Design Document

Core Body Temperature Estimation to Detect Ebola Virus Disease

CS 461, Fall 2017, Group 34

Claude Maimon, Brian Lee Huang, and Bianca Beauchamp

Abstract

The end goal of this project is to end up with a research paper. The paper should outline the problem that the project is trying to solve, the steps taken to solve the problem and how successfully we were at solving the problem. The paper should also allow the project to be continued if someone chooses to. The whole process should be explained in detail allowing whoever wants to continue the project to continue without any problem. The main body of this research paper will be about the program that we develop to predict a person's core body temperature. The program should first be able to extract data from a thermal image. The data of the image should come from the top half, focusing on the head. It will then interpret the data to create a mathematical model that uses the temperature of a person's skin as data and analyzes that information and predicts what their core body temperature is. A high accuracy rate is not strictly required as that is not the point of the project, the goal of the is to determine if this method will be effective to detect whether a person is symptomatic with Ebola. A high accuracy rate is a good indicator that a mathematical model is a good way to predict, wherse a low accuracy rate indicates that we should look for an alternative method.

CONTENTS

1 Introduction

2 Importing images and data

The first step for image processing is importing the image from the camera. It's important that the image will stay in the right format to keep the pixels' data. This means that the image needs to be in a format which maintains the temperature data.

We will use the FLIR Tools software for this part. FLIR Tools is a software for importing and analyzing images from FLIR cameras. The software can be used to import images to a personal computer, search the image library using various filters, store search criteria and manipulate images. The software is free to use and has a simple installation process. FLIR Tools is suitable for our FLIR A315 camera. Other than importing images the tool can be used to create PDF reports, add headers logos to images and sort the image folder by specific variables.[1]

This software is the best option for us because it's free and it offers all the features we need for this process. Even though FLIR Tools+ offers more features, we don't need them. By using the free option, we will make the project cheaper and still get all the capabilities we need to transfer the files. Other than just importing images we will also collect ear temperature from people. We need these temperatures to compare them to the estimated temperature from the camera. We will collect that data by taking people's temperature and store them in a file. Every temperature will be paired with an image, so we will know which temperature belongs to which person.

Our team still doesn't have the thermal camera. There is a chance that the camera will not get to the school in time for this project. In case that that happens, we will have to fake the data. We will use normal black and white images taken from a web camera. We will treat these images as if they were thermal images. The results that we will get from this images won't be correct but they will help us build our model.

3 Images into right format

In order to process the thermal images, they have to be in the right format. After importing the images from the camera to the computer, our program will transfer the images into a two-dimension array format. This format will allow for an easy image processing. Every element in the two-dimension array will hold a value of a pixel in the image. This structure will allow for an easy processing of the pixels' values.

Our program will use OpenCV with Python to manipulate the image. OpenCV (Open Source Computer Vision Library) is a free open source computer vision software. It has interfaces for C++, C, Python, and Java and it supports various operating systems. OpenCV's library has more than 2500 algorithms which offer many features. Some of those features are facial recognition, gesture recognition, motion understanding, biomedical analysis and more. The software is used all around the world and has a strong user community and offers technical support. OpenCV is also known by our client so we will have access to people that know how to work with it. OpenCV is a good pick for this part because it is free and has a large users' community. It has shown to have best performances in comparison to other image processing libraries and it has many easy to understand tutorials available. Moreover, our client suggested that we will use it.[1]

4 Image Processing 3

5 learning 1

6 learning 2

7 learning 3

8 Statistical Analysis

The statistical analysis will be the intermediate step between the training portion of the project and the production code. The analysis will be used to determine which model that the training portion creates will be the best, and will give the most accurate prediction. This piece of the project can only really be completed after we have gathered data, and after the learning portion of project is complete. Without the model produced by the learning portion there will be no way to compare to see how the model is doing, however the analysis of the data can really be done as soon as we receive the data to analyze.

For this piece we have chosen to use R as the analysis tool as it is open source and the whole purpose of R is statistical analysis. For the analysis we should simply be able to input the data, that should come in a text file into R. If the text file is not in the correct format, then we would build small parser to change the format into something R can interpret. To analyze the data we would simply use the built in functions in R to produce some sort of analysis. Since R is a statistical analysis tool there should be many different regression models that we can use for this analysis. If none of those produce desirable results we would then research different models we could use, then build them ourselves. The output of this analysis should be multiple graphs and equation of models that we can test and compare the accuracy of. The leaning portion should the try and build the model that we decided that was best.

