Skin cancer classification computer system development with deep learning

Anastasia V. Demidova[a], Dmitry S. Kulyabov[a,b] and Eugeny Yu. Shchetinin[c]

[a]*Department of Applied Probability and Informatics, Peoples' Friendship University of Russia, 6, Miklukho–Maklaya St., Moscow, 117198, Russia*

[b]*Laboratory of Information Technologies, Joint Institute for Nuclear Research, 6, Joliot-Curie St., Dubna, Moscow region, 141980, Russia*

[c]*Department of Data Analysis, Decision Making and Financial Technologies, Financial University under the Government of the Russian Federation, 49, Leningradsky pr., Moscow, 111234, Russia*

Abstract

Melanoma is a deadly form of skin cancer that is often undiagnosed or misdiagnosed as a benign skin lesion. Its early detection is extremely important, since the life of patients with melanoma depends on accurate and early diagnosis of the disease. However, doctors often rely on personal experience and assess each patient's injuries based on a personal examination.

Clinical studies allow us to get the accuracy of the diagnosis of melatoma from 65 to 80 percents, which was a good result for some time. However, modern research claims that the use of dermoscopic images in diagnosis significantly increases the accuracy of diagnosis of skin lesions. The visual differences between melanoma and benign skin lesions can be very small, making diagnosis difficult even for an expert doctor. Recent advances in the use of artificial intelligence methods in the analysis of medical images have made it possible to consider the development of intelligent medical diagnostic systems based on visualization as a very promising direction that will help the doctor in making more effective decisions about the health of patients and making a diagnosis at an early stage and in adverse conditions.

In this paper, we propose an approach to solving the problem of classification of skin diseases, namely, melanoma at an early stage, based on deep learning. In particular, a solution to the problem of classification of a dermoscopic image containing either malignant or benign skin lesions is proposed. For this purpose, the deep neural network architecture was developed and applied to image processing. Computer experiments on the ISIC data set have shown that the proposed approach provides 92% accuracy on the test sample, which is significantly higher than other algorithms in this data set have shown.

Keywords

cancer, melanoma, deep learning, convolutional networks, image classification

Workshop on information technology and scientific computing in the framework of the X International Conference Information and Telecommunication Technologies and Mathematical Modeling of High-Tech Systems (ITTMM-2020), Moscow, Russian, April 13–17, 2020

demidova-av@rudn.ru (A. V. Demidova); kulyabov-ds@rudn.ru (D. S. Kulyabov); riviera-molto@mail.ru (E. Yu. Shchetinin)

https://yamadharma.github.io/ (D. S. Kulyabov)

0000-0003-1000-9650 (A. V. Demidova); 0000-0002-0877-7063 (D. S. Kulyabov); 0000-0003-3651-7629 (E. Yu. Shchetinin)

1. Introduction

Recently, skin cancer becomes one of the most common forms of cancer in the world [1, 2, 3, 4]. Squamous cell carcinoma, melanoma, intraepithelial carcinoma, basal cell carcinoma are just some of the common types of skin lesions, but among them, melanoma is the most dangerous and extremely malignant. Early diagnosis of melanoma can cure almost 95% of cases [5], and with the help of dermatoscopy, the accuracy of treatment of skin lesions will be 60-80%. A manual skin lesion detection system is labor-intensive for a person who needs to zoom in and illuminate the skin images to improve the clarity of the spots. The ABCD rule (asymmetry, border, unevenness, color and diameter changes), 3-point checklist, 7-point checklist, and Menzies method are several procedures for improving the effectiveness of dermatoscopy and monitoring malignant melanoma at the earliest stage, but many clinicians constantly rely on their experience. Manual dermatoscopy is more prone to errors because it requires many years of experience in complex situations, a huge amount of visual research, similarities and differences between different skin lesions. Modern research claims that the use of stereoscopic images in diagnosis significantly increases the accuracy of diagnosis of skin lesions. But the visual differences between melanoma and benign skin lesions can be very small, making diagnosis difficult even for an expert doctor.

Recent advances in the use of artificial intelligence methods in the analysis of medical images have made it possible to consider the development of intelligent medical diagnostic systems based on visualization as a very promising direction that will help the doctor in making more effective decisions about the health of patients and making a diagnosis at an early stage and in adverse conditions. Deep learning algorithms provide computer systems for detecting, classifying, and diagnosing diseases using medical image analysis [6]. In the past few years, dermatoscopy has produced a significant number of well-annotated images of skin lesions that help machine learning methods actively classify, predict, and detect various skin lesions. Based on deep learning, medical image analysis tools can be useful to help the dermatologist focus on several areas, such as skin lesions segmentation, classification and detection.

