Designing a Decision Support Mobile Application for Lung Cancer

Computer Engineering
Senior Project Report

Advisor : Tuğba Önal-Süzek, PhD

Group Members : Emre Taşkın

Date : 12/06/2022

Designing a Decision Support Mobile Application for Lung Cancer

Emre Taşkın
Computer Engineering
Muğla Sıtkı Koçman University
12/06/2022

Summary

Lung cancer (both small cell and non-small cell) is the second most common cancer in both men and women.[1] Lung cancer is divided into two main classes as Small Cell Lung Cancer (SCLC) and Non-Small Cell Lung Cancer (NSCLC). NSCLC type lung cancer constitutes approximately 80-85% of lung cancers.[2] The types that will be included in the data set that we will use during our research are Lung Adenocarcinoma (LUAD) and Lung Squamous Cell Carcinoma (LUSC), which are two of the three subtypes of Non-Small Cell Lung Cancer (NSCLC), as well as Small Cell Lung Cancer (SCLC).) type.

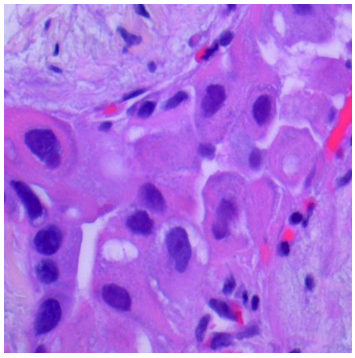
Although there are artificial intelligence models produced for glaucoma testing, the fact that none of them has been turned into a website and made available to doctors makes our project unique. This project aimed to minimize the problems faced by ophthalmologists during the decision-making process and to provide them with a second opinion. The product we will reveal will be a website that every ophthalmologist can easily use and get support while diagnosing glaucoma.

When our project is considered from a macro perspective, it consists of two main stages:

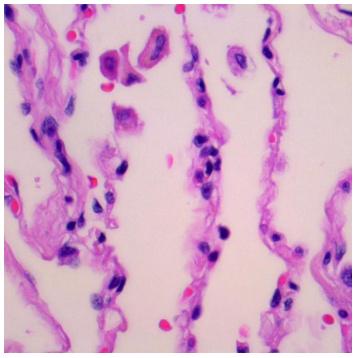
The first stage is the creation and training of the machine learning model. Kaggle contains data of approximately 11000 patients. Lung cancer image data (LUAD, LUSC) belonging to 2 subtypes in Kaggle has been downloaded. Our data set was divided into two sets as training and testing without disturbing the balance of the response variable. The machine learning model was trained with the help of the images in the training set with the help of the

CNN algorithm using the appropriate parameters.

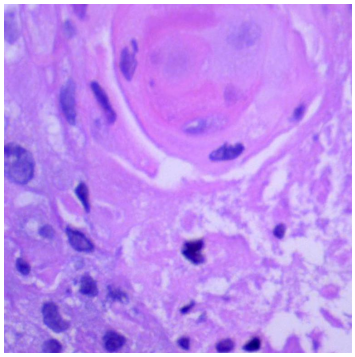
As the second stage, mobile application development is planned. The application, which will be developed by targeting the Android platform, has a very simple interface in order to be easy to use. Our application, which will work with the principle of uploading the tissue images obtained as a result of the test and transferred to the digital environment, on the clinician's own phone or taking the photo, will analyze the image with the help of a machine learning algorithm that will run in the background. The connection of the trained model with the application was made with the help of TFLite.



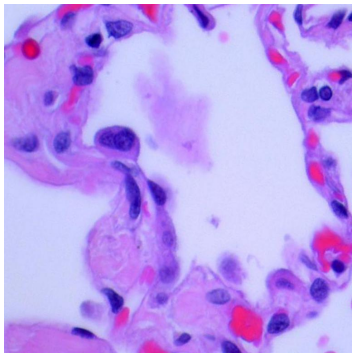
Adenocarcinoma



Benign



Squamous cell carcinoma



Benign

1. Introduction

2. Methods

i. Data Collecting

This dataset[3] contains 15,000 histopathological images with 3 classes. All images are 768 x 768 pixels in size and are in jpeg file format. The images were generated from an original sample of HIPAA compliant and validated sources, consisting of 750 total images of lung tissue (250 benign lung tissue, 250 lung adenocarcinomas, and 250 lung squamous cell carcinomas) and augmented to 15,000 using the Augmentor package. There are three classes in the dataset, each with 5,000 images, being:

1. *Lung benign tissue*
2. *Lung adenocarcinoma*
3. *Lung squamous cell carcinoma*

ii. Training Artificial Intelligence Model Using Open Source Machine Learning Libraries

I trained a sequential model using python's open source machine learning libraries, keras and tensorflow. Using Convolutional Neural Networks, I developed algorithms and models to distinguish between benign and malignant skin cancers. For source code editing, I utilised Colab[4].

I setuped data generators to read images from our source folders, transform them to float32 tensors, and feed them to our network . Data that is fed into neural networks is normally normalised in some way to make it easier for the network to process.

