Summer Internship Project
Report

Opioid Counter-Diversion

Submitted by

Simon LEBEAUD
4th year Student at ITI Department
National Institute of Applied Sciences of Rouen

Under the guidance of

Dennis SHASHA
Designation of the guide

Department of Physics
University of New York
Address

Summer Internship 2020

Thanks

First I would like to warmly thank professor Dennis Shasha from New York University for giving me the opportunity two work on this outstanding project, for the guidance given and his indubitable benevolence.

I would also like to thank my collaborator Julien Leclerc from the GM3 department of INSA Rouen who as done some marvelous work as well as Enzo Chouisnard and Caspar Lant for the great advises throughout our collaboration and weekly meetings.

Finnaly I would like to thank professor Benoit Gaüzère and the internship department of INSA Rouen for there guidance and work trough the administrative procedures.

Contents

Chapter 1

Introduction

1.1 Institution presentation

The University of New York, commonly called NYU is a private research university historically based in GreenWich Village on the island of Manhattan in New York. NYU was founded in 1831 par Albert de Galladin on a simple principle, student should be chosen upon merit and not birth right.

Research universities are institutions that put research as one of there main mission. NYU is the largest independent research university in the United States. The university is divided in 6 campuses, the main one being in Greenwich Village. NYU also has campus in Abu Dhabi, Shanghai and has academic centers trough out the world, even in Paris in the 5th borough at the École Spéciale des Travaux Publics (ESTP), hosting 250 students each years.

The University is composed of 10 undergraduate schools, and is organised around 25 schools in total which able it to have a very diverse and complete span for research.
NYU welcome 40 000 students every years, who can choose trough 2500 different courses. New York University also employs 6 755 professors. Known for the quality of the teaching, NYU is ranked 35th at the QS World University 2020 ranking.

My internship has been done under the supervision of professor Dennis Shasha, professor in computer science at the Courant Institute of Mathematical Sciences that composed a group of 4 students, Enzo Chouisnard from Polytech Sorbonne in Paris, Caspar Lent from New York University, and Julien Leclerc and myself from INSA Rouen, to work on the Opioid counter diversion project. The project was seperated into 2 groups, a group having for mission to design and build a dispensing mechanism and one group composed of Julien Leclerc and I having for mission to develop the software using this dispensing unit.

1.2 Topic

Opioids are substances that act on opioid receptors to relieve pain and produce a feeling of pleasure. Medically they are primarily used for pain relief, but pleasure make it the primary reason why they are addictive for our nervous system. To avoid redistribution and resale of legally sold opioids to third parties (either addicts or criminal agents for addicts), we seek to ensure that the correct person takes each pill that is sold. This is a big problem in most developed countries where medicines can easily be prescribed and accessed.

The goal of the internship is to approach the problem using high technological resources that are easily available in those countries and more particularly in the United States where the problem is well known. Our approach is to develop a minimally privacy-intrusive surveillance mechanism called OApp that films the taking of a pill to prevent pill diversion. The surveillance takes place only during the time starting from when the pill is removed from the dispenser to the time it has been placed in the mouth of the target patient for at least 10 seconds. At all other times, the dispenser-camera setup can be covered.

The solution we developed involves a lot of detection and tracking to monitor the patient's behavior. Generally, OApp analyzes a video each time a patient takes the pill. Specifically, the person in the video needs to be recognized, letters on the pill need to be detected, and the pill itself needs to be tracked until it remains inside the patient's mouth for ten seconds. In addition, we need to do hand detection to ensure the patient doesn't take out the pill during the dispenser-to-mouth process.

In summer 2019 a first implementation of the solution as by Xiyuan Zhao, Hashim Hayat, Handi Zhang, Tairi Zheng, Shrey Jain, and advisor Dennis Shasha. This first implementation was to be run on a computer. This year we started by improving some of the detection algorithms and then we started a totally different, yet challenging approach.

