Design and implementation of a classification model dealing with imbalanced data for melanoma detention

Luis Castañeda[1,✉], **Rodrigo López**[1,✉], **and Mateo Fernandez**[1,✉]

[1]Illinois Institute of Technology, Private university in Chicago, Illinois

Abstract

Overall, people who are suffering from an illness, whether serious or light, will have a better chance of overcoming it the earlier they are diagnosed. In the case of cancer, this constant is even greater, it is a disease with high morbidity and any help that leads to early diagnosis is welcome and of vital importance. In these times, the weight of computers and artificial intelligence algorithms is growing every day, so they must be used in a way that can help the health of any person.

In the medical field, a professional can diagnose a melanoma to a patient by observing a spot, mole or mark and comparing it with other spots on the patient's skin through medical knowledge. In the field of machine learning, a machine through an algorithm may be able to predict a melanoma by learning the patterns of an image in which a melanoma can be seen.

This research focuses on the early detection of melanoma through a classification algorithm based on convolutional neural networks. Both medical professionals and machines may encounter a common problem in melanoma diagnosis. The problem is that there are many spots or moles on the skin that may be possible melanomas. In the field of data, this is known as imbalanced data and is one of the challenges facing this research.

The model created in this research based on convolutional neural networks aims to learn those details of melanomas in order to make a better prediction of them. For this purpose, several hidden layers are included to add complexity to the model. The results show that the accuracy of the melanoma class has been prioritized, since it is considered better to diagnose a melanoma when it does not really exist, than not to diagnose a melanoma when it does exist.

cancer | melanoma | imbalanced | CNN | SMOTE | oversampling

Correspondence: ***lcastanedalopez@hawk.iit.edu***
rlopezpuente@hawk.iit.edu
mfernandezlopezareal@hawk.iit.edu

Introduction

Cancer is one of the major causes of mortality in the world. In the United States, 2021 statistics report that the number of estimated new cases of cancer is 1.8 million and the number of estimated deaths due to cancer is 606,570 [1]. Specifically, one out of three cases of cancer diagnosed is skin cancer and in United States there are over 3.5 million cases of skin cancer diagnosed each year in more than two million people and there are over 9,500 deaths each year [2]. Among the different skin cancers, one of the most dangerous is melanoma. As it is shown in the Figure 1, one of the most common cancer type is melanoma and in 2021, it was estimated 106,110 new cases of melanoma and 7,180 of these cases would result in death [1].

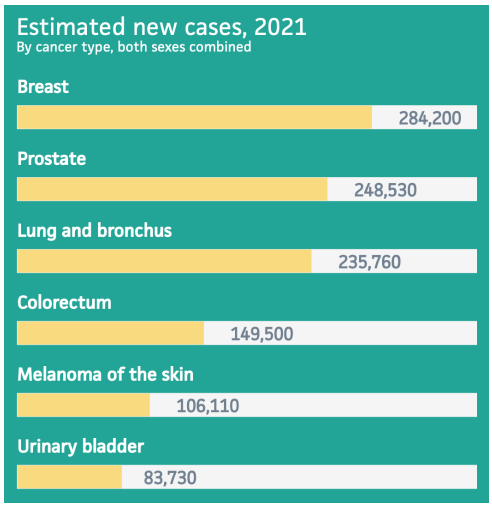
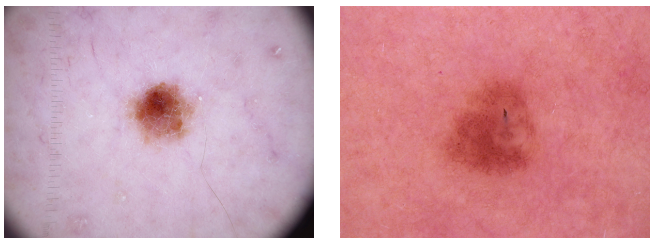


Fig. 1. Estimated new cases by cancer type.