In this paper, we propose an approach to solving the problem of classification of skin diseases, namely, melanoma at an early stage, based on deep learning. In particular, a solution to the problem of classification of a dermoscopic image containing either malignant or benign skin lesions is proposed. For this purpose, the deep neural network architecture was developed and applied to image processing. Computer experiments on the ISIC data set have shown that the proposed approach provides 93% accuracy on the test sample, which is significantly higher than other algorithms in this data set have shown.

2. Modern achievements in the field of computer processing of dermatoscopic images

Most of the classic medical methods in the field of melanoma classification rely on manual selection of signs, such as: type of lesion (primary morphology), configuration of the lesion (secondary morphology), color, distribution, shape, texture, and border irregularity. After extracting the main characteristics, machine learning methods are used, such as the k-nearest

neighbor algorithm (kNN), logistic regression, and it is also possible to use decision trees and support vector machines (SVM). Modern computer research in the diagnosis of skin diseases for the purpose of detecting cancer using deep learning methods is aimed at improving existing and developing new models of deep neural networks, primarily convolutional neural networks (CNN).

Nasr-Esfahani et al. proposed a CCN architecture for the diagnosis of melanoma, where clinical images were pre-processed in such a way as to reduce the illumination of the image. Research results have shown that the proposed method is able to diagnose cases of melanoma in 70% of cases [7]. Giotis et al. [8] introduced the MEDNODE expert system to help doctors detect melanoma. The proposed system used extracted lesion areas in the image, then calculated features such as color and texture, as well as visual attributes provided by experts. In the work of Mahbod et al. it is shown that convolutional neural networks are superior to traditional methods of machine learning [9]. The authors proposed a hybrid automatic computerized method for classifying skin diseases using three pre-trained deep networks (AlexNet, VGG16, ResNet-18) to extract signs. The features extracted in this way are then used to train the support vector machine on 150 images from the ISIC 2017 dataset. Jaisakthi, et al. suggested a method of segmenting lesions in images and their classification types of skin cancer [10]. The proposed method consists of preprocessing and segmentation using a hybrid learning algorithm. The goal of the first stage is to remove noise using the filtering method. In the second stage, images are segmented based on the clustering method. In the work of Codella et al. the authors report a new state of art performance using CNN to extract image characteristics using a pretrained model from data presented at (ILSVRC) 2012 [11].

3. Dermatoscopic data description and their preprocessing

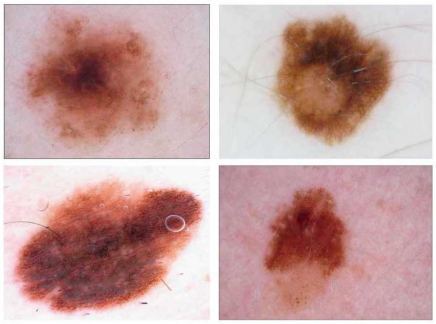
In dermatology, there are a few data sets with digital images of skin lesions. Most of these sets are too small or not publicly available, which creates an additional obstacle to reproducible research in this field. Examples of dermatology-related image datasets used in recent studies include: the Dermofit image Library [12] is a dataset containing 1.300 high-quality images of skin lesions collected in 10 different classes. Dermnet — the website-enabled skin diseases Atlas contains more than 23.000 skin images divided into 23 classes [13]. In 2016 the international Symposium on Biomedical Imaging [14] released a new set of data for analyzing skin lesions in order to detect melanoma. The photos in this data set were obtained from ISIC (international Skin Imaging Community) [15].

3.1. Dataset

In order to support the training of clinical dermatologists and the development of new information technologies, the International Society for Skin Imaging (ISIC) has developed an international repository of dermoscopic images, known as the ISIC archive [15]. Every year ISIC increases its archive and promotes the task of implementing computer methods for detecting melanoma and other skin diseases. In 2019 the number of samples totaled 25.331 images for Dermoscopy, available for training in 8 different categories.

The ISIC data set used in this work is publicly available and contains 1.279 images, pre-divided into 900 training images and 379 test images. Due to the unbalanced nature of the data set, the test and training data sets were reduced by reducing the sample to obtain a balanced ratio of the number of images for each class. Some examples from ISIC archive are presented in the Table 1.