Chapter 2

Work Done

2.1 Improving text recognition from Oapp 2019

As I said before, in 2019, a research group as been successfully working on the development of Python based software. This first version was made to be used on a fixed processing unit, that would be able to dispense the medications, and who would be internet enabled in order to run heavy computation on a distant NodeJS Server.**Appendix**[**A**]**:**OAPP-2019's architecture

2.1.1 Specifications

The biggest drawback of the 2019's implementation was time computation. A very important step of the Opioid Counter-Diversion system is that we need to verify if the pill the patient is showing to the camera is the right pill. In order to do that, 3 numbers are printed on the pill. The system is done such that the patient doesn't have time to falsify the pill and create a new one copying the numbers.
One of the first instruction given to the patient is to put the pill in front of his mouth with the numbers clearly visible.
The moment the person follows this command, the numbers on the pill need to be detected and recognized in order to know if the sequence of numbers correspond to the right pill. If the numbers correspond we can assume that this is the right pill.

The process of detection and recognizing scene text is very time and resource consuming, knowing that at simultaneous time we have to run face, mouth, hand, pill detection and face recognition, recognizing the numbers in

real-time is not achievable in this case. That is why when the patient show the pill, frames are captured and store so that the recognition can be ran afterward.
Even if the computation was done afterward, the process was taking a little less than 2 minutes in order to compute the classification. So the first mission was to improve the time computation while keeping the detection very accurate.

2.1.2 Solutions and critics

Even with the documentation we couldn't run the Oapp 2019 text detection an classification, and the main author of the implementation wasn't reachable so we decided to make our own implementation.
Two steps are conducted, detection and classification. Those two steps need to be understood, detection is done in order to find the text in the image and classification is only done on the text we found inf order to know what characters are on the pills.
In our case we know that the text is going to be on the pill and that the pill is in the bounding box of the mouth. That said we thought that we didn't need to run text detection since we already knew where the text was. The first problem encountered was that the numbers on the pill are very small. We didn't get any good result running the classification. In fact, the character being small, the bounding box of the mouth is still to big for the classification method so we needed to implement a text detection method to only get the bounding box of the text and run the classification on it.
Browsing the internet we came across many method to do scene text detection. Scene text is text that is capture in frames with a complex background. We came across many scene text detection methods such as EAST (Efficient and Accurate Scene Text Detector) or DGST (Discriminator Guided Scene Text detector).
Our goal was to improve time computation without compromising accuracy, that is why we prioritized the use of OpenCv, which is a vision library specialized and optimized for image treatment and other python library that have time efficient computation and accuracy.

2.1.3 Chosen Solution

First, it has been decided that we would replace numbers on the pill with capital letters. We found that we had better result in detection and classification using letters, it is also better since you get more combinations (26^3).

As I said before the work is divided into two steps, detection and recognition.

Detection

For detection we decided to go with a EAST based detection. This choice was made because it seemed to be the fastest in comparison to other implementation that we found on the internet.
The EAST based detection is a deep convolutional neural network pipeline that is capable of detecting scene text at any orientations with a 360p minimum resolution. Forward passing an image through this pipeline gives us in output for every detected text, a score and the geometry of his bounding box. We are then able to crop the given image and only take what's in the bounding box in order to recognize the text in it. Some preprocessing as been done after this step in order to improve text recognition, it will be addressed in the next section.

Recognition/Classification

Now that we can detect scene text on the medication and take only what we need from the frame, we can try to accurately read the text on the pill. For this step I knew a python tool that I used during my image treatment (TIM) class last year. So we chose to use the python library pytesseract.
Pytesseract is a optical character recognition tool for python. Pytesseract is in fact a wrapper for Google's Tesseract-OCR Engine. This OCR is well documented, accurate and gives us the direct result of the text recognition compared to some OCR that write the result in a file.

