**Malignant skin leason screening with binary classification**

# Kivonat/Abstract (100-200 szó)

# 1. Introduction

Even though people are getting more conscious and careful about sun exposure and medical screenings, melanoma is still the most lethal of all skin cancers, and unfortunately, many diagnoses are made far in the advanced states of the illness.

The possibilities provided by neural networks and image processing are certainly a promising way for advancements in melanoma screening, not only helping to detect the illness in its early stages, but also preventing unneeded removals, increasing doctors’ confidence and reducing variability coming from human error.

Using the ISIC archive of images and diagnostic data, we decided to build a neural network solution classifying images of skin lesions as malignant or benign.

# 2. Melanoma classification, related works

A probléma sajátosággai: bőr, képfeldolgozás, orvosi megoldás, kritikus info, informális jelleg. Manuális módszerek

Korábbi megoldások: nem tudom

# 3. System design

A 2D convolution layers were used, in the making of the system. Usually it is used for different classification or regression tasks. If the structure is looked at, it’s made up of convolutional, pooling and activation layers, typically followed by fully connected layers.

The input parameters of the convolution layers are as follows:

1. {depth}@{input\_x}x{input\_y}
2. {filter\_x}x{filter\_y}@{filter\_depth}
3. {stride\_x}x{stride\_y}
4. {zeropadding\_x}x{zeropadding\_y}

In this case, the input of the network is a picture with the size of 64X64 (width, height) and thanks to the RGB channels the depth is 3. The number of filters was set to 32 and the kernel size was set to 3x3.The stride and zero padding were left on their default values.

The structure of the network is made up of three convolutional layers, followed by a max pooling layer with a 2x2 kernel size in all three cases, thus somewhat reducing the size of the network and then a dropout layer. The dropout layer helps prevent overlearning, here 30% of the neurons of the layer are deactivated randomly. After the three convolutional, max pooling and dropout layers, there is a flattening layer that “flattens” the values into one line. The flatten layer is followed by two dense layers, which is really the fully connected layers. At the dense layers, the only data needed is the specific number of outputs and the activation function. The activation function selected for the first dense layer was the ReLU, and the function selected for the second one was softmax, since ultimately the object of the task is binary grading. The activation functions of the convolution layers are all ReLU.

# 4. Implementation

## 4.1. Data aquisition and preprocession

For this project we decided to use the data provided in the open source public access ISIC (International Skin Imaging Collaboration) archive maintained by the International Skin Imaging Collaboration: Melanoma Project. This archive is a vast collection of close-up images depicting various skin lesions like nevi, dermatofibromas and melanomas. Data regarding the subject of the image like sex and age, location data describing the site on the body the skin lesion is found, diagnosis describing whether the lesion is malignant or not and segmentation images along with segmentation masks helping to hide the lesion’s surroundings during processing can also be found in the database.

Our plan was to use all available data and to use the flow\_from\_directory function of keras, so a script was written that creates a directory structure we could organize the thumbnails of images based on their malignancy diagnosis, and the segmentation masks and also saving them into the array of tuples containing the images in memory and their metadata.

Since downloading the whole database resulted in errors, and other sources had also had problems with the aforementioned functionality, we decided to use the public API available, which made it possible to download images one by one programmatically. The API provides an endpoint which returns a set size of JSON array containing metadata collections of images. These metadata collections are the diagnostic information and also basic information like image id to use in other queries. The other queries include the downloading of thumbnails, segmentation images, segmentation masks.

We used the json, request and opencv2 modules in the data acquisition code. By requesting the list of image metadata, we could extract right away the diagnosis. If it was anything other than benign or malignant, we dropped the record. Then using the diagnosis we generated a path for the file, for example: image\_thumbnails/benign, in case of a sample being diagnosed as benign, processing its thumbnail image. We also read the file immediately with opencv and put them in an array of sample data like this: thumbnail\_image, segmentation, segmentation\_mask, metadata.

We also split the data in this phase into two parts. The first was to be used as training and validation datasets, and the second as data for testing purposes.

The metadata we looked for was age, anatomic site where the lesion was found, the melanocytic nature of the lesion and the sex of the subject. We normalized age, filled in missing data with avarages, and one hot encoded categorical properties, but since many records showed missing data, we decided against using metadata.

We also considered generating segmentation masks based on the existing ones via using the grabcut algorithm implemented in opencv or via using a separate network and then using those images in the classification process, but decided not to do it due to inaccuracies and potential data loss.

We used keras’s ImageDataGenerator class and its flow\_from\_directory function to preprocess the images. Two separate generators were created: one for training and validation, and one for testing. We used the class\_mode=’binary’ argument, which made the input data generation from the premade directory structure really easy. We also used flipping, shearing and zooming for data augmentation.

## 4.2. Training

## 4.3. Evaluation

Initially there were only two convolutional nets with the flatten layer and dense layer. The last layer’s activation function was softmax, but there were relatively big validation loss and low accuracy. To further improve our network, the learning rate was modified with the ReduceLROnPlateau function, so that if the validation accuracy is no longer improving, the learning rate is reduced. After that an early stopping point was set, then a model checkpoint was made to save the values of the net when the validation accuracy is the highest. At this point, the validation error was still larger than intended, so the net was completed with a newer convolutional layer that used a 64 filter instead of a 32. After this was done, a few dropouts were set to prevent overlearning, so the precision got to an acceptable value.

loss: 0.3564 - acc: 0.8431 - val\_loss: 0.2628 - val\_acc: 0.8750

## 4.4. Testing

# 5. Conclusion

At the beginning of the semester, we preferred classifying each of the birthmarks according to whether they were malicious or not. For this task, we intended to use not only the photos of the birthmarks themselves, but also the “parameters” of the individual: age, gender, case history and on which part of the body the birthmark is located.

To do this, we processed the meta data that we received with the images. However, after making the neural network, we realized that based off of the images we can only achieve an accuracy of 80% on whether the birthmark is malicious or not.

Our future plans include that we will better define this with the help of the meta data. And to teach and test the system with higher-resolution images, because of the low-resolution, the more pronounced alterations and signs might go unnoticed.