Table 1
Images examples from ISIC archive

Melanoma	Benign
	

3.2. Images preprocessing

Data preprocessing is one of the most important stages in machine learning. It's idea is to use a systematic approach to the preparation of the data before feeding them into a machine learning model. Images of the skin surface, even those made by a professional digital camera, usually contain light and noise effects that need to be eliminated. These effects are the result of inhomogeneous lighting and the reflection of incident light from the skin surface. To reduce the impact of these factors on network training and classification, first, the correction step is performed on the input images. Thus, lighting effects are discarded by excluding a certain range of gradients. Note that this is done without destroying the edge of the original image. Another factor to consider is that the image contains both healthy (normal skin) and affected skin areas. However, cropping healthy areas can lead to loss of information, such as the difference in color between the affected and normal skin of the patient. For this purpose, a segmentation mask is used by applying the k-means classifier to the pre-processed image.

Input images are preprocessed before they are sent to the neural network in the next steps: 1) normalizing pixel values from $[0, 255]$ to the range $[0, 1]$; 3) resizing the original image to 224 pixels.

Segmentation is usually a necessary preprocessing approach when you need to analyze and detect objects and object boundaries in a digital image. In operation, data set images are in RGB format (red, green, blue), and since RGB images are more related to the amount of light and illumination, this makes it difficult to extract image features and borders.

In the article, data set images are in RGB format (red, green, blue) and since RGB images are more related to the amount of light and illumination, this makes it difficult to extract image features and borders. Thus, all images were converted to HSV color space (hue, saturation,

value) which are more useful and relevant for detecting objects in digital images. The second stage of preprocessing the dataset is to apply a two-way filter to all images to preserve their sharp edges. A two-way filter replaces the intensity of each pixel with a weighted average of the intensity of neighboring pixels. The third step is to convert images to halftone images to reduce the complexity and dimension of images, and automatically detect the edges of objects in images using the Canny border selection algorithm.

3.3. Data augmentation

If there is a large class imbalance in the data, rebalancing methods, such as minority oversampling, are usually used for stable operation of classification algorithms. Deep learning models require a large amount of data. Since that our training set contains only 2000 images, so we have increased their number by rotating them, flipping, randomly cropping, adjust brightness, adjusting the contrast, pixel jitter, changing the aspect ratio of the image, random shifting, zooming, and vertical and horizontal shifting. This makes the data set for training less unbalanced and improves the functioning of the neural network [16, 17].

4. Basic architectures of deep neural networks in the analysis of dermatoscopic images

A neural network is a model that displays input data to a specific goal in a self-learning form. This is achieved through the network architecture. Neural networks consist of different layers that are sequentially superimposed on the input data. Each layer consists of several "neurons". Each neuron calculates the weighted sum of the output data from the previous layer, and then applies a nonlinear transformation. These weights are what is learned during network training. Non-linearities can lead to various effects, for example: scaling the output to a significant value is only possible when the sum exceeds a certain threshold (Sigmoid), or when the sums cannot become negative (Relu). The exact choice is often just a detail of the implementation, but their existence is essential. A convolutional neural network (CNN) is a deep learning algorithm that takes an input image, assigns the studied weights and offsets to various functions and objects in the image, and is then able to distinguish one from the other. There are 4 basic steps to create a CNN:

Step 1: Convolutional Layer: Creates an object map that contains arguments written by the filter for the input image. The value from the input image is initially inserted into the upper-left cell of the object map and moves the block to the right, recording the observation of each step.

Step 2: Pooling Layer: the pooling Layer is a layer that uses functions with a feature map as input and processes them using various statistical operations. A join layer in a CNN model is usually inserted after several convolution layers. The merge layer in the CN model architecture allows you to gradually reduce the size of the output volume in the feature map in order to reduce the number of parameters and calculations used in the network, and to control reconfiguration. Most of CNN is usually used max pooling (Max pooling). It divides the output of the convolution layer into several small grids, and then takes the maximum value from each grid to produce a reduced image matrix.

Step 3: Flatten Layer: the merged feature map is used as input for this layer, and the Flatten layer converts it to a column.

Step 4: Fully Connected Layer.

Step 5: Compiling the network model (Compiling the CNN).

Step 6: Fitting the CNN. Building estimates using the compiled architecture.