2.1.4 Implementation

My collaborator and I started this project at the end of the first week. Before that, we first had to understand the implementation that was done last year, and since we couldn't run last year's text detection implementation due to package compatibility issues, we started to build our own.
We still worked in two steps. Our idea wasn't to rebuild the all wheel, that is why we chose tools that we knew were available to us. For the detector we first found a pretrained frozen EAST model and Kaggle. This model was presented at the ICDAR 2015 so we knew we could trust this model. During one of the step the patient has to follow we know that the pill is going to be in front of the mouth of the patient with the number clearly visible. Knowing that and having the bounding box of the mouth for the 5 frames we need

to analyse during this step, we can crop everything but the mouth for each frame and keep that in memory to be processed later. Those 5 frame need to be processed to determine if the text is right, and if so it would be the correct pill. We chose to follow the same pipeline than Oapp 2019 and decided to process those frames at the end of the opioid consumption verifiction process. Here is how we decided to analyse the image:

- First we load the model
- Than we need to resize the frame at size 640*640 in order to fit the input of the model
- We forward pass the image trough the model and extract score and geometry for each possible text
- We only keep bounding box that has a score over 0.5. We then need to apply non-maximum suppression over those box in order to counter the problem of having overlapping bounding box.
- Then in our case, we only obtain one rectangle that contains our 3 letters. We re-scale the box to fit the original image and we extract only the box containing the text

After all those steps we came across our first problem. It sometimes append that all the letters weren't contained in the bounding box. So we couldn't run character recognition having letters with missing part or cut in half. So we decided to scale the box up with a ratio of 1.1. This gives us full letters and able us to run the recognition with valuable inputs.
After this minor tweak we can run text recognition using Pytesseract. We came across our second problem, we didn't have very accurate result. What we found with the OCR didn't match the letters we were looking for.
Reading the documentation I found out that the OCR work better with gray images, so I decided to binaries the image given to the OCR. This changed everything on our result.

2.1.5 Obtained Results

Having accuracy and time in mind we did obtain relatively good result, we could fully process those 5 frames, in between 15 to 20 second.
For the text detection, when done in good illumination and not having the sun chinning right in the face of the patient we were able to correctly detect the right letters on at least 3 of the 5 frames. So we decided that 3 good detection/recognition out of 5 was our threshold to decided whether it was the right pill or not. This first project took us a little more than a week.

2.2 Application Android

The second part of the internship was a change in direction for the Opioid Counter Diversion system. It as been decided with professor Shasha to take the project a step further and to build a dispensing unit that would dispense the pill. With Julien we thought that it would be a good idea to use technologies widely available in develop countries. Nearly everyone has a smartphone, so with the approbation of professor Shasha we started developing an Android application while another group of two collaborators was designing and building the dispensing unit.

2.2.1 Specifications

The app needed to follow the same system than the OAPP 2019. First the patient connects to the application. On the first connection the patient need to have his picture taken to verify his identity. Then if the time is right, the patient can choose to start the verification and dispense a pill. Before a pill is dispensed to the patient, face verification is done and if the right person is recognized, a pill is dispensed.
During verification process the patient as to follow 7 steps once the pill is dispensed :

- **Step 1 :** Patient as to place the pill in front of his mouth with the letters clearly visible for 5 seconds.
- **Step 2 :** Patient as to place the pill on his tongue (or between his lips) and remove his hands from the visual field, the pill still need to be seen, the pill shouldn't disappear between the two steps (or the verification will be labeled as suspicious)
- **Step 3 :** Patient as to close is mouth during 10 seconds, during this time, neither hands nor the pill should be visible or the verification is labeled as suspicious.
- **Step 4 :** Patient as to open is mouth and show that the pill is still in his mouth. In this step, no hands should be detected. After this step the patient can swallow the pill.
- **Step 5 :** Patient as two wait 5-10 seconds for the frames to be analysed, and a the results are shown to the patient.

All those verification steps imply that we need different tools for :
- face and facial landmark detection
- face verification
- hand detection
- pill detection
- text detection and recognition
- user authentication

We choose to develop the app for devices supporting at least Android 5 Lollipop.

2.2.2 Solutions and critics

Same as the first project, we want tools that are fast and accurate. Most of the vision based tools have to be usable in real time. A phone doesn't have as much computing power than our laptop computers so I knew we couldn't have our tools running only on the smartphone device. We first had to implement or find a backend system who would have more computation power and on which we could run our detection tools and store data.

