CS Capstone Problem Statement Design Document

May 20, 2019

Rheumatoid Arthritis Detection

Prepared for

Karate Health

Brett Adelmen

Prepared by

Group5

Chase Denecke
Ethan Patterson
Jared Tence

Abstract

This document contains the roadmap for the rest of the year for the capstone group working on a rheumatoid arthritis application. Our plan is to begin collecting data for a joint detector, which could eventually form the first filter in a convolutional neural network designed to measure joint swelling. This network could eventually be the first filter in a neural network designed to measure inflammation by using the number of swollen joints in someone's hand as a proxy.
The first thing we will do is begin creating a Federal Information Security Management Act compliant mobile app for data collection. The second step will be developing a data processing pipeline for the neural network to remove noise and outliers from our data set. The third step will be collecting data for the joint detector, which will form the first layer of the CNN we could embed in the mobile app.

CONTENTS

Section	**Original**	**New**
Abstract	Contained a lot of information about our plan to deploy an app capable of measuring RA inflammation	Contains a scaled back, less ambitious project that ends with us collecting data for the joint filter.
Overview	Talked about making an app to measure RA	Talked about making an app to measure RA
Project Stakeholders	Talked about why the students and founders of Karate Health care about the project	No changes
Data Cleaning	Talked about image resizing and filler, removing pictures of right hands, masking out background, and adjusting lighting	Talks about resizing images and masking out the background
Image Cropping/Resizing	Contained methods describing how we would fill space and resize images to some standard dimensions	Unchanged
Right Hand vs Left Hand Classifier	Talked about how to distinguish between right and left hands	Removed section
Masking Out Image Backgrounds	Using opencv and possibly a neural network for this issue	Definitely need to use a neural network for this issue.
Data Cleaning Timeline	Contained deadlines for data cleaning and data collection that we completely missed and were for portions of the project we will never complete	removed section
Basic Network Architecture	Talked about how deep neural networks work	Talks about using MASK_RCNN
Hyperparameter Optimiazation and Feature Finding	Talked about hyperparameter optimization and how to perform the task (including talk of automatically calculating hyperparameters)	Removed section
Convolutional Neural Network Architecture	Talked about how to design and train a deep learning neural network	Removed section
Convolutional Neural Network Training	Talked about training, validation, and test sets	Talked about the same stuff but clarified that it is a stretch goal
Moving the Trained Network	Talked about how to move the weights file for a trained CNN from a training environment to a mobile app.	Talked about the same stuff but clarified that it is a stretch goal

Section	Original	New
Troubleshooting and Debugging	Talked about all the ways neural networks can have trouble	Talked about the same content but mentioned that we expect MASK_RCNN to not have any of those problems because it has been designed and tested extensively.
Neural Network Timeline	Set Jan 30th 2019 as target date for running network and March15th as target date for completion of CNN.	Kept dates but stated we may not meet them.
Data Collection with Food and Drug Administration MyStudies Application	Planned to implement and use	”Scrapped due to the MyStudies app being flawed
Neural Network Timeline	Set Jan 30th 2019 as target date for running network and March15th as target date for completion of CNN.	Kept dates but stated we may not meet them.
Data Collection with Food and Drug Administration MyStudies Application	Planned to implement and use	Scrapped due to the MyStudies app being flawed
App to Measure Rheumatoid Arthritis	Originally was going to use My Studies	”Used React Native with expo
App Deployment Timeline	We planned on releasing a prototype prior to expo	”Basic application (proof of concept) created by expo ”
Server	Planned on using Apache as our server framework	”Used NodeJS instead of Apache ”

1 Overview

The purpose of this paper is to give an overview of the technologies and methods we will use to complete the project. The purpose of this project is to lay the groundwork necessary to develop a mobile application capable of measuring joint inflammation due to rheumatoid arthritis. This work will consist of a mobile application, data cleaning tools like a segmentation network that masks out the background behind the image of a hand, and the dataset which can be used to train the neural network to identify joints in pictures of hands. This document is intended for individuals with a reasonable understanding of a convolutions neural network (CNN) and mobile development.