Melanoma is a type of skin cancer that appears when melanocytes, the cells that give the skin its tan, grow out of control [3]. Melanoma is dangerous because of its ability to spread to other organs quickly if not detected and treated early. As soon as it is detected, there is a higher probability that a person can be cured [4]. For this it is necessary to periodically check for new spots or moles that may appear on the skin. Moles, sores, bumps, spots or unusual marks on the skin can be a sign of melanoma. There is a guide to the common signs of melanoma, the ABCDE rule [5]:

- Asymmetry: one half of the spot does not match the other.
- Border: the borders are irregular or blurred.
- Color: the color is not the same as the other spots.
- Diameter: the spot is larger than 6 millimeters in diameter, however there are smaller melanomas.
- Evolution: the spot is changeable.

In the next Figure you can see an example of two different spots, image (a) is a melanoma and image (b) is not.



(a) (b)

Fig. 2. Example of different spots.

Problem description

Obviously, this research is about how to detect the existence of melanoma at an early stage. To do this, it has been intended to perform a similar approach to what any doctor would do when faced with a new and strange spot that has appeared on the skin, to analyze it. A medical professional may be more knowledgeable about melanoma but, as mentioned above, the most important thing is to analyze the spot or mole in comparison with other existing marks on the skin. At the time of this analysis, the HAM10000 dataset has been used, which has 10000 dermoscopic images that will be used to classify an input image into melanoma or non-melanoma. These images were collected from different populations, acquired and stored by different modalities. The dataset is divided in all important diagnostic categories in the realm of pigmented lesions: melanoma, vascular lesions, basal cell carcinoma, dermatofibroma, melanocytic nevi, benign keratosis lesions and Bowen´s disease. More than half of the diagnosis are confirmed by the use of histopathology, the diagnosis and study of diseases of the tissues that involves examining tissues and cells under a microscope. The rest of the lesions' diagnosis was based on a consensus of dermatologists. However, the project is a binary classification model, needing to modify this data into two classes, with melanoma and without melanoma. Which leaves an imbalanced dataset where 88.89% belong to the group of non-melanomas and which leads to one of the main difficulties faced by this research, which was the treatment of a highly imbalanced data set.
Therefore, this research aims to balance a dataset representing different medical images with or without melanoma to create a classification model that is able to label or not an image as melanoma in order to achieve the main purpose of early detection of skin cancer.

SMOTE

One of the major problems that this project has to face is a balanced dataset. Unbalanced datasets are a big problem for the training of Machine Learning algorithms. In order to train a Machine Learning algorithm in the best possible way, the training dataset must be balanced. This is necessary to prevent the algorithm from obtaining a clear bias towards the majority class, otherwise the algorithm will know how to identify the majority class too well and the minority class too weakly. Therefore, techniques must be used to create a balanced dataset before training the Machine Learning algorithm.

Therefore, in this project it has been decided to use the SMOTE algorithm for the creation of synthetic samples of the minority class, in order to create and have a new balanced dataset as input to the algorithm to be trained.

Synthetic Minority Oversampling Technique (SMOTE) is an oversampling technique that uses the feature space to generate new synthetic samples for the minority class [6]. This algorithm helps to overcome the fitting problem generated by random oversampling. It focuses on the feature space to generate new instances with the help of interpolation between positive instances found together.

This algorithm works by choosing a random observation from the minority class and its k nearest neighbors. Then, one of these neighbors is chosen and the new instance will be between the random observation and its chosen neighbor [7].

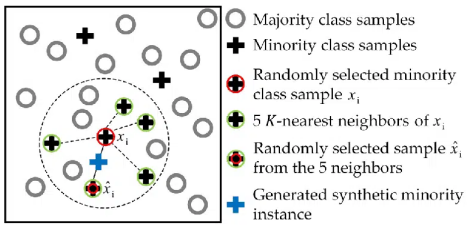


Fig. 3. SMOTE algorithm performance [7].