Step 7: Evaluating accuracy metrics, building the confusion matrix. Four evaluation measures are used to estimate the performance of proposed model. These measures are accuracy, sensitivity, specificity, and precision. They are computed using the following equations:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}, \tag{1}$$

$$sensitivity = \frac{TP}{TP + FN}, \tag{2}$$

$$specifity = \frac{TN}{FP + TN}, \tag{3}$$

$$Precision = \frac{TP}{TP + FP}, \tag{4}$$

where TP, FP, FN, and TN are true positive, false positive, false negative, and true negative respectively.

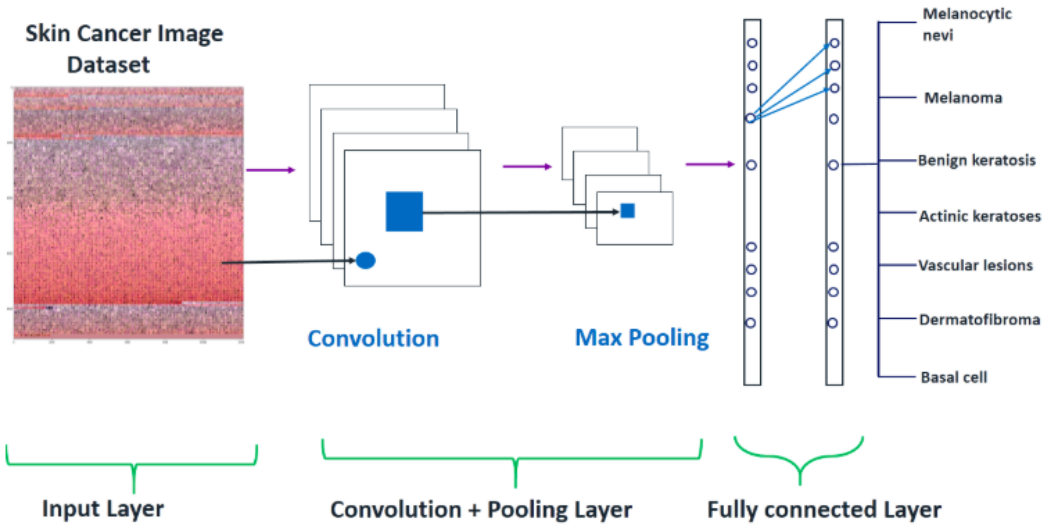


Figure 1: Implementation of the algorithm for building the CNN architecture Step 1 – Step 7

5. Model building and computer experiments

The CNN model is initialized as a sequence of layers using the Sequential class. Next, the conv2d convolutional layer is added, with the input parameters of the feature map $inputshape =$

(32, 32, 3), where 32 is the size of the spatial features of the input map, 3 is the number of color channels (in this case, the color of the image in RGB format). The convolution is determined by the following parameters: the size of the templates extracted from the input data — (3×3); the depth of the output feature map – the number of filters calculated by the convolution. In this model, the first convolutional layer outputs a feature map of size (30, 30, 32) and calculates 32 filters from the input data.

Each of these 32 output channels contains a grid of (30×30) values — a map of filter responses to input data that defines the response of this filter template for different parts of the input data. The last parameter is the activation function that we use to activate neurons in the neural network, such as 'relu'. The third step uses the pooling layer (MaxPooling2D) with the map (3×3). The main purpose of using this layer is to reduce the number of coefficients in the feature map for processing, and to implement hierarchies of spatial filters by creating successive convolution layers for viewing larger windows. Then create a vector for the fully connected layer (Flatten ()). The Flatten layer serves as a link between the data received by the algorithm and the output vector, converting the multi-dimensional output of the previous layer into a one-dimensional one. The last step builds a fully connected layer-the Dense layer. The Dense function has 2 parameters – the number of nodes for the output layer(128) and the activation function, which will again be 'relu'. The output layer has 1 node where the 'sigmoid' activation function is used. Next, we need to compile the model and optimize the weight coefficients and the loss function to evaluate the model. The final graph of our model is demonstrated on Figure 2.

The question that remains is how each weight should be changed to improve the performance of our model. This is taken care of by the optimizer, which seeks to find the minimum for our loss function. Training of a deep neural network is a process of iterative refinement of its parameters (weights of neurons) to increase its productivity. This is done using the loss function, which iteratively evaluates the predicted values and compares them with the true values, and is used to update the weights according to the calculated error.

