ECE492: Senior Design II
Mar.io
Technical Progress Report: Winter 2019-20

Jared Balakrishnan, Sarthak Babbar, Akshar Amin
Sai Talla, Arjun Sachdeva, Abhiroop Verma

Advisor: Dr.Matthew Stamm

Department of Electrical and Computer Engineering
Drexel University

March 16, 2020

Contents

List of Figures

1 Executive Summary

With the meteoric rise of research activity in the area of Computer Vision, numerous applications have surfaced in the world, ultimately leading to a burgeoning multi-billion dollar market. The number of problems being solved in this area, ranging from simple handwriting recognition to autonomous driving has brought to our attention the extent to which human life could be simplified with the help of such technology, thereby helping us eliminate costs as well as leave a smaller carbon footprint.

Mar.io aims to solve a segment of the problems in the path detection area for autonomous system by building a small-scale version of a computer vision system which when iterated in large scale can be used in the areas of robots used for autonomous delivery as well as wheelchair assistance for the disabled.

The project aims to build technology that is committed to improving the quality of life, whilst respecting the communities in which it would be used by striving to make computer vision more simpler and accessible.

Mar.io is designed to detect traversable areas on sidewalks in addition to attempting pathway recognition by building a machine learning model that is trained upon sidewalk images fed through a microcontroller. The machine learning model is intricately composed of components that specialize in collecting, cleaning and analyzing the data prior to using it for the purposes of training and inference. Once this task of inference is successfully achieved, the microcontroller system would be able to perform this detection in real-time. This can then be viewed live on a web application. The entire model is slated to be built using the Python programming language and the libraries associated with it, whereas the web application is slated to be built using a mix of JavaScript and React.

The prototype of *Mar.io* is also designed to be extremely cost-efficient, with the bare bones prototype costing less than $600 and the large-scale business prototype projected to cost a little less than 2 million dollars over a period of two years.

2 Technical Updates

The technical design of *Mar.io* has not changed in any way since the submission of the last technical progress update report in December 2019. The bulk of the work carried out during this term was geared towards the development of the code base: that is, the translation of the high level data flow diagram into code. The high level data flow of the entire system is shown below as a refresher:

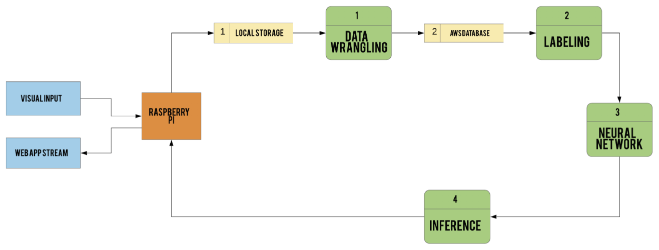


Figure 1: High Level Data Flow Diagram of the Mar.io system

The updates for each and every single part of the project since the beginning of the Winter term are as follows:

2.1 Data Collection

Using the Raspberry Pi based micro-controller system built in the Fall term, 8 to 10 hours of video have been collected for training purposes. Decomposing all of the collected video footage into each of its component frames therefore yields a total of 691, 200 images.

The challenges associated with the collection of training data largely relate to the weather conditions encountered while collecting data. We have collected data over varying climatic conditions in order to provide diversity to the training data set, which in turn implies that the machine learning model developed will not be over-fit or under-fit in any way. However, collecting data during sunny afternoons poses a challenge to the labeling process (discussed subsequently): handling the presence of shadows on the sidewalk could result in the incorrect labeling of some frames, considering how the number of frames in hand warrants an entirely automated process to label all the images.

2.2 Data Wrangling and Labeling

The sub component of the software system tasked with processing the collected video footage and subsequently labeling them correctly (in a binary fashion) as **sidewalk** and **not sidewalk** was one of the largest tasks in the project as of now. This is because of the fact that we used a camera operating at 24 frames per second to collect 8-10 hours of training data. This in return translates into a total of approximately 700,000 individual images (frames). Labeling these many images manually would take an unnaturally long amount of time, and given the fact that this is a supervised learning problem, it is imperative that all the training data be labeled correctly. Therefore, it was decided that an approach to automatically and correctly label all the images be devised, the high level diagram of which is shown below:

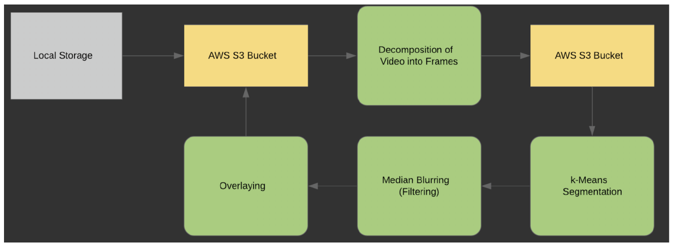


Figure 2: High Level Data Flow Diagram of the Data Wrangling and Labeling System

Once the training data was collected from the data collection system, the footage was transferred from local storage to an Amazon Web Services (AWS) S3 bucket. This S3 bucket is then used to serve these video files to a group of computers (namely the personal computers of the team members), where every second of video gets broken down into its individual frames. These frames are then stored in a specific folder, which is named and sorted according to the video number and then uploaded back to the AWS S3 Bucket.