Being s the new instance, and x the randomly chosen sample, xR a neighbor of this sample and $0 \leq u \leq 1$, it follows that [8]:

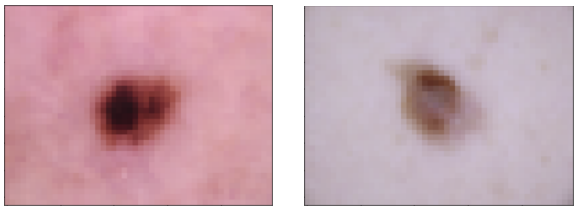
$$s = x + u \cdot (x^R - x) \tag{1}$$

While this algorithm is quite useful, it has some drawbacks of its own.

i) The synthetic instances generated with this algorithm are in the same direction, in other words, connected by an artificial line their diagonal instances. This, in turn, complicates the decision surface generated by some classifier algorithms.

ii) SMOTE tends to create a large number of noisy data points in the feature space.

As previously mentioned, one of the great challenges of this project is not just balancing the dataset, but balancing a dataset of images. Perhaps at the outset there were some doubts as to whether the SMOTE algorithm would be able to create synthetic images for better training of the algorithm. Below is an example of one of the images obtained synthetically using the SMOTE algorithm.



Original Image *Synthetical Image*

Fig. 4. Melanoma Images

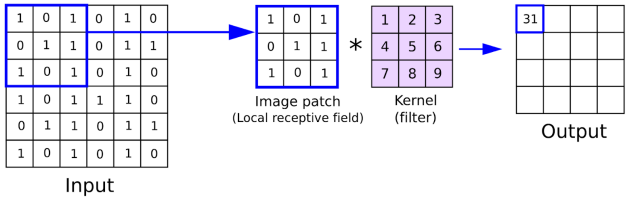


Fig. 5. Process of Convolutional Layers

In the images above you can see two photos, the photo on the left is one of the photos from the original dataset, and the photo on the right is one of the images created synthetically with the SMOTE algorithm.

Convolutional Neural Network

The name Convolutional Neural Network, or CNN, indicates that the network employs a mathematical operation called convolution. Convolutional networks are a specialized type of neural networks that use convolution in place of general matrix multiplication in at least one of their layers. CNN is able to successfully capture the spatial dependencies in an image (data grid) through the application of relevant filters. CNNs can reduce images into a form which is easier to process without losing features that are critical for getting a good prediction. CNN are very effective in reducing the number of parameters without losing on the quality of models, and as images have high dimensionality it it the prefect fit.

CNN itself carry out the process of extracting and describing the features of the data, in order to leave the classification of this feature to another neural network. For this, it can be said that the CNN are divided into two big blocks: the block that makes the particularity of this type of neural networks, the feature extractor; and the block that classifies the extracted features using a another neural network, the classifier.

To extract the features the CNN applies convolution filtering operations returning feature maps, which can be then normalized and resized using a pooling layer. The pooling layers reduce the number of parameters and calculations in the classification network, improving the efficiency of the model and avoiding over-learning.

The most important things to configure in a convolutional layer are the number of kernels and its size, the activation function and its input shape. The Convolutional layer is basically applying a convolution operation through a matrix obtaining a number of feature maps. Then, the activation function takes the feature maps created by the convolutional operation and generates the activation maps as its output. For example, in the following Figure 5, it can be seen how the convolutional operation works. It extracts image patches and multiplies them with the filter or kernel of the convolutional layer, obtaining a value for that patch and reducing the size of the matrix. However, as the convolutional layer uses more than one kernel, the will be several feature maps for the same matrix.

Taking into account that a convolutional layer uses n filters, and that the number of how many filters are used in one stage is depends in the depth of the volume of output feature maps, the output $Y_i^{(l)}$ of layer l consists of $n^{(l)}$ feature maps of size $k_1^{(l)} \times k_2^{(l)}$., and the i^{th} feature map, denoted as the output $Y_i^{(l)}$, is obtained from

$$Y_i^{(l)} = B_i^{(l)} + \sum_{j=1}^{n^{(l-1)}} K_{i,j}^{(l)} * Y_j^{(l-1)} \tag{2}$$

where $B_i^{(l)}$ is the bias and $K_{i,j}^{(l)}$ is the filter connecting the j^{th} feature map in layer $(l-1)$ with i^{th} feature map in layer.