In my situation, I preprocessed our images by converting the pixel values to the [0, 1]

range (all values are now in the [0, 255] range). The input data must be scaled to 224×224 pixels as an input, as specified by the networks.

```
Building model with https://tfhub.dev/google/tf2-preview/mobilenet_v2/feature_vector/2
Model: "sequential_4"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
keras_layer_10 (KerasLayer)  (None, 1280)              2257984
_________________________________________________________________
flatten_11 (Flatten)         (None, 1280)              0
_________________________________________________________________
dense_22 (Dense)             (None, 512)               655872
_________________________________________________________________
dropout_11 (Dropout)         (None, 512)               0
_________________________________________________________________
dense_23 (Dense)             (None, 2)                 1026
=================================================================
Total params: 2,914,882
Trainable params: 656,898
Non-trainable params: 2,257,984
_________________________________________________________________
```

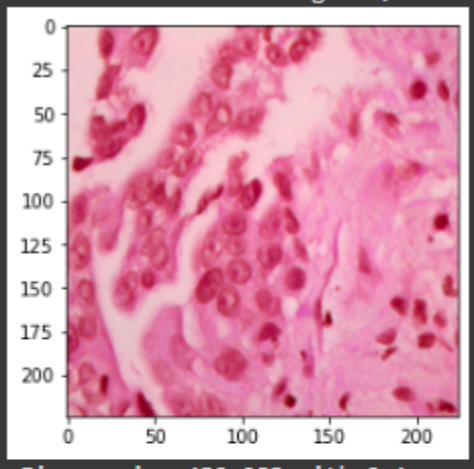
Validated each step by training the model with the validation dataset. I achieved 99% accuracy after 15 epochs.

```
Epoch 1/15
656/656 [==============================] - 1905s 3s/step - loss: 0.0283 - accuracy: 0.9919 - val_loss: 0.0095 - val_accuracy: 0.9978
Epoch 2/15
656/656 [==============================] - 656s 1000ms/step - loss: 0.0182 - accuracy: 0.9954 - val_loss: 0.0138 - val_accuracy: 0.9951
Epoch 3/15
656/656 [==============================] - 652s 993ms/step - loss: 0.0125 - accuracy: 0.9967 - val_loss: 0.0064 - val_accuracy: 0.9976
Epoch 4/15
656/656 [==============================] - 648s 987ms/step - loss: 0.0090 - accuracy: 0.9975 - val_loss: 0.0148 - val_accuracy: 0.9967
Epoch 5/15
656/656 [==============================] - 652s 994ms/step - loss: 0.0104 - accuracy: 0.9978 - val_loss: 0.0076 - val_accuracy: 0.9978
Epoch 6/15
656/656 [==============================] - 649s 990ms/step - loss: 0.0108 - accuracy: 0.9977 - val_loss: 0.0253 - val_accuracy: 0.9956
Epoch 7/15
656/656 [==============================] - 649s 989ms/step - loss: 0.0099 - accuracy: 0.9980 - val_loss: 0.0200 - val_accuracy: 0.9960
Epoch 8/15
656/656 [==============================] - 650s 990ms/step - loss: 0.0116 - accuracy: 0.9970 - val_loss: 0.0109 - val_accuracy: 0.9980
Epoch 9/15
656/656 [==============================] - 643s 980ms/step - loss: 0.0091 - accuracy: 0.9980 - val_loss: 0.0147 - val_accuracy: 0.9962
Epoch 10/15
656/656 [==============================] - 643s 980ms/step - loss: 0.0058 - accuracy: 0.9988 - val_loss: 0.0036 - val_accuracy: 0.9991
Epoch 11/15
656/656 [==============================] - 641s 977ms/step - loss: 0.0074 - accuracy: 0.9980 - val_loss: 0.0119 - val_accuracy: 0.9969
Epoch 12/15
656/656 [==============================] - 638s 973ms/step - loss: 0.0076 - accuracy: 0.9984 - val_loss: 0.0043 - val_accuracy: 0.9982
Epoch 13/15
656/656 [==============================] - 642s 979ms/step - loss: 0.0049 - accuracy: 0.9988 - val_loss: 0.0289 - val_accuracy: 0.9942
Epoch 14/15
656/656 [==============================] - 644s 982ms/step - loss: 0.0053 - accuracy: 0.9988 - val_loss: 0.0053 - val_accuracy: 0.9984
Epoch 15/15
656/656 [==============================] - 651s 992ms/step - loss: 0.0087 - accuracy: 0.9978 - val_loss: 0.0346 - val_accuracy: 0.9944
```

