**Internship Update**

**January 15 2018**

**Mark Shelton**

**Thickshake**

**Metadata Parsing (mtd)**

Functions:

* Reads in MARC file, parses, and loads into RDBMS
  + Extracts addresses and performs geocoding
  + Extracts image sizes from urls with minimal-loading
  + Extracts dates
* Flattens RDBMS and dumps into HDF5 for parallel access

Dependencies:

* postgres / psycopg2 / sqlalchemy - RDBMS (dockerized)
* hdf5 / h5py - parallel data store
* pymarc - MARC parsing library
* nominatim - OpenStreetMap geocoding API

Next Steps:

* Convert geocoding to async, multi-threaded requests
* Retrieve metadata from SLWA API
* Dump final metadata back to MARC? SQL?

**Image Processing (img)**

Functions:

* Pre-processes images (channel-based CLAHE)
* Detects and normalizes faces
* Generates and saves facial landmarks and embeddings to HDF5

Dependencies:

* cv2 - image processing & manipulation
* dlib - face recognition, embeddings
* hdf5 / h5py - parallel data store

Next Steps:

* Parallelize and prepare to run on Pawsey
* Currently 10,000 photos takes 10 hours - 1 image / 3.5 sec - @ JPEG 1024

**Classifier (clf)**

Functions:

* Loads faces and metadata from HDF5, selects relevant features and label
* INPROGRESS: Pre-processes dataset to prepare for machine learning
* TODO: Runs training & testing, outputs results and predictions

Dependencies:

* tensorflow - machine learning framework
* hdf5 / h5py - parallel data store

Next Steps:

* Build machine learning pipeline in Tensorflow
* Match embeddings across images – create anonymous ids
* Work out how to output classification results
* Parallelize and prepare to run on Pawsey
* Build basic web app to use as interface

**Other Notes:**

* Build is automated with Makefile commands (make start, stop etc.)
* Uses Docker Compose to manage application containers (app, db)
* Uses Docker to manage image builds, tags, publishing (with Docker Cloud)
* Environmental config files for application and image

**Old Perth**

Functions:

* Reads in MARC file, parses, and loads into RDBMS
* Dumps RDBMS into JSON file that has all info needed for website
* Website maps photos from JSON file onto locations
* INPROGRESS: Locations can be clicked on which loads info about the photo

Dependencies:

* thickshake.mtd - MARC file metadata processing
* google maps - mapping visualization and search API

Next Steps:

* Fix some issues with clicking on locations and loading photo info
* Parse more images - dependent on thickshake geocoding working async
* Add social media integrations - Facebook, Twitter
* Make cosmetic / content fixes