After convolution, the feature maps are passed to the pooling layer, which will reduce the dimension of the feature maps. There are two common functions used in the pooling layers, the average Pooling that calculates the average value for each patch on the feature map, and the maximum Pooling, also known as Max Pooling, that calculates the maximum value for each patch of the feature map. Most of the CNN use Max Pooling, as average pooling method smooths out the image and smoothing the sharp features that may not be identified after the pooling. Figure 6 shows how the max pooling layer works in a simple case. As a result of the pooling layer, the number of parameters to learn and the amount of processing in the network are both reduced. The pooling layer summarizes the features found in a specific region of the feature map created by the convolution layer.

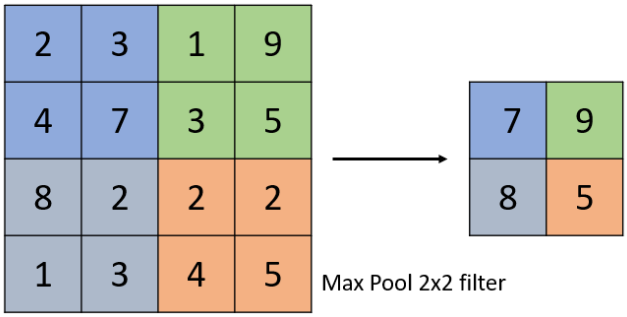


Fig. 6. Process of Pooling Layers

The use of this convolutional and pooling layer can be repeated several number of times, but at the end it needs to pass through a flatten function that transforms the feature maps into a vector. The vector is the input of the second block, where the Fully Connected Neural Network classifier reside.

Neural networks are a set of non-linear functions that are interconnected. Each connection goes from one node to another. This node is responsible for each particular function. The node uses a weights matrix to apply a linear transformation to its input. A non-linear activation function is then used in order to obtain a non-linear transformation to the product. Neural Networks are formed by layers, normally can be divided into an input layer, a hidden layer and an output layer. In fully connected neural network all possible connections layer to layer are present, meaning every input of the input vector influences every output of the output vector. This gives the network the ability to mix signals, since each neuron has a connection to each subsequent layer, there is now a flow of information between each input dimension and each output layer.

Proposed Solution

As explained above, the objective of this project is the detection of melanoma through a simple image in which the mole, mark or spot in question can be properly appreciated. For this purpose, a learning model based on convolutional neural networks has been elaborated and in order to make this model more accurate in order to reach the desired objective, a previous work has been done on the processing of these images, which is detailed below.

A. Images Pre-proccessing. First of all, the model needs all the images of the HAM10000 library mentioned above for the training of the algorithm. For this reason, all JPG images have been vectorized and normalized between the values 0 and 255. It has been decided to use images with a resolution of 50x37 pixels in RGB format, which is the optimum quality that can be handled with a 1GB RAM machine. In the following Figure 7 it can be seen the quality of one of the images that have been reconstructed from the different vectors previously generated.

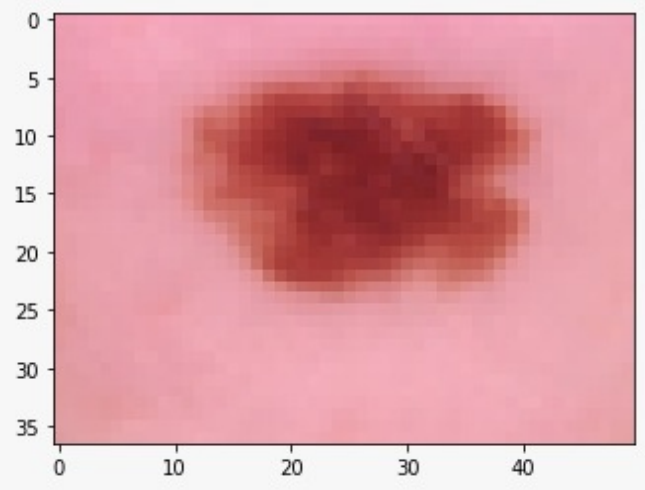


Fig. 7. 50x37 pixels image example.