This portion does not require much testing or debugging as most of the analysis are done by function calls. Much of the testing would just be testing different models to see which model best fits the data.

The timeline for the statistical analysis is to start building the framework that the data would be inputted into. This way when we get the data there will be at least some groundwork to make it easier. As soon as we obtain the data the analysis can begin since we should already have the framework working. Then we would wait for the learning portion to be done to then compare and choose what model should be used in the production code. The time we start this piece of the project really depends on when we get the data.

9 Production Model

The production model is the second piece of the overall project. What the production code does is it takes the model produced by the learning portion of the project as well as a thermal image, It then analyzes the image to produce a temperature which then gets fed into the model to predict a temperature. The output of the production model would then produce a single temperature based on the model. The model will be written in python like the other pieces of the project to eliminate any need for cross communication of languages. Although this is an important piece of the project most of it can only be done after the image analysis portion is complete. The production code also requires the model created by learning portion of the project. As long as there is an established format for model that will be produced we can create the code based on that format, then debug when the learning portion is complete.

We would not explicitly start on this portion at the very beginning, but instead we would need to research what kind of libraries we would need to process the image and model. Although it is a different part, the image processing is still

integral in both the learning portion and the production portion. The first piece of the project that would be developed would be the image processing portion, which would be indirectly working on the production model. Both the learning, and production code rely on processing the thermal image to get a temperature to be used as input and analyzed. Besides the image processing portion, there most likely will not be a need for any outside libraries. This means that the research for this portion of the project will be minimal and would mostly focus on ways to optimize the run time of the code.

We would like the production model to be accurate as possible since that is what will be used to predict a patient's core body temperature. This means we would want to test as many models as possible and run as many test cases in those models to produce the best possible case for core temperature prediction. To test the program itself we would use unit tests and mutation tests to test various temperature cases to make sure the program works with fringe cases as well as normal cases. The goal of the production model is to be as accurate as possible, but to also keep the false negative rate as low as possible. We are aiming for a 40

The timeline for the production code will roughly be, we wait till the image analysis is almost complete, then we start working on the code that analyzes the temperature. Second we work on the code that parses the model that is produced by the learning portion. Third we integrate the image analysis and debug anything that comes up from combining the two pieces. Fourth, we test the code to make sure everything runs smoothly and without any errors. Lastly, we test which model will be the best and give the lowest false negative rate.

10 User Interface

Once the project gets started a user interface is going to be one of the first things constructed. The first iteration of the user interface will most likely just be created out print statements that print out the results to the screen, and the input to the program will be done through command line. The user interface will be created using pythons file input and print statements. It will be up to us to properly parse the input files and properly input the relevant information. This simple interface will most likely be used for most of the project, if not the whole project. As the project becomes more and more complete the user interface should also improve with the project. The user interface will slowly develop itself as the project continues to progress as we would need to print relevant information to the screen. The user interface for the majority of the project would just take command line arguments for input, and then output information using print statements. We would not need to explicitly work on the user interface until the end of the project when we want to fine tune what is printed to the screen.

This project contains two major pieces of code, the training portion of the program and the production model. The training piece takes in a picture and the actual core temperature, and then produces a model that is some line that will be used to predict core temperature. The production piece of this project takes in the model created by the training portion and a image. It will then produce a core body temperature. The user interface won't deviate from command line, until the project is reaching completion. Then a new user interface will be made from python that is nicer but still simple to use. Instead of using command line for input and output the user interface should allow the user to select the input file using the mouse and keyboard and then allow the user to choose where the output file should go. The output of the training portion should output a text file that is a model for the production model to take as input. The output of the production model should just output a temperature that is predicted from the model. Instead of outputting a text file, the production model should just output the predicted temperature to the screen. This user interface would be for our end product that will be handed over to a graduate student to finish the research and implementation.

There will be not extensive testing of the user interface. To test the command line user interface we would simply need to test which piece of data we would need to print to the screen and which piece of data we would need to take as input. For the simple non-command line user interface we would need to test if it is grabbing the correct information from the input, and then correctly outputting the information to the screen. This would require some research into how the user interface libraries in python work and would need to be done before the project is near completion.

The timeline of the user interface will roughly be, we first create a command line user interface and develop is as we develop the main programs. Second we do research into the various user interface libraries in python. Third we create a simple user interface using those libraries for the final product.

11 Conclusion

References

[1] C. Maimon, "Claude's technical review," https://github.com/maimonc/Ebola-Virus-Project/blob/master/Documents/TechReview/Claude_Maimon_Tech_Review/Tech_Review_Claude_Maimon.pdf, 2017.