Next we will choose the loss function for our network. Since we have two classes (1 or 0; Benign or Malignant tumors), in our choice the loss function is binary crossentropy. The overfitted model will make random predictions, and so the loss function will generate the large values. As the model improves and the accuracy of its forecasts increases, the amount of losses approaches zero. There are many different methods for minimizing the loss function, which in most cases are based on the gradient descent method. In this article, we chose Adam [18] as the optimizer for our model, since it is one of the most frequently used and effective optimizers. An important setting of the optimizer is the correct choice of learning rate. If the learning rate is selected too low , the network parameters will be changed only very slightly, and the search for the minimum will take a very long time. On the other hand, if we choose a very high learning rate, it can lead to the optimizer changing the parameters too much (exceeding), and we will never be able to find the minimum at all. So, we choose a learning rate of 1e-03. An accuracy metric that evaluates the model's accuracy, such as accuracy, which is achieved by dividing true forecasts by general forecasts [19].

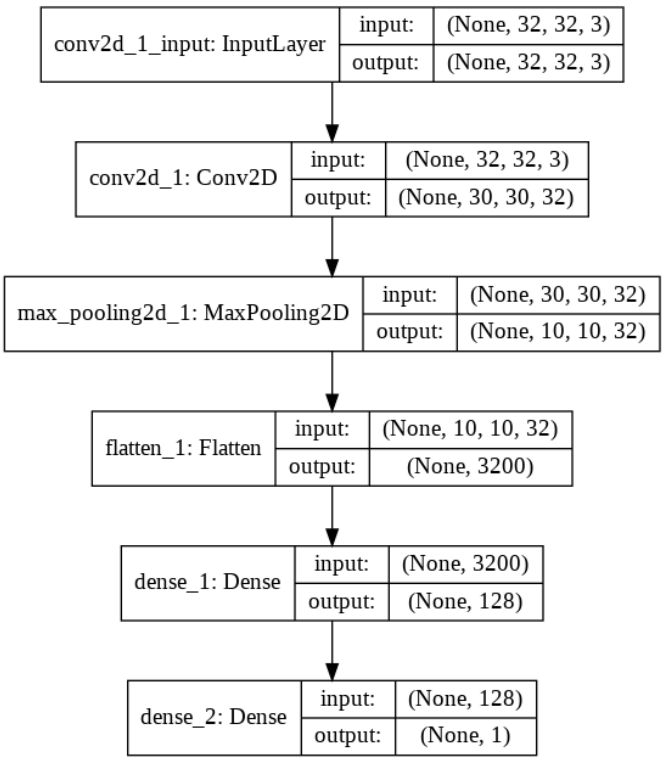


Figure 2: CNN-model

Table 2
Confusion Matrix for proposed DCNN

Matrix		Actual class	
		Melanoma	Non-Melanoma
Prediction Class	Melanoma	96	6
	Non-Melanoma	9	89

6. Results and Discussion

The paper investigates a publicly available set of ISIC data containing 1255 images, as well as 600 test images and 200 validation images selected from them. Due to an unbalanced data set, the test and training data sets were reduced by reducing the sample to produce the same number of images for each class. The final training set contains 400 images, and the final test set contains 200 images. All images are placed as benign and malignant and include a binary image mask to indicate damage within it. Creating a balanced data set involves the removal of the images from the training and test data of images. In the training set, images were selectively removed from the data set and 200 images of each class were obtained, for a total of 400 images.

The model was evaluated using accuracy and sensitivity and loss function. Accuracy is defined as the number of correct predictions divided by the total number of predictions made.

Sensitivity is defined as the number of true positie divided by the sum of true and false positive and measures the percentage of positive results that are correctly identified. See formulas 1–4. From Table 2 we could calculate the *accuracy* = 0.925. Losses are defined as the balance between the quantitative estimates for predicted images and the true values of their labels. As an optimization function the class ADAM and binary cross entropy as a loss function were used respectively. To make sure that the model during the calculation process produces the same results after each epoch, the random number generator was selected with an arbitrary value that would remain constant throughout the training.