After storing all the vectors corresponding to each image in an array, the data have been divided into two groups, one for the training of the algorithm and another to test how good is the classification that the model will predict.

B. Data balancing. In order to perform the balancing of the dataset, necessary in our project due to the unbalance that exists between the two classes, whose importance has been discussed in the previous sections.
The following Figure 8 shows a representation of the images of the data set, in which the clear unbalance that exists between one class and the other can be observed. For this particular data set there are 6412 images of the "non-melanoma" class and 798 images of the "menaloma" class.

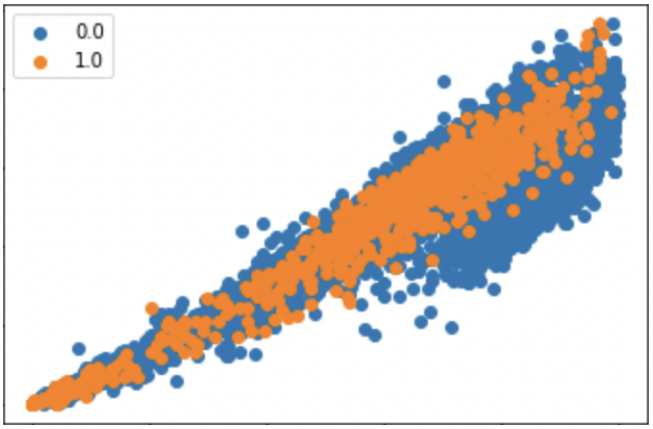


Fig. 8. Before applying SMOTE algorithm

This will be the starting point on which the SMOTE algorithm will begin to work. For this purpose, the function belonging to the imblearn package has been used for the realization and execution of the algorithm. This function allows different input parameters [9]. We have chosen to use several of them, which will now be commented.
The first parameter it has been decided to use an 80% strategy, this is used so that the SMOTE algorithm generates synthetic samples of the minority class until this class has 80% of the number of samples that the minority class has. A seed has also been added so that the synthetic samples that it generates are always the same during training and so as not to disturb the data that will be used in the training of the neural network. A third option has been added for the creation of synthetic samples in which the algorithm should use 10 neighbors when creating the synthetic sample. The reason for increasing the number of neighbors is to create higher quality samples. Finally a 4th parameter has been added, this does not affect the algorithm or the synthetic samples created, it simply improves the CPU performance when the algorithm is running.

Once the SMOTE algorithm has been run for the generation of synthetic samples of the minority class, we have the following samples available for training: 6412 images of the "non-melanoma" class and 5129 images of the "melanoma" class. The following Figure 9 shows the dispersion of the dataset samples, in which the synthetic samples created by SMOTE are already included and ready to start with the training.

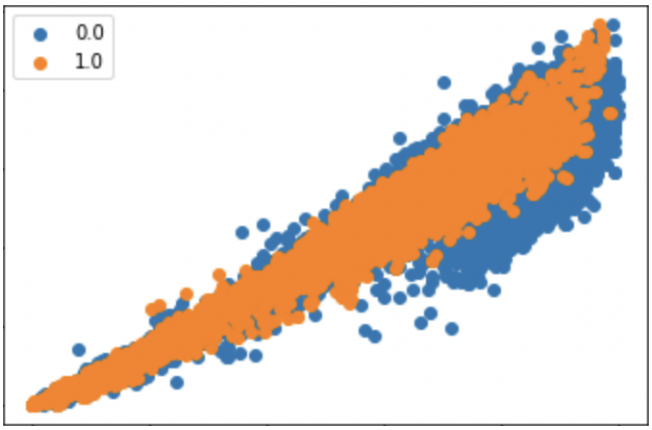


Fig. 9. After applying SMOTE algorithm

C. Model Creation. In order to feed the model with the balanced training and validation datasets, some data transfomation is needed. First of all, the labels of the model will use categorical encoding transforming the no melanoma class to a vector $\begin{bmatrix}1 & 0\end{bmatrix}$ and the melanoma class to vector $\begin{bmatrix}0 & 1\end{bmatrix}$. To ensure that every observation from the dataset has the chance of appearing in training and validation set, K-fold cross validation was used. K-fold cross validation calculates model's predicted level of fit to a data set that is unrelated to the data used to train the model. For this reason, the training and validation dataset are concatenated in one, so the algorithm can divide it depending on the number of splits.

