**Research – Week 5**

* Find out what different implementaiton options - different ways you can build a ml model on a mobile device next week?

**Resources:**

* <https://www.youtube.com/watch?v=pMIwu5FwJ78>
* <https://www.freecodecamp.org/news/machine-learning-for-mobile-and-embedded-devices/>
* <https://www.freecodecamp.org/news/end-to-end-machine-learning-project-turorial/>
* <https://www.youtube.com/watch?v=uTdUUpfA83s>
* <https://github.com/tensorflow/tfjs-models/tree/master/coco-ssd>
* <https://cloud.google.com/vision/automl/object-detection/docs/label-images-edge-model>
* <https://developers.google.com/learn/topics/on-device-ml>
* <https://www.tensorflow.org/lite/models/modify/model_maker/object_detection>
* <https://github.com/tensorflow/tfjs-models/tree/master/mobilenet> - tfjs models
* <https://topflightapps.com/ideas/how-to-create-a-machine-learning-app/> - explains mobile ML options
* <https://blog.tensorflow.org/2020/02/how-modiface-utilized-tensorflowjs-in-ar-makeup-in-browser.html>
* <https://developers.google.com/learn/topics/on-device-ml/learn-more>
* <https://stackshare.io/stackups/ml-kit-vs-tensorflow-js>

**Questions**

* Since Tensorflow.js is a ml library for JS do we even need to bother about any python implementation?
* How does ReactJs work with building a mobile app, as lot of implementations suggest using React Native?
  + My understanding is that React Native was for native mobile applications and React/React.js is for building web applications?

**Considerations about Implementation**

* We want all the ML to happen on the mobile device
  + No need for a connection to a server to do any backend work
  + On-device ML API
    - Benefits of on-device
      * Low latency
        + The client doesn’t need to access the server to process any requests/ inputs
        + Can leverage hardware acceleration (GPU or TPU) on device for fast results

Some mobile devices have been built with machine learning in mind – i.e. processors, graphics cards

* + - * + Low latency inference allows for more responsive real time results
      * Privacy
        + The data handled is limited to the device and won’t be processed by any server
      * Works offline
        + Don’t need network connectivity
      * Cost Benefit
        + Won’t need to pay for cloud options or server solutions
    - Issues/ concerns of on-device
      * Mobile device vs servers
        + Less computing power on mobile device
        + Models can become quite large

This is no issue for servers, however due to the limited hardware of mobile devices

ML model will have to smaller and not as powerful to adapt to mobile device

* + - * + Considerate about user’s hardware on mobile devices

Will they meet the requirements to run the application on their device?

* General plan so far:
  + CNN – deep learning
  + Tensorflow.js or ML Kit/ Firebase ML seem most favourable for on -device ML
  + Building a library of relevant ML models
  + Supervised machine learning for image classification
  + Cordova or Ionic
* Inference
  + During the training of the model when an image is passed and the model makes a prediction about what the image is

**Options for Implementation**

* Tensorflow lite
  + Native coding
* Tensorflow JS
  + - Can execute ML models in JS
    - On-device
    - Deep learning
    - ML in the
      * Browser client side
      * Server side > Node.js
      * Mobile native via react native
      * Desktop native > Electron
      * IOT devices via Node.js on Raspberry Pi
      * <https://www.tensorflow.org/resources/learn-ml/basics-of-tensorflow-for-js-development>
* Scikit-Learn
  + Tensorflow is better for deep learning
  + Python
  + Scikit-Learn is better for traditional machine learning
  + <https://scikit-learn.org/stable/>
  + <https://www.npmjs.com/package/scikit-learn>
* Amazon Rekognition
  + [https://medium.com/@glen.bray/text-detection-with-mobile-camera-using-react-native-and-aws-rekognition-7826b3e2aeef \](https://medium.com/@glen.bray/text-detection-with-mobile-camera-using-react-native-and-aws-rekognition-7826b3e2aeef%20\)
  + <https://morioh.com/p/4b289b022040>
  + Best for in depth facial recognition
    - Facial comparisons for same person
    - Facial feature detection
* Einstein Vision
  + Einstein Image Classification
    - Good for scale
    - Create and train models to detect and classify images
    - Use of pre-trained model?
  + Not on-device
    - Need Heroku
* ML5.js
  + On-device
  + JavaScript
  + Image classification
* Synaptic
  + On-device
  + JavaScript
  + Image classification
* Brain.js
  + On-device
  + JavaScript
  + Image classification
* Clarifai
  + Easy to implement API for tagging images
  + The API has strong concept modelling
  + Allows for potentially creating and training our own models to test against
* Google vision api
  + Good for detecting extensive details about the image
  + Requires Node.js
  + Not on-device
* Mobile vision api
  + Deprecated now ML Kit
* ML Kit
  + Vision Api’s
  + Native code focus
  + ML Kit
    - On-device api
* Firebase ML
  + Custom model deployment
  + Over the air?
  + ML Kit's on-device ML models are TensorFlow Lite models, so you can run similar models yourself on every platform that support TFLite
  + AutoML Vision Edge
    - On-device custom image classification models with an easy to use web interface
* Edge Impulse
  + Tiny ML model
  + Access sensors with JS
* Core ML
  + Native coding
* Keras or Tensorflow for python
  + Can train the deep learning model in python
  + Can save the model in a language independent format like json or HDF5
  + That trained model can then be loaded into JS ML library
* Tensorflow
* Flask + Keras
  + Python framework Flask
  + Machine learning library Keras
  + REST API capable of classifying images
  + Can use with React
  + Not on-device
    - Needs a connection to the backend
  + <https://medium.com/sopra-steria-norge/build-a-simple-image-classification-app-using-react-keras-and-flask-7b9075e3b6f5>
  + Example project structure:

Text

Description automatically generated with low confidence