Then, the labeling process commences by extracting every single frame from each one of these folders. The K-Means algorithm is applied with a value of $k = 2$, in order to make a binary separation of the area into **sidewalks** and **not sidewalks**. Upon doing this, a computer vision technique called Median Filtering is applied to the image in order to remove the noise from the images. Upon the application of the K-Means algorithm and the removal of noise, this resulting image is then overlaid on the original image by using the Python Pillow framework. The results are then fed back to the S3 bucket online for storage.

The results of this labeling process as applied to a sample frame is as shown below:

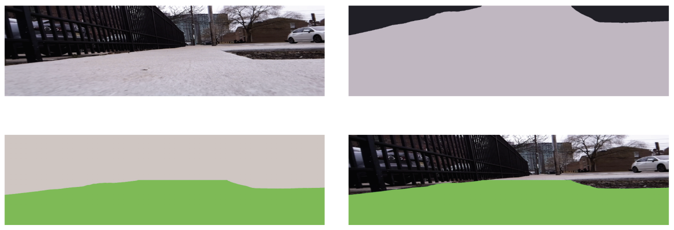


Figure 3: Results of the Labeling Process

In the above set of images, the original image is the one as seen on the top left. To the right of this image is the same frame after the application of the k-Means algorithm with the value of k set to 2. The bottom left image is the correctly labeled frame, which when overlaid on the original image yields the image at bottom right.

The major challenge associated with this part of the project was the process of labeling the images. This is particularly because of the presence of bottlenecks in the form of shaky internet upload-download speeds, and the lower computing power associated with a typical MacBook Pro when working with a large amount of data. On average, it took around 3-4 hours to label a video of 86400 images. Very little amount of video footage is left to be labeled.

As mentioned in the previous subsection, the presence of shadows constitutes a challenge to label every single image accurately. Methods to tackle this problem as well as alternate methods of segmentation such as Ground Truth Labeling and Bilayer Segmentation are being explored.

2.3 Training of the Neural Network

Upon the completion of the labeling process, a convolutional neural network (CNN) is trained with the newly labeled data. For this project, given the abundance of training data, it was decided that a CNN available **open source**, such as **UNet** or **SegNet** would be retrained with the labeled dataset.

This is the part of the project that the team is currently working on. As of today, an approach of using a small subset of the labeled images (around 44,000) to train different CNNs with is being used. This is done so in order to ascertain the better performing CNN for our purposes.

Testing **SegNet** with 44,000 labeled images from the data seems to have been particularly successful, yielding the below shown classification of a random sidewalk:

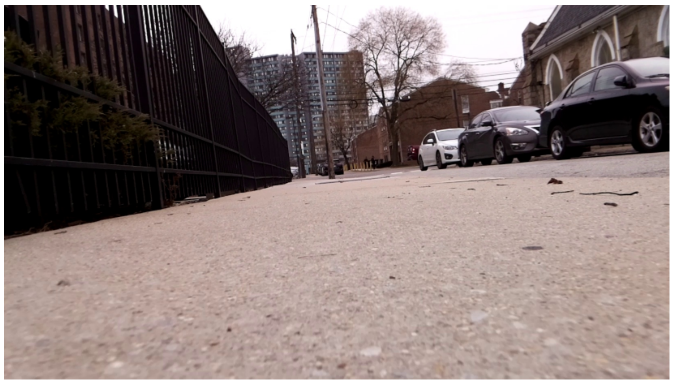


Figure 4: Sidewalk before classification by SegNet

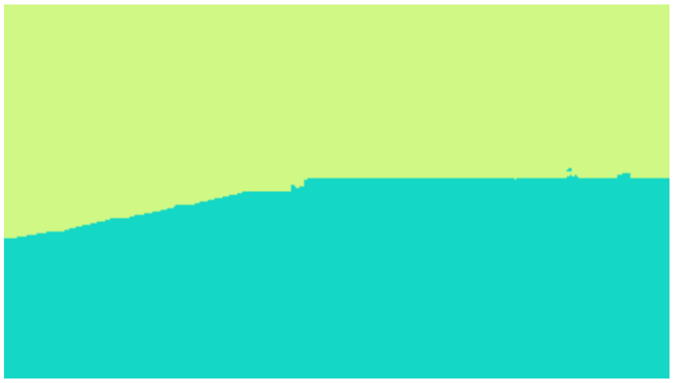


Figure 5: Sidewalk classified by SegNet

From the above figures, it could be seen that in Figure 5, the parts in cyan indicate the traversable parts of the sidewalk whereas the green indicates areas that cannot be traversed.