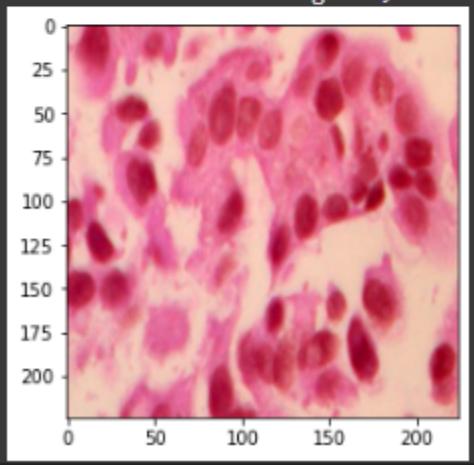
iii. Testing the Artificial Intelligence Model

```
print("SOURCE: class: %s, file: %s" % (os.path.split(filename)[0], filename))
img = upload(filename)
prediction = predict_reload(img)
print("PREDICTED: class: %s, confidence: %f" % (list(prediction.keys())[0], list(prediction.values())[0]))
plt.imshow(img)
plt.figure(idx)
plt.show()
```

```
SOURCE: class: malignant, file: malignant/lungaca1602.jpeg
PREDICTED: class: Malignant, confidence: 1.000000
```




```
<Figure size 432x288 with 0 Axes>
SOURCE: class: malignant, file: malignant/lungaca2603.jpeg
PREDICTED: class: Malignant, confidence: 1.000000
```



iv. Designing the Mobile Application

I have developed a simple and easy to use interface using the Kotlin language with Android Studio. Users can add photos from files or take a new photo and use it if they want.

Figure 3: *Mobile interface of our project*

Select Photo: opens the panel to select photos from files

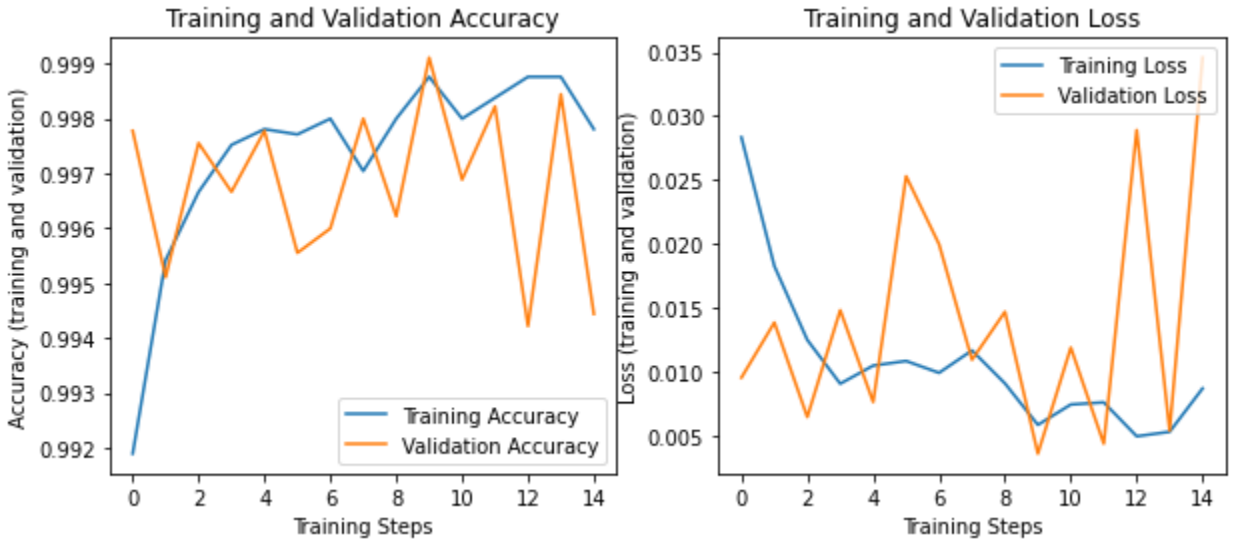
Start Camera: starts the camera to take a new photo

Detect: runs the model. makes the prediction result print just below the Detect button

v. Publishing the Application on Play Store

After the necessary fees are paid, an application can be created to be uploaded to the play store.

3. Results



MobileNet[5] was used while training the model. Currently, the model works with 99% accuracy. There is a ready-to-use apk file with a very easy to use interface.Designed to be functional in every aspect as a decision support mobile application

4. Conclusion

In order to decide which model should be the most accurate in data science projects, we need to evaluate the demands coming from the business units. Accuracy is a metric that is widely used to measure the success of a model.

The CNN model that I collected data from kaggle and trained with mobilenet is ready to use. Combined with a simple android interface with Tensorflow Lite[6], it makes predictions with an accuracy rate of 99%. Thanks to the mobile application I developed to support pathology specialists, pathologists will be able to both be sure of the accuracy of the results they interpret and detect an overlooked phenomenon. Pathologists will introduce the images they have obtained to the application through the phone camera or uploading the image on the device, and the machine learning algorithm that will run in the background will analyze the uploaded image.

5. References

[1] Cancer.org. 2020.

https://www.cancer.org/content/dam/CRC/PDF/Public/8703.00.pdf

[2] Cancer.org. 2019. What Is Lung Cancer? | Types Of Lung Cancer.

https://www.cancer.org/cancer/lung-cancer/about/what-is.html

[3]

https://www.kaggle.com/datasets/andrewmvd/lung-and-colon-cancer-histopathological-images

[4] https://colab.research.google.com/notebooks/welcome.ipynb?hl=tr

[5] https://keras.io/api/applications/mobilenet/

[6] https://www.tensorflow.org/lite#:~:text=TensorFlow%20Lite%20is%20a%20mobile,Lite%20Android%20and%20iOS%20apps.