Core i7-8750H with 4.1 GHz and an NVIDIA GeForce GTX 1050Ti GPU. Adam 64 optimizer used as the optimization function with a learning rate 10-4 initially. One callback function utilized to lessen the learning rate factor by (0.1).5 during the training when the loss of validation is not diminishing for seven iterations.

**Conclusion**

In much research, multiple convolutional neural networks are proposed to classify seven types of skin lesions in dermoscopic images by using ISIC 2018 dataset to train and validate the proposed deep learning model. And the evaluated results are 0.99 and 0.72 for training and validation sets correspondingly.We have shown that the use of the spectral and spatial domains will increase the classiﬁcation performance of convolutional neural networks. Our results show that with a relatively small data set we are able to get the same or slightly better positive prediction values as clinicians.

We construct a computer-aided skin lesion classifier system using four different dilated deep neural network architectures (VGG16, VGG19, MobileNet, and InceptionV3) with transfer learning techniques. Several different data preprocessing and augmentation rules applied to lessen the effect of class imbalance characteristic of HAM10000. We tried several evaluation approaches such as top-1 accuracy, recall, precision, f-1 score, and confusion matrix to compare our proposed model with the basic one. These models produce better outcomes than any known methods on skin lesions classification after considering image noise presence, the number of classes, and the issue of class imbalance. Among all the proposed architectures, InceptionV3 delivered superior classification accuracy, and MobileNet exhibits fewer parameters.

# **The details of hardware and software**

Details of hardware:

Operating system name: Microsoft Windows 10 Home Chinese Edition

Version: 10.0.17134 Version 17134

Operating system manufacturer: Microsoft Corporation

System Manufacturer: ASUSTeK COMPUTER INC.

System type: based on × 64 computer

Processor: Intel (R) Core (TM) i5-8300H CPU @ 2.30GHz, 2304 Mhz, 4

RAM: 16.0 GB

Software version: RStudio Desktop 1.2.5042

**Description of chosen dataset:**

The dataset is taken from the ISIC (International Skin Image Collaboration) Archive. It consists of 1800 pictures of benign moles and 1497 pictures of malignant classified moles. The pictures have all been resized to low resolution RGB. The task of this kernel is to create a model, which can classify a mole visually into benign and malignant. We have followed following 9 steps for model building and evaluation which are as follows :

Step 1 : Importing Essential Libraries

Step 2: Loading pictures and making Dictionary of images and labels

Step 3: Categorical Labels

Step 4: Normalization

Step 5: Train and Test Split

Step 6: Model Building

Step 7: Cross-validating model

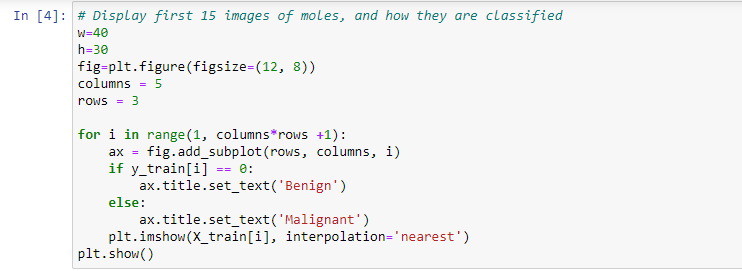
Step 8: Testing model

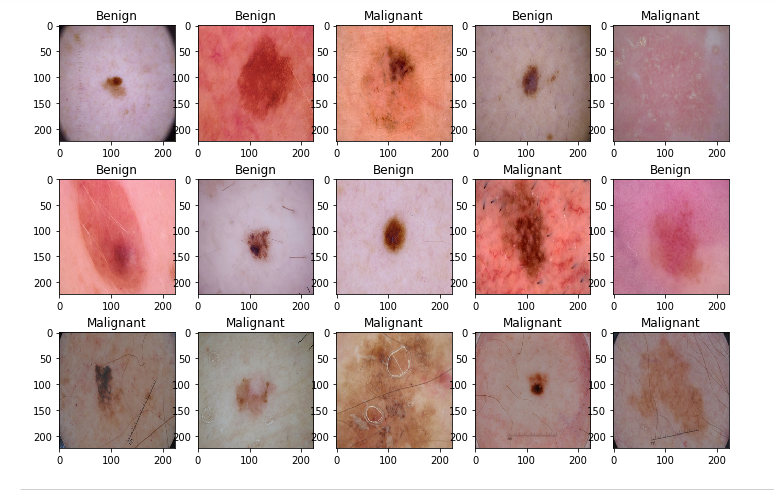
Step 9: ResNet50

**Data distribution of raw or original data**

# Loading pictures and making Dictionary of images and labels

In this step we load in the pictures and turn them into numpy arrays using their RGB values. As the pictures do not have any labels, these need to be created. Finally, the pictures are added together to a big training set and shuffled.