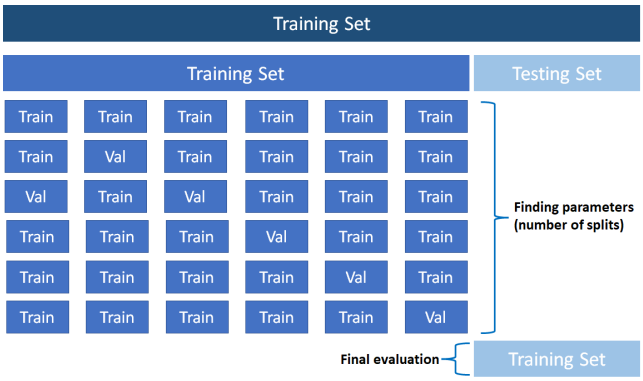


Fig. 10. KFold structure

This data will enter the Convolutional Neural Network through the first convolutional layer that is configured to accept the 50x37 size RGB images. This is a 128 3x3 filter convolutional layer with a ReLU activation function. The rectified linear unit (ReLU) is particular activation function that combines non-linearity and rectification layers. A ReLU is defined as[10]:

$$Y_i^{(l)} = \max\left(0, Y_i^{(l-1)}\right) \tag{3}$$

being l a non-linearity layer that takes the feature volume $Y_i^{(l-1)}$ from a convolutional layer $(l-1)$.
After passing through the convolutional layer, a dropout function is applied to this layer. Dropout it's a regularization technique used to reduce the overfitting by dropping out nodes of a layer at random, during every step of the training. The standard dropout is applied to each input node with a single parameter p during training, controlling the activation of each node x_j with a gating variable α_j for each forward pass:

$$y_i = \frac{1}{p}\sum_{j=1}^{N} w_{ij}\left(\alpha_j \cdot x_j\right), \alpha_j \sim \text{Bernoulli}(p) \tag{4}$$

where w_{ij} is the weight of the connection from input neuron x_j to the output neuron y_i, and where $\cdot$ denotes scalar multiplication and α_j is an independent Bernoulli random variable which takes the value 1 with probability p (the retain ratio) and the value 0 with probability $q = 1 - p$ (the drop ratio). The scaling factor $\frac{1}{p}$ scales the output activation to maintain the predicted value of the output. In this case the probability q is 0.25.

Next, the pooling layer receives the activation maps and a max pooling function using blocks of 2x2, reducing the size of the feature maps to ease the complexity of the data. After this, the activation maps passes trough an new 64 3x3 filters convolutional layer with a ReLU activation function, with a dropout of 0.25. The activation maps generated from this new convolutional layer are reduced by the pooling layer because the 2x2 blocks. The activation or feature maps are flatten after this pooling layer to feed the fully connected neural network with 128 dense nodes that follow a ReLU activation function and a dropout probability of 0.8. The output of this layer is finally the input of the output layer that consists of two dense nodes with a softmax activation function that converts the input vector of numbers into a vector of probabilities. The probability for each class is deduced from each value in the softmax function's output.

In the two following it can be found how the network is constructed and some important information about it, as it is the output shape of each layer or the number of parameters [11].