Currently, Tensorflow [20] and Keras [21] libraries are very popular among developers. To build our model, we use Keras and Tensorflow for the backend [22]. Further, Pandas and Scikit Learn are used, respectively, for pre-processing data, as well as for evaluating the proposed model. Trainig model in 25 iterations and take 10 as a mini-batch size. Some fragments of program code are placed in paragraph 8. Computational procedures for training and fine-tuning the network are highly expensive processes that require significant time and memory to store data and calculation results. Therefore, it is necessary to have both effective computer tools and software tools. All experiments were performed using an IBM computer equipped with Intel core i5 processor, 8 GB SDRAM, and an NVIDIA GeForce 820M graphics card.

7. Conclusions

The paper implements a computational method based on deep learning algorithms (convolutional neural network) that uses a set of images provided by the ISIC (International Skin Imaging Collaboration). The proposed method involves pre-processing images to extract the area of the assumed location of the disease in the image itself, and then enlarging some images to obtain a larger data set. The resulting data set was used in the CNN model to train a model that consists of multiple layers, such as convolution layers, pooling layers, and fully connected layers. The accuracy of the model on the test sample is 92%. This result motivates us to conduct the research for online diagnosis of melanoma at early stages.

Our further research will focus on the development of the CNN architecture to improve accuracy, getting more image data for training, applying new algorhythms to train the model using more data. Ultimately the future plan is to make this model available and usable as a mobile application.

References

[1] Skincancer.org, ”Melanoma — SkinCancer.org”, 2016. URL: http://www.skincancer.org/skin-cancer-information/.

[2] H. W. Rogers, M. A. Weinstock, S. R. Feldman, B. M. Coldiron, Incidence Estimate of Nonmelanoma Skin Cancer (Keratinocyte Carcinomas) in the US Population, 2012, JAMA Dermatology 151 (2015) 1081–1086. URL: https://doi.org/10.1001/jamadermatol.2015.1187. doi:10.1001/jamadermatol.2015.1187.

[3] R. L. Siegel, K. D. Miller, A. Jemal, Cancer statistics, 2019, CA: A Cancer Journal for

Clinicians 69 (2019) 7–34. URL: https://acsjournals.onlinelibrary.wiley.com/doi/abs/10.3322/caac.21551. doi:10.3322/caac.21551.

[4] American Cancer Society, Cancer Facts and Figures, 2019. URL: https://www.cancer.org/research/cancer-facts-statistics/all-cancer-facts-figures.html.

[5] N. Eisemann, A. Waldmann, A. C. Geller, M. A. Weinstock, B. Volkmer, R. Greinert, Non-melanoma skin cancer incidence and impact of skin cancer screening on incidence, Journal of Investigative Dermatology 134 (2014) 43–0. doi:10.1038/jid.2013.304.

[6] S. Zhou, H. Greenspan, D. Shen, Deep Learning for Medical Image Analysis, Elsevier Inc., 2017.

[7] E. Nasr-Esfahani, S. Samavi, N. Karimi, S. Soroushmehr, M. H.-M. Jafari, K. Ward, K. Najarian, Melanoma detection by analysis of clinical images using convolutional neural network, in: 2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), volume 2016, 2016, pp. 1373–1376. doi:10.1109/EMBC.2016.7590963.

[8] I. Giotis, N. Molders, S. Land, M. Biehl, M. Jonkman, N. Petkov, MED-NODE: A Computer-Assisted Melanoma Diagnosis System using Non-Dermoscopic Images, Expert Systems with Applications 42 (2015) 6578–6585. doi:10.1016/j.eswa.2015.04.034.

[9] A. Mahbod, G. Schaefer, C. Wang, R. Ecker, I. Ellinge, Skin lesion classification using hybrid deep neural networks, ICASSP 2019 — 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (2019) 1229–1233. doi:10.1109/icassp.2019.8683352.

[10] S. M. Jaisakthi, C. Aravindan, M. Palaniappan, Automatic skin lesion segmentation using semi-supervised learning technique, 2017. URL: https://arxiv.org/abs/1703.04301.

[11] N. Codella, Q.-B. Nguyen, S. Pankanti, D. A. Gutman, B. Helba, A. C. Halpern, J. R. Smith, Deep learning ensembles for melanoma recognition in dermoscopy images, IBM Journal of Research and Development 61 (2017) 5:1–5:15. doi:10.1147/jrd.2017.2708299.

[12] Dermofit image library, 2020. URL: https://licensing.eri.ed.ac.uk/i/software/dermofit-image-library.html.

[13] Dermnet – skin disease atlas, 2020. URL: http://www.dermnet.com/.