2 Project Stakeholders

The stakeholders in this project are the students, whose grades and resumes depend on the success of this project, and the company Karate Health, whose financial and technical backing make success a possibility. The student project managers have an interest in at least the completion of the project, which is required for a passing grade in the class, and at most the success of the project, which would have a positive impact on their job prospects and resumes. Karate Health has an interest in the completion and success of the project because they are providing financial and technical assistance for the project. If it were to succeed, they might have a considerable business opportunity on their hands which would dovetail nicely with services the company already provides to customers.

3 Data Cleaning

One of the most important and time-intensive steps in any machine learning process is data cleaning. Loosely speaking, data cleaning is the process of removing portions of the data to reduce irrelevant information and increase desirable characteristics like normal distributions. Without data cleaning, the model may not converge, may produce biased results, or it may take far more examples to achieve the desired accuracy.

The steps in our data cleaning process will be as follows: re-size the images so they all have the same dimensions and mask out the background of the image.

3.1 Image Cropping/Resizing

Image cropping is the most straightforward of these steps. Smart phones take pictures of different resolution, and CNNs require images of the same resolution in order to function properly. We will take the largest dimensions of any image taken by a phone, and pad all other images with zeroes in order to bring them up to these dimensions. Getting the hand cropping tool to work after this process may be tricky, so revision of this technique based on our future findings is possible. We plan to do accomplish this step with the Tensorflow function tf.image.resize_image_with_crop_or_pad().

3.2 Masking Out Image Backgrounds

This will be done using a second convolutional neural network to identify the boundaries of the hand in the image. The network will then remove anything that is not apart of the hand. This image segmenting network will be hosted as a server that our phone application's server can connect to and request image processing jobs as needed. Letting these systems live separated form one and other will help with scalability. The image processing network will be very resource demanding and should be run on a dedicated system, or over the cloud on some computing service such as Amazon

Web Servers (AWS). This network will be trained on images we annotated our selves. We plan to implement transfer learning using the already created coco dataset that can identify people, chairs, books, cars, etc. This will increase the rate of convergence for learning a new object. By implementing the method of transfer learning we should be able to train the network to identify hands by using data that we annotate our selves. We plan to use the 11kHands dataset to accomplish this[1]. The annotation software we will be using is the VGG Image Annotator[2].

4 The Neural Network

4.1 Basic Network Architecture

We will be forking a GitHub repository called MASK_RCNN and using that network to segment images of hands. Should we get around to actually creating the joint detector before expo, it is likely we will also use that network for joint detection. Mask_RCNN works with JSON files, so when we generate the training data it will be in JSON format. If we get around to actually training a network, we will evaluate its performance in the typical manner with a training, validation and test set.

4.2 Convolutional Neural Network Training

If we get around to actually training a network before expo, we will begin by splitting the collected data into three categories. The first category will be the training set. The training set will be composed of labeled images for the CNN to learn from. The second will be a validation set. The validation set will allow us to adjust the hyperparameters and evaluate the performance afterwards. The Third will be a test set. The test set will only be used once when we are confident he network preforms as expected. This will allow us to verify that the CNN is unbiased. To create these sets we will likely need a very large set of images. Ideally at least 1,000 will be apart of the training set, 500 for the validation set, and 100 for the testing set.

4.3 Moving the Trained Network

The TensorFlow API supports save and restore functionality for a CNN. This will allow us to train the network on a different machine and save the model. Once a model has been saved we can move it to our targeted mobile environment for production use. TensorFlow also supports mobile devices through the use of TensorFlow lite. TensorFlow lite supports a feature for converting a saved model into a compressed model for use on a mobile device. This is how we could move a trained model to a mobile device. Since this is a stretch goal, we likely won't achieve it. But should we do so, the method above would be our plan for accomplishing it.

For the network that removes background noise. Weight files can be generate and saved in a .h5 file. This makes the transfer of the wights to a new machine easy, however, it will be to large to run on a phone and will need a dedicated machine. To make the deployment of the background server easy, a docker file can be created to setup the dependencies and reduce the overhead of operating the instance.

4.4 Troubleshooting and Debugging

CNN are complex and because of this complexity are difficult to debug. However generally neural network debugging can be broken down into two issues. The first issue could potently be a problem with the data set itself. The second problem could potently be with the network itself.