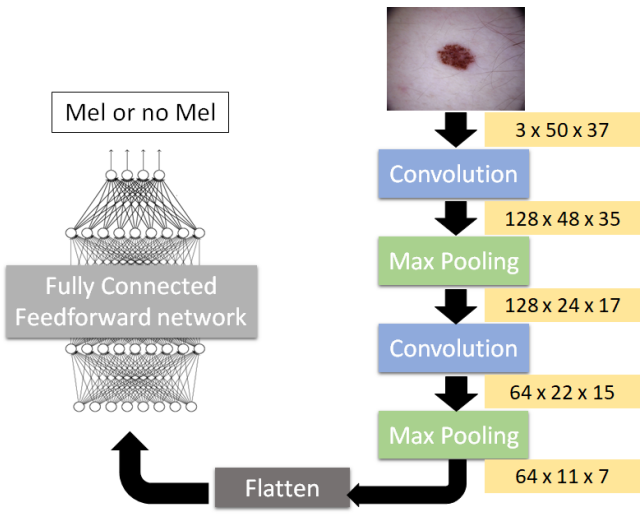


Fig. 11. Pipeline of the Convolutional Neural Network

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
FirstConvLayer (Conv2D)      (None, 48, 35, 128)       3584
Dropout (Dropout)            (None, 48, 35, 128)       0
FirstPooling(MaxPooling2D)   (None, 24, 17, 128)       0
SecondConvLayer (Conv2D)     (None, 22, 15, 64)        73792
Dropout2 (Dropout)           (None, 22, 15, 64)        0
SecondPooling(MaxPooling2D)  (None, 11, 7, 64)         0
Flatten (Flatten)            (None, 4928)              0
Dense (Dense)                (None, 128)               630912
Dropout3 (Dropout)           (None, 128)               0
OutputLayer (Dense)          (None, 2)                 258
=================================================================
Total params: 708,546
Trainable params: 708,546
Non-trainable params: 0
```

Fig. 12. Model Summary in layers

Finally, when fitting the data the model uses an Adam optimizer with a learning rate of 10^{-4}, batch size of 64 and 50 epochs. These values were obtained by trial and error and some prior knowledge. The update rule for Adam is as follows[12]:

$$\theta_{t+1} = \theta_t - \frac{\eta \cdot \hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \tag{5}$$

where η is the learning rate, ϵ is just a small term preventing division by zero and where:

$$\begin{aligned} \hat{m}_t &= \frac{m_t}{1-\beta_1^t} \\ \hat{v}_t &= \frac{v_t}{1-\beta_2^t} \end{aligned} \tag{6}$$

where m_t and v_t are moving averages of gradient and squared gradient, and β_1 and β_2 are introduced hyper-parameters of the algorithm. The moving averages are updated as:

$$\begin{aligned} m_t &= (1-\beta_1)\, g_t + \beta_1 m_{t-1} \\ v_t &= (1-\beta_2)\, g_t^2 + \beta_2 v_{t-1} \end{aligned} \tag{7}$$

where g_t is the gradient on current mini-batch.

As the labels are encoded using categorical encoding, it uses categorical cross-entropy instead of binary cross-entropy as loss function. It has the following form [13] [14]:

$$L(\hat{y}, y) = -\sum_{i=1}^{C} y_i \cdot \log \hat{y}_i \tag{8}$$

where $\hat{y}_i$ is the i-th predicted value, y_i is the corresponding target value, and C is the number of classes in the model output.

Results

As explained above, the images have been preprocessed to obtain these vectorized and normalized images. Once the data were balanced and the model based on convolutional neural networks was created, the training of the model was started. The k fold cross validation strategy and a voting strategy were used for the training.

First, the data set was divided into two datasets, one for training and one for testing. For the first strategy, k fold cross validation, the training set is chosen and divided again in two, one set for training and the other for performing a validation of the trained model. This process will be repeated 11 times, resulting in 11 different models that are able to classify an image as melanoma or non-melanoma. After training the 11 models it is time to check the accuracy of the models.
Last but not least, the voting strategy. Now the data set created for testing has been chosen, and 11 preditions are performed based on the 11 models trained with the previous strategy. The 11 predictions generated have been evaluated and the model with the best results is the final model chosen. To make the decision, the accuracy for each of the classes (melanoma and non-melanoma), the overall accuracy and the balanced accuracy of the model have been taken into account. It is important to highlight that as the dataset is imbalanced it is necessary to take into account the balanced accuracy, to see how well the model predicts the minority class.