[14] IEEE International Symposium on Biomedical Imaging, 2020. URL: http://biomedicalimaging.org/.

[15] International Skin Imaging Collaboration: Melanoma Project Website, 2020. URL: https://isic-archive.com/.

[16] A. Esteva, B. Kuprel, R. Novoa, J. Ko, S. Swetter, H. Blau, S. Thrun, Dermatologist-level classification of skin cancer with deep neural networks, Nature 542 (2017) 115–118. doi:10.1038/nature21056.

[17] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, Mobilenets: Efficient convolutional neural networks for mobile vision applications, 2017. URL: http://arxiv.org/abs/1704.04861.

[18] Adam optimizer, 2020. URL: https://keras.io/optimizers/adam/.

[19] L. A. Sevastianov, E. Y. Shchetinin, On methods for improving the accuracy of multiclass classification on imbalanced data, Informatics and Applications 14 (2020) 63–70. doi:10.14357/19922264200109.

[20] Tensorflow, 2020. URL: https://www.tensorflow.org/.

[21] Keras, 2020. URL: https://keras.org/.

[22] M. N. Gevorkyan, A. V. Demidova, T. S. Demidova, A. A. Sobolev, Review and comparative analysis of machine learning libraries for machine learning, Discrete and Continuous Models and Applied Computational Science 27 (2019) 305–315. doi:10.22363/2658-4670-2019-27-4-305-315.

A. Program Code

Deep CNN model

```
%Importing Keras packages
from keras.models import Sequential
from keras.layers import Convolution2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
from keras.layers import Dense

%initializing the CNN
classifier = Sequential()

%Adding the Convolution Layer
classifier.add(Convolution2D(32, 3, 3,
input_shape = (32, 32, 3), activation = ”relu”))

%Adding the Pooling Layer
classifier.add(MaxPooling2D(pool_size=(3, 3)))

%Flattening the layer
classifier.add(Flatten())
% Full Connected layer
classifier.add(Dense(output_dim = 128, activation = ’relu’))
classifier.add(Dense(output_dim = 1, activation = ’sigmoid’))

#Compiling the CNN
classifier.compile(optimizer = ’adam’, loss = ’binary_crossentropy’,
metrics = [’accuracy’])

#Fitting our CNN to the image dataset
from keras.preprocessing.image import ImageDataGenerator

#Model summary
classifier.summary()
from keras.utils import plot_model
plot_model(classifier,show_shapes=True, show_layer_names = True,
```

```
to_file='netork.png')

train_datagen = ImageDataGenerator(
rescale=1./255,
shear_range=0.2,
zoom_range=0.2,
horizontal_flip=True)

test_datagen = ImageDataGenerator(rescale=1./255)

training_set = train_datagen.flow_from_directory(train_data_dir,
target_size = (img_width, img_height),
batch_size=10,
class_mode='binary')
test_set = test_datagen.flow_from_directory(validation_data_dir,
target_size=(img_width, img_height),
batch_size=32,
class_mode='binary')
history=classifier.fit_generator(training_set,
steps_per_epoch=1255,
epochs=25,
validation_data= test_set,
validation_steps=400)

# Get training and test loss histories
import matplotlib.pyplot as plt
import numpy as np
# Set random seed
np.random.seed(0)

# Get training and test loss histories
training_loss = history.history['loss']
test_loss = history.history['val_loss']

import matplotlib.pyplot as plt
import numpy as np
# Set random seed
np.random.seed(0)

training_loss = history.history['loss']
test_loss = history.history['val_loss']
# Loss Curves
plt.figure(figsize=[8,6])
plt.plot(history.history['loss'],'r',linewidth=2.0)
```

```
plt.plot(history.history['val_loss'],'b',linewidth=2.0)
plt.legend(['Training loss', 'Validation Loss'],fontsize=18)
plt.xlabel('Epochs ',fontsize=16)
plt.ylabel('Loss',fontsize=16)
plt.title('Loss Curves',fontsize=16)

# Accuracy Curves
plt.figure(figsize=[8,6])
plt.plot(history.history['acc'],'r',linewidth=2.0)
plt.plot(history.history['val_acc'],'b',linewidth=2.0)
plt.legend(['Training Accuracy', 'Validation Accuracy'],fontsize=18)
plt.xlabel('Epochs ',fontsize=16)
plt.ylabel('Accuracy',fontsize=16)
plt.title('Accuracy Curves',fontsize=16)
```