**Data preprocessing: Loading and resizing of images**

We resize the images as the original dimension of images are 450 x 600 x3 which TensorFlow can't handle, so that's why we resize it into 100 x 75. As this step resizes all the images dimensions into 100x 75.



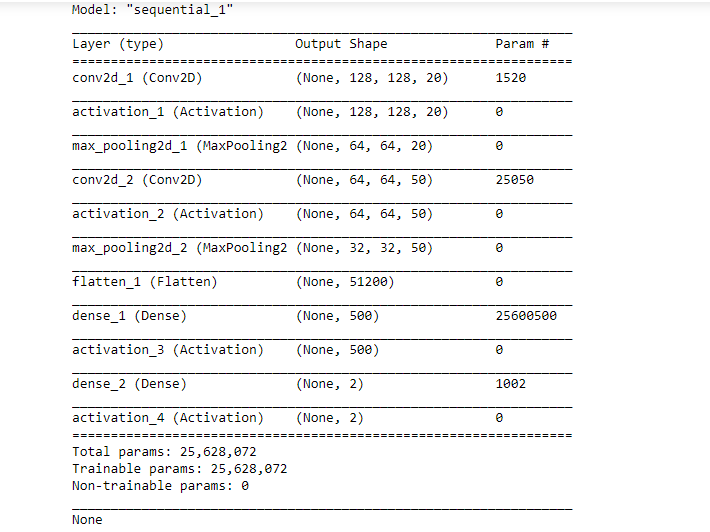
**Visualization of preprocessed dataset**

# **Model Building: CNN**

We used the Keras Sequential API, where we have just to add one layer at a time, starting from the input. The first is the convolutional (Conv2D) layer. We choose to set 32 filters for the two firsts conv2D layers and 64 filters for the two last ones. Each filter transforms a part of the image using the kernel filter. The kernel filter matrix is applied on the whole image. Filters can be seen as a transformation of the image.

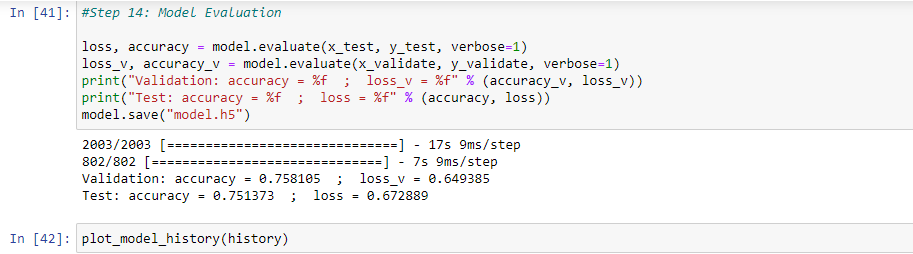
The second important layer in CNN is the pooling (MaxPool2D) layer. This layer simply acts as a downsampling filter. These are used to reduce computational cost, and to some extent also reduce overfitting. Dropout is a regularization method, where a proportion of nodes in the layer are randomly ignored for each training sample. This drops randomly a proportion of the network and forces the network to learn features in a distributed way. 'relu' is the rectifier. The rectifier activation function is used to add non linearity to the network.

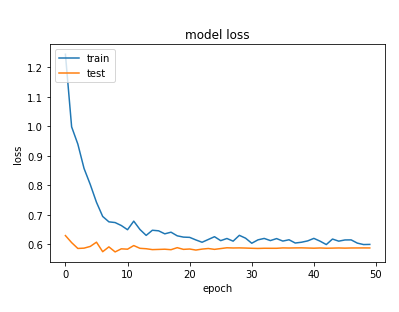
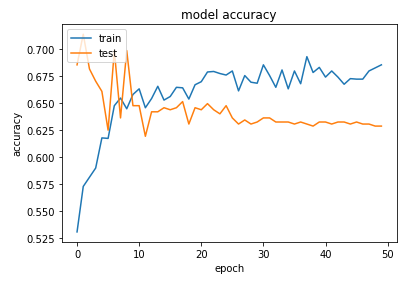
The Flatten layer is used to convert the final feature maps into a one single 1D vector. It combines all the found local features of the previous convolutional layers. In the end I used the features in two fully-connected (Dense) layers which is just an artificial neural networks (ANN) classifier. In the last layer the net outputs distribution of probability of each class.



**Results and Discussion: Model Evaluation**

In this step we checked the testing accuracy and validation accuracy of my model,plot confusion matrix and also checked the misclassified images count of each type.