2.4 Inference and Streaming Application

The work on this part of this project is yet to commence because of the fact that we are working on choosing a good neural net to train the available training data with. Once a neural network is chosen, it would be retrained with the training dataset in hand. Upon training, feeding new sidewalk data to the system would yield a classification of the sidewalk into the areas that can be traversed and the areas that cannot be traversed. For the purposes of establishing inference, the Python framework **Keras** will be used.

For the purposes of demonstration, a web application will be developed with

React and JavaScript to live stream the output of the system as it is put into motion.

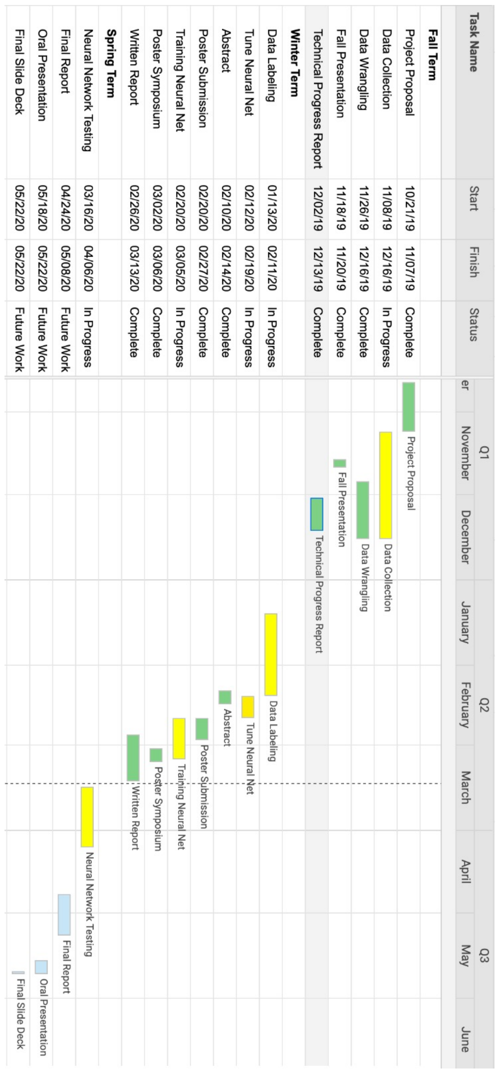
3 Other Updates

Fortunately, there are no other updates to the project, be it relating to the technical aspect of things or the business aspects of things. No new purchases have been made (all the purchases for the project were completed in the Fall term), because of which there is no updates to the bill of materials in any way either.

However, if in case there is any kind of update to be made in the remaining life cycle of the project, it shall be promptly reflected in the reports and deliverables due for the next term.

4 Updated Gantt Chart

The Gantt Chart for the project, to reflect the updates as of now, is as shown below:



Task Name	Start	Finish	Status
Fall Term			
Project Proposal	10/21/19	11/07/19	Complete
Data Collection	11/08/19	12/16/19	In Progress
Data Wrangling	11/26/19	12/16/19	Complete
Fall Presentation	11/18/19	11/20/19	Complete
Technical Progress Report	12/02/19	12/13/19	Complete
Winter Term			
Data Labeling	01/13/20	02/11/20	In Progress
Tune Neural Net	02/12/20	02/19/20	In Progress
Abstract	02/10/20	02/14/20	Complete
Poster Submission	02/20/20	02/27/20	Complete
Training Neural Net	02/20/20	03/05/20	In Progress
Poster Symposium	03/02/20	03/06/20	Complete
Written Report	02/26/20	03/13/20	Complete
Spring Term			
Neural Network Testing	03/16/20	04/06/20	In Progress
Final Report	04/24/20	05/08/20	Future Work
Oral Presentation	05/18/20	05/22/20	Future Work
Final Slide Deck	05/22/20	05/22/20	Future Work

Figure 6: Updated Gantt Chart for *Mar.io*