One of the benefits of developing a Android app was that android java is compatible with OpenCv and Tensorflow, which we new we were going to need. With OpenCv we could reuse some algorithms that were implemented with OpenCv in OAPP 2019. Also we knew that we were going to have to use Tensorflow for machine learning based detection tools.

The verification process needed to be done in a way that a patient cannot trick the app and substitute his medication.
To do that we had to try ourselves to trick the app into thinking we consumed the pill when we did not. Countering human activity is difficult since we had to "think like criminals". This train of thoughts made the app much more draconian in the verification process.
For testing purposes we used standard, white, 00 size pills, with a sticker containing 3 letters on it. The sticker did take the form of the pill so it imitated pretty well what we would of had with a manufactured pill.
All our testing was during the day, not in direct light.

2.2.3 Chosen Solution

In this section I will explain all the tools we put together to make the Android application.

First, as I said, we needed a backend system, that was reliable, fast and accurate. Our first idea was to build our own backend system. For example if we had built a backend in Django we could of re-use some of the code we already had in python. We didn't took this option since we were also going to work with the stream of data between front and back and also because we found a more reliable and faster tool. We chose to use Firebase instead. Firebase is Google's mobile application development platform. This platform is particularly interesting in our case since the Firebase's API is compatible with Android, can manage authentication, storage, and has plenty of machine vision tools that we needed. So we chose to utilize FirebaseAuth for the app authentication system, and the ML kit's face detection API that able us to detected any face in a given frame and to detect any facial landmark that we would need, like the mouth. We also used the text recognition API from firebase in order to recognize the text and the pill.

Pill detection

Knowing that we were going to use OpenCv java, we kept the algorithm implemented for the pill detection from OAPP 2019, and adapted it to java. This detection consist of looking all the pixel in the bounding box of the detected mouth and hand and to look for a specific amount of "white" pixels, that would correspond to the pill. We haven't found any other solution to do that since pills are quite small objects.

Face verification

Face verification was quite difficult since some phones don't have a built in face recognition firmware, and if they have it we cannot utilize it. To solve this problem, we used a widely known machine learning unified system for face verification, recognition, FaceNet.
FaceNet is a face verification system developed by Google researchers in 2015. This system able us to extract important face features from a picture of the person (face embedding) and compare it to a to a reference face embedding to see if it is the right person.

Hand Detection

In order to determine whether the consumption of the pill is suspicious, Oapp 2020 has to determine whether the patient has removed or swapped the pill. To do that, the app has to determine where the user's hands are. So, after the person puts the pill on his/her tongue, Oapp 2020 shouldn't detect any hand on the frame. If the person does so, Oapp 2020 tags the process as

suspicious and the step is not validated.
In order to run hand detection, we used a framework called MediaPipe. MediaPipe is a open-source cross-platform framework maintained and shared by Google. MediaPipe shares MLtools to facilitate the deployment of machine learning technologies into demos and applications. One of the tools is a hand and finger tracking system that can infer 21 three dimensional landmarks of a hand with only one frame. Oapp 2020 needs to know only if a hand was visible in the frame, so in the hand tracking pipeline from MediaPipe, Oapp 2020 uses only one of the first steps which is called hand_palm detection. We so used a DNN called a hand_palm model from MediaPipe in order to detect hands. With this solution, Oapp 2020 can accurately know whether there is a hand in a given frame. But sometimes we gets a false positive when the background isn't plain or if the user is moving the camera too much.

Throughout the process of verification we use OpenCv to capture the camera view of the smartphone. The video stream of a phone is 30FPS and we process 1 for 20 frames.

2.2.4 Implementation

We have worked with Java before this project but we weren't familiar with Android Java so we started with a one week Android training course on Openclassrooms. We decided to first define all the java "Activity" that we were going to need and we came with:

- FirstPageActivity is responsible for the users authentication.
- ChoicePageActivity give the user two options, change is picture or start the dispensing/verification.
- ChangeProfileImageActivity able the user to change his picture.
- GetTextActivity, this activity is only needed for our prototype, once the user choose to start the detection process, he first as to input the 3 letters that are supposed to be on the pill, so this activity is only for testing purposes.
- CameraActivity is the activity in which the user as to follow the recognition and verification steps
- ResultsActivity, when the verification process is over, the user goes into the ResulsActivity in which he can see the steps that were validated and those who are not.