It seems our model has a maximum number of incorrect predictions for Basal cell carcinoma which has code 3. In conclusion, We expect than we can also further tune the model to achieve the accuracy above 80% easily.

**Appendix**

**#Step 1 : Importing Essential Libraries**

import os

import sys

from tqdm import tqdm

import matplotlib.pyplot as plt

import numpy as np

import keras

from keras.preprocessing import image

from keras.preprocessing.image import ImageDataGenerator

from keras.layers import \*

from keras.models import Model

from keras.layers.convolutional import Conv2D

from keras.layers.convolutional import MaxPooling2D

from keras.layers.convolutional import UpSampling2D

from keras.layers.core import Activation

from keras.layers.core import Flatten

from keras.layers.core import Dense

from keras import backend as K

from keras.utils import np\_utils

from sklearn.model\_selection import train\_test\_split

from keras.preprocessing.image import img\_to\_array

from keras.layers.advanced\_activations import LeakyReLU

from keras.layers.advanced\_activations import ThresholdedReLU

from skimage.io import imsave, imread

from sklearn.metrics import confusion\_matrix, f1\_score, precision\_score, recall\_score

from skimage.transform import resize

from keras.optimizers import Adam

from keras.callbacks import Callback

from keras.callbacks import EarlyStopping

import tensorflow as tf

# Set some parameters

IMG\_WIDTH = 128

IMG\_HEIGHT = 128

IMG\_CHANNELS = 3

**#Step 2: Loading pictures and making Dictionary of images and labels**

#Step 2 : Loading pictures and making Dictionary of images and labels

folder\_benign\_train = 'D:/FDU/Data Mining and Data Warehouse/Group Project/skin-cancer-malignant-vs-benign/train/benign'

folder\_malignant\_train = 'D:/FDU/Data Mining and Data Warehouse/Group Project/skin-cancer-malignant-vs-benign/train/malignant'

folder\_benign\_test = 'D:/FDU/Data Mining and Data Warehouse/Group Project/skin-cancer-malignant-vs-benign/test/benign'

folder\_malignant\_test = 'D:/FDU/Data Mining and Data Warehouse/Group Project/skin-cancer-malignant-vs-benign/test/malignant'

read = lambda imname: np.asarray(Image.open(imname).convert("RGB"))

# Load in training pictures

ims\_benign = [read(os.path.join(folder\_benign\_train, filename)) for filename in os.listdir(folder\_benign\_train)]

X\_benign = np.array(ims\_benign, dtype='uint8')

ims\_malignant = [read(os.path.join(folder\_malignant\_train, filename)) for filename in os.listdir(folder\_malignant\_train)]

X\_malignant = np.array(ims\_malignant, dtype='uint8')

# Load in testing pictures

ims\_benign = [read(os.path.join(folder\_benign\_test, filename)) for filename in os.listdir(folder\_benign\_test)]

X\_benign\_test = np.array(ims\_benign, dtype='uint8')

ims\_malignant = [read(os.path.join(folder\_malignant\_test, filename)) for filename in os.listdir(folder\_malignant\_test)]

X\_malignant\_test = np.array(ims\_malignant, dtype='uint8')

# Create labels

y\_benign = np.zeros(X\_benign.shape[0])

y\_malignant = np.ones(X\_malignant.shape[0])

y\_benign\_test = np.zeros(X\_benign\_test.shape[0])

y\_malignant\_test = np.ones(X\_malignant\_test.shape[0])

# Merge data

X\_train = np.concatenate((X\_benign, X\_malignant), axis = 0)

y\_train = np.concatenate((y\_benign, y\_malignant), axis = 0)

X\_test = np.concatenate((X\_benign\_test, X\_malignant\_test), axis = 0)

y\_test = np.concatenate((y\_benign\_test, y\_malignant\_test), axis = 0)

# Shuffle data

s = np.arange(X\_train.shape[0])

np.random.shuffle(s)

X\_train = X\_train[s]

y\_train = y\_train[s]

s = np.arange(X\_test.shape[0])

np.random.shuffle(s)

X\_test = X\_test[s]

y\_test = y\_test[s]

**#Step 3: Categorical Labels**

y\_train = to\_categorical(y\_train, num\_classes= 2)

y\_test = to\_categorical(y\_test, num\_classes= 2)

**#Step 4 : Normalization**

# With data augmentation to prevent overfitting

X\_train = X\_train/255.

X\_test = X\_test/255.