In the following Figure 13 a confusion matrix can be seen in which the true positives, false positives, true negatives and false negatives are shown. Based on these positives and negatives the following accuracies have been calculated and can be seen bellow:

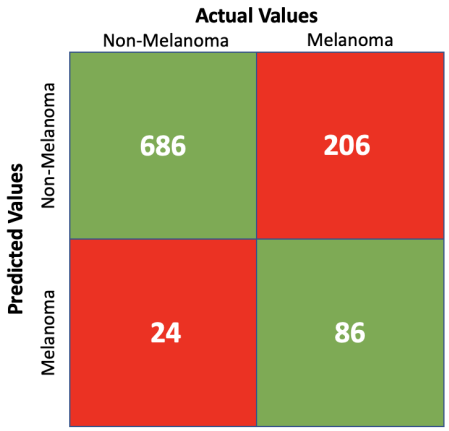


Fig. 13. Confusion Matrix.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} = 77.05\% \tag{9}$$

$$Sensitivity = \frac{TP}{TP+FN} = 76.91\% \tag{10}$$

$$Specificity = \frac{TN}{TN+FP} = 78.18\% \tag{11}$$

$$BalancedAccuracy = \frac{Sensitivity + Specificity}{2} = 77.55\% \tag{12}$$

Conclusions

Observing the results obtained, it can be said that they are not excellent, but they are good if we take into account that the main resource available in this research has been a machine with 1GB of RAM.

In medical terms, it is better to diagnose a disease when the patient does not really suffer from it, than not to diagnose a disease, when you really do suffer from it. At this point, the accuracy of the model created is quite acceptable, and it is reasonably accurate in predicting whether a spot, a mole or a mark is a melanoma.

Future Work

After analyzing the results of the research, they indicate that there may be some improvements that can be made to improve the accuracy of the model. One of these improvements could be the use of a better performing machine that would allow better quality images to be used during model training.

Another improvement that would allow greater ease and simplicity would be the creation of an application, in which by taking a photo with a phone camera of a spot, a mole or a mark on the skin, it would predict whether it is a melanoma or not through the use of the model created.

Bibliography

References

[1] American Cancer Society. 2021 estimates cancer statistics. https://cancerstatisticscenter.cancer.org/! [Online]; access 10/27/2021.

[2] Skin Cancer Foundation. Leader in the fight against skin cancer. https://cancerdepiel.org/quienes-somos/lider-en-la-lucha-contra-el-cancer-de-piel. [Online]; access 10/27/2021.

[3] American Cancer Society. What is melanoma skin cancer? https://www.cancer.org/cancer/melanoma-skin-cancer/about/what-is-melanoma.html, 2019. [Online]; access 10/27/2021.

[4] Skin Cancer Foundation. Melanoma overview. https://www.skincancer.org/skin-cancer-information/melanoma/, 2021. [Online]; access 10/27/2021.

[5] American Cancer Society. Signs and symptoms of melanoma skin cancer. https://www.cancer.org/cancer/melanoma-skin-cancer/detection-diagnosis-staging/signs-and-symptoms.html, 2019. [Online]; access 10/28/2021.

[6] Swastik Satpathy. SMOTE | Overcoming Class Imbalance Problem Using SMOTE, 01 2021.

[7] Rich Data. SMOTE explained for noobs – Synthetic Minority Over-sampling TEchnique line by line | Rich Data, 08 2021.

[8] Rok Blagus. SMOTE for high-dimensional class-imbalanced data, 03 2013.

[9] imblearn. SMOTE — Version 0.8.1, 2021.

[10] A. Kohler. Convolutional Neural Networks for Image and Video Processing , 10 2016.

[11] deeplizard. CNN Output Size Formula - Bonus Neural Network Debugging Session, 05 2019.

[12] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[13] How to choose cross-entropy loss function in Keras?, 08 2021.

[14] Categorical crossentropy loss function | Peltarion Platform.

Supplementary Note 1: Source Code

https://github.com/mfernandezlopezareal/Machine-Learning-CS-584-Group-8

NOTICE:

- Original HAM10000 dataset is not included. They can be downloaded from the Harvard directly, instead the CSV files of the processed images can be found.
- The outputs and results files are not included. They can be obtained by running the source code.