Since some of the detection tools that we used needed a lot of testing, I decided that we were going to build for a mini-app for each tool that would only utilize the one tool. This helped a lot in testing and designing the final application. Once we knew that a tool was working correctly and that the time processing fitted our needs we then implemented it in the app.

While storage, face and mouth detection, text detection and authentication is handled using Firebase's APIs, we had to implement 2 core detection steps :

1. Face verification: Analyzing how many people are in the video and recognizing the user. Throughout the process the person should be the patient and the patient only.
2. Hand detection: Detecting whether there are hands in the frames and what they do.

Face verification

For the prototype, when a user connects for the first time to the application the user needs to take a picture of himself/herself. Eventually, this picture will be done by health authorities.

This image is saved on Firebase's cloud storage and is used to run face verification. The procedure is to compare the saved image to the images retrieved from the frames taken during the first phase of detection, after the pill is dispensed.

In order to run face verification we use a deep neural network model well known called FaceNet. We had to convert the keras FaceNet Model to a Tensorflow Lite model. TensorFlow Lite models (.tflite) are made to be used on smartphones and smaller devices in order to reduce computation time.

First, to verify the user's identity in a given frame, we need to detect his/her face on both the referenced image stored on the cloud and in the frame(s). Face detection is done by using the Firebase's ML kit API. If more then one face is detected on the frame, a warning is shown on screen.

Second, the model takes color images of size 160×160 in input, so once the system has detected the face, it crops the image to just the face and then reshapes the image to the right input size if needed.

Third, the image goes through the network. We than obtain the image's embedding vector of size 128.

This vector is what enables us to compare the two faces. In fact once the software FaceRecognitionDetector class obtains the embedding vector of the reference face and the face to be verified, it can compare those two vectors and determine whether the image shot from the dispenser is the intended recipient.
For now this embedding computation is done every time for the reference image, but it would be more efficient to store the embedding on the cloud.

The final step is to compare the two vectors (A and B) using cosine similarity:

$$Similarity = \frac{A \cdot B}{\|A\| \cdot \|B\|}$$

Once computed, the similarity gives us a result that ranges from -1 if the two vector are very dissimilar to 1 if the two vectors are the same.
Testing the face verification, depending on the hardware used, we determine that a threshold of 0.85 gives us a very accurate result.

Hand detection

Oapp 2020 hand detection runs on the phone itself, so the Tensorflow Lite model is stored in the assets folder of the application.
First Oapp 2020 loads the model. Than Oapp 2020 runs the input through the interpreter and gets the best result for the detection. If the output value is greater than a specified threshold, than we assume that a hand is detected. During Step 2,3 and 4, if hands are detected, we assume that the person has removed the pill from his mouth or that he exchanged the pill.

Once all the tools were developed we put together the recognition algorithms using those. This means that depending on the steps, a instruction is given to the user and for each steps we control if the patient respect the requirements and isn't suspicious.

2.2.5 Obtained Results

By the end of the internship we were able to control and verify the good consumption of the medication, and we did respect the real-time constraint. We don't have the same time processing problem from last years implementation. We still have some issues with false positive detection for the hand. The problem is due to the fact that sometimes a object or a form in the background can be falsely recognized as a hand, the problem is manageable when the user has a plain background. We also had issue with our phone.

My phone I bought 4 years ago and running android 8 couldn't quite process the app fast enough. But more recent phones we had no problems.

Chapter 3

Conclusion

During this 12 week internship at New York University, I have been working that keeps up with the times. This work was done in collaboration with Julien Leclerc, a student from the GM3 department of INSA Rouen. A great organisation was needed in order to coordinate our work. I learned a great much about communication, since Julien had less experience in development it taught me to explain clearly my point of view and guide him throughout our work. That was also a great experience that expanded my development skills.
This project impressed me at first since it seemed very complex to develop with the knowledge we had, but starting and working everyday made me more and more comfortable with it.