**#Step 5: Model Building**

#CNN

# See learning curve and validation curve

def build(input\_shape= (224,224,3), lr = 1e-3, num\_classes= 2,

init= 'normal', activ= 'relu', optim= 'adam'):

model = Sequential()

model.add(Conv2D(64, kernel\_size=(3, 3),padding = 'Same',input\_shape=input\_shape,

activation= activ, kernel\_initializer='glorot\_uniform'))

model.add(MaxPool2D(pool\_size = (2, 2)))

model.add(Dropout(0.25))

model.add(Conv2D(64, kernel\_size=(3, 3),padding = 'Same',

activation =activ, kernel\_initializer = 'glorot\_uniform'))

model.add(MaxPool2D(pool\_size = (2, 2)))

model.add(Dropout(0.25))

model.add(Flatten())

model.add(Dense(128, activation='relu', kernel\_initializer=init))

model.add(Dense(num\_classes, activation='softmax'))

model.summary()

if optim == 'rmsprop':

optimizer = RMSprop(lr=lr)

else:

optimizer = Adam(lr=lr)

model.compile(optimizer = optimizer ,loss = "binary\_crossentropy", metrics=["accuracy"])

return model

# Set a learning rate annealer

learning\_rate\_reduction = ReduceLROnPlateau(monitor='val\_acc',

patience=5,

verbose=1,

factor=0.5,

min\_lr=1e-7)

**#Step 6: Model Evaluation**

loss, accuracy = model.evaluate(x\_test, y\_test, verbose=1)

loss\_v, accuracy\_v = model.evaluate(x\_validate, y\_validate, verbose=1)

print("Validation: accuracy = %f ; loss\_v = %f" % (accuracy\_v, loss\_v))

print("Test: accuracy = %f ; loss = %f" % (accuracy, loss))

model.save("model.h5")

**#Step 7: Testing the model**

# Fitting model to all data

model = build(lr=lr,

init= init,

activ= activ,

optim=optim,

input\_shape= input\_shape)

model.fit(X\_train, y\_train,

epochs=epochs, batch\_size= batch\_size, verbose=0,

callbacks=[learning\_rate\_reduction]

)

# Testing model on test data to evaluate

y\_pred = model.predict\_classes(X\_test)

print(accuracy\_score(np.argmax(y\_test, axis=1),y\_pred))

**#Step 8: ResNet50**

input\_shape = (224,224,3)

lr = 1e-5

epochs = 50

batch\_size = 64

model = ResNet50(include\_top=True,

weights= None,

input\_tensor=None,

input\_shape=input\_shape,

pooling='avg',

classes=2)

model.compile(optimizer = Adam(lr) ,

loss = "binary\_crossentropy",

metrics=["accuracy"])

history = model.fit(X\_train, y\_train, validation\_split=0.2,

epochs= epochs, batch\_size= batch\_size, verbose=2,

callbacks=[learning\_rate\_reduction]

)

# list all data in history

print(history.history.keys())

# summarize history for accuracy

plt.plot(history.history['acc'])

plt.plot(history.history['val\_acc'])

plt.title('model accuracy')

plt.ylabel('accuracy')

plt.xlabel('epoch')

plt.legend(['train', 'test'], loc='upper left')

plt.show()

# summarize history for loss

plt.plot(history.history['loss'])

plt.plot(history.history['val\_loss'])

plt.title('model loss')

plt.ylabel('loss')

plt.xlabel('epoch')

plt.legend(['train', 'test'], loc='upper left')

plt.show()

# **References**

[1] Guo, Yanhui, and Amira S. Ashour. 2019. “Neutrosophic Multiple Deep Convolutional Neural Network for Skin Dermoscopic Image Classification.” Neutrosophic Set in Medical Image Analysis, 269–285. doi:10.1016/b978-0-12-818148-5.00013-8.

[2] Noel Codella, Veronica Rotemberg, Philipp Tschandl, M. Emre Celebi, Stephen Dusza, David Gutman, Brian Helba, Aadi Kalloo, Konstantinos Liopyris, Michael Marchetti, Harald Kittler, Allan Halpern: “Skin Lesion Analysis Toward Melanoma Detection 2018: A Challenge Hosted by the International Skin Imaging Collaboration (ISIC)”, 2018; https://arxiv.org/abs/1902.03368

[3] Tschandl, P., Rosendahl, C. & Kittler, H. The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. Sci. Data 5, 180161 doi:10.1038/sdata.2018.161 (2018).

[4]Ratul, A. R., Mozaffari, M. H., Lee, W.-S., & Parimbelli, E. (2019). Skin Lesions Classification Using Deep Learning Based on Dilated Convolution. doi: 10.1101/860700

[5]Pölönen, Ilkka, et al. “Convolutional Neural Networks in Skin Cancer Detection Using Spatial and Spectral Domain.” Photonics in Dermatology and Plastic Surgery 2019, 2019, doi:10.1117/12.2509871.