A common way to avoid data set issues is to build the simplest network that solves the original problem and to build iteratively upon that network until a the global problem is solved. Another option is to try to over fit a network with small data sets. If the loss does not decrease there is mostly likely something wrong with the network itself. Mean centering can also be used to remove noise from an image. This is what we plan to do in the data cleaning phase using MASK_RCNN to remove background noise.

One common problem with networks is caused by their activation functions. In our case we must watch out for dead ReLUs. This will occur if the problem has negative numbers. This could result in a neuron activation weight of zero. If the activation weight is zero then the neuron will never fire resulting in a dead neuron. Another issues that is common for deep networks is network accuracy degradation. This will be a potential issue that we run into as our network becomes deeper. Microsoft solved this issue by using residual layers that introduce identity connections between each layer [3]. This issues occurs when the gradient signal vanishes in the training of the network. These identities reduce the signal lose of these gradient signals Hopefully none of these issues will occur since the authors of MASK_RCNN are well versed in their subject and many people have forked their repository, which means it probably doesn't have any major flaws.

4.5 Neural Network Timeline

The data cleaning must be completed first before we move onto building the CNN. This is very important to our client as it demonstrates that further investment is less of a risk. If we were to build this CNN first there is a possibility it may be unable to extract usable features. This would be the worse case resulting in a wast of resources collecting these image data sets. However, if we can perform some data cleaning this would provide some incentive to build a larger data set. Once this is complete we will begin working on the CNN. The goal for a running network is no later then January 30th 2019. The targeted date of completion for the CNN is March 15th 2019. We may not hit these goals.

5 Rheumatoid Arthritis Application

5.1 Data Collection with React Native

We will be using the React Native with the expo javaScript package as our framework when developing our application. We choose this over the alternatives because it provided us with the most flexability. Unlike the FDA MyStudies application React Native applications run on both IOS and Android. This allows the majority of our user base to use our application. Also React Native is robust as Facebook and Instagram use it as the base for their application and regularly update React Native.

The React Native base comes with all the major building blocks for our application. It has a listview for displaying queried data in a user friendly list. It has navigation options to allow for the user to navigate multiple screens using Tab,

Drawer, or Stack navigation. We can create custom components which will be usefully when creating surveys for RA joint inflmation. Last, we don't have to verify that our additions are within HIPPAA and FISMA regulations because we are not working with insurance companies or hospitals. While this will be important in the future currently compliance with these regulations isn't required when building our application.

Last the expo javaScript package comes with implementations of machine specific code such as accessing and taking photos using the camera. Expo is also the tool that will allow us to compile our React Native code into a Android or iOS binary so that we can publish or application to the store.

5.2 Rheumatoid Arthritis Application

As we develop the RA application we will also be developing the NodesJS server and Mongo Database along side it. The NodeJS server will be responsible for accepting requests from the RA application. The NodesJS server will need to service POST requests from the cleint so that we can save client RA data. It will also need to service GET requests and send RA data submissions to their respective users. When servicing POST and GET requests from the client the NodeJS server will also have interact with the Mongo Database and query and obtain stored information.

5.3 App deployment timeline

- Basic application and swollen joint survey created - Feburary 10, 2019
- Photo taking ability and user agreement implemented - February 28, 2019
- NodeJS server services the RA application requests - March 30, 2019
- Neural Network integration - April 16, 2019
- Fully functional application usable by RA patients - June 1st, 2019

REFERENCES

[1] M. Afifi. (2017) Gender recognition and biometric identification using a large dataset of hand images. [Online]. Available: Ihttps://sites.google.com/view/11khands

[2] A. Dutta, A. Gupta, and A. Zissermann. (2016) VGG image annotator (VIA). http://www.robots.ox.ac.uk/ vgg/software/via/. Version: X.Y.Z, Accessed: INSERT$_DATE_HERE$.

[3] A. Linn. (2016) Imicrosoft researchers win imagenet computer vision challenge. [Online]. Available: https://blogs.microsoft.com/ai/microsoft-researchers-win-imagenet-computer-vision-challenge/