As you might have guessed, due to the current health risks, I wasn't able to make it to New York, and all the work was done in teleworking. Working at home required me to be rigorous with my work since I had a lot of distraction around me, but I found the project very interesting and useful so it helped quite a bit.
I realize now that working from home without having someone to verify and guide our work on a daily basis makes it harder to know if the work done is of good quality of if something more is needed, but this project was still of great intellectual enrichment.

To conclude on the work done, we did not manage to quite complete all the project. We did manage to complete the processing algorithm to verify the good consumption of pills. One downside of the project was that we couldn't quite connect the mobile device to the dispenser and manage communication with it in order to dispense the pill. Even if this hasn't been done, we did documented all our code and tried to make as evident as possible for future developers that would continue our work.

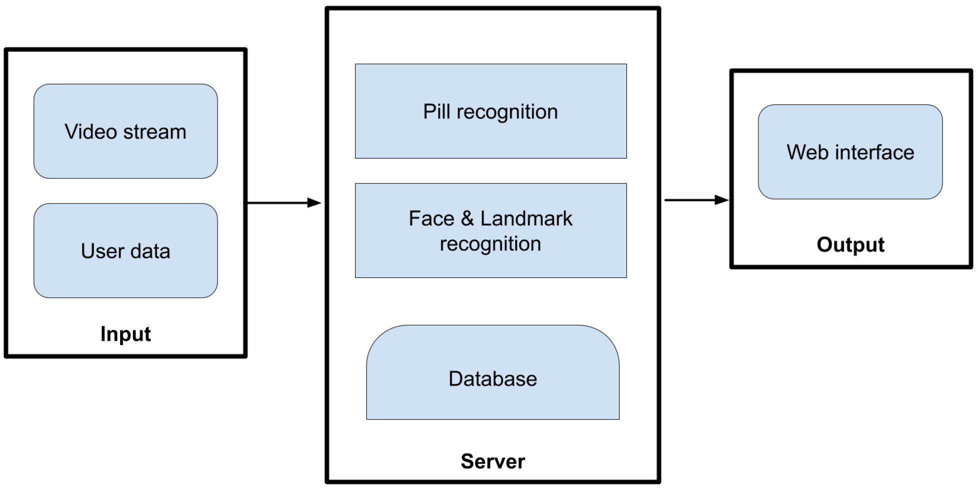
References

[1] Write the reference here

Appendix A

OAPP-2019's architecture

Author :



There are five major components

1. Video stream: The video stream comes in from the camera and is broken down into frames and then each frame is broken down to the pixel level for detailed analysis. In order to optimize the speed up the analysis in processes 2 and 3, OApp skips some of the video frames.

2. Pill recognition process: This process is responsible for the recognition of the pill at various locations such as in the user's hands and tongue. Pill recognition also comprises reading the number of symbols written on the pill, using number recognition tools. The process outputs the number written on the pill and determines whether the number was correct or not. The output information is sent to the server which stores them in a NoSQL database.

3. Face and landmark recognition process: The face detection and recognition process is responsible for detecting the patient's face and then compares the face with the list of pre-stored patient faces in order to determine whether that person is authorized to take the pill (i.e. "authentication"). The landmark recognition process is also responsible for recognizing and tracking patient landmarks such as hands, mouth, and tongue. Authentication and tracking output results are sent to the NoSQL database.

4. Server: The server, written in NodeJS, is responsible for managing various tasks and processes as the analysis occurs. It handles the communication between different components and synchronizes the data flow. The server gets triggered as the user starts to take a pill and launches processes 2 and 3 in different threads and listens to the results and updates the database in real time. The server turns these processes off when all calculations are complete.

5. Client interface: The web interface opens a realtime web socket connection to the database and listens to any updates pushed by the server. Based on the data received by the interface, the interface displays intuitive messages to the patient to guide them through the process. At the end of the process, the interface displays the outcome of the analysis.