Youtube Scraper

This script will scrape videos from youtube, sort by precense of faces and conversation, and upload to s3 accordingly. It has the ability to work with multiple workers simultaneously and will behave asynchronously when possible.

Usage

```
usage: youtube_scraper.py [-h] [-q,--query QUERY]
                          [-t, --num_threads NUM_THREADS]
                          [-n, --num_videos NUM_VIDS] [-v, --verbose]
                          [-r, --rebuild] [-b, --backup_every BACKUP_EVERY]
                          [--open] [--categorize] [--convert] [--clean]
                          [--upload]

Perform a video search and sorts the results into the proper.

optional arguments:
  -h, --help            show this help message and exit
  -q,--query QUERY      Search term to use, separated by comma
  -t, --num_threads NUM_THREADS
                        Number of concurrent Threads
  -n, --num_videos NUM_VIDS
                        Number of videos for each keyword that will be
                        downloaded
  -v, --verbose         Verbose output
  -r, --rebuild         Rebuild the search cache?
  -b, --backup_every BACKUP_EVERY
                        Backup to CSV every N number of videos
  --open                Open every new video on download
  --categorize          Categorize into Faces, Conversation, multimodal, and
                        trash.
  --convert             Convert videos.
  --clean               Cleans the downloads directory
  --upload              Uploads to S3
```

Example Command: `python2 youtube_scraper.py -v -q "Test1, test2, test3" -n 100 --categorize --upload --convert -t 5 -b 2`

Docker Image

```
COMMAND='-q "interview" -n 10'
sudo docker pull reichenbachian/youtube_dl
IMAGE_ID=`sudo docker images | sed -n 2p | awk '{print }'`
docker run -e AWS_ACCESS_KEY_ID='[Your access key]' -e AWS_SECRET_ACCESS_KEY='[your secret key]' -it $IMAGE_ID /bin/sh -c 'cd youtube_video; python /usr/src/app/youtube_video/youtube_scraper.py -q "interview, colbert" -n 10 -b 10 --categorize --upload --convert'
```

Manual Install

Prerequisites

1. Create a virtualenv and install requirements.
2. Install the requirements.txt
3. pip install cv2
4. brew install pkg-config
5. Compile Movement Detector using swig like so... Generic:

```
g++ -c MovementDetect.cpp $(pkg-config --libs opencv) -o MovementDetect.o
swig -I{Location of opencv library} -I{Location of opencv headers} -python -c++
MovementDetect.i
```

Example:

```
g++ -std=c++0x -c MovementDetect.cpp $(pkg-config --libs opencv) -o
MovementDetect.o
swig -I$(pwd)/opencv-swig/lib/ -I/usr/local/Cellar/opencv/HEAD-01e34b6/include/
-python -c++ MovementDetect.i
```

Generic:

```
g++ -shared -fpic MovementDetect_wrap.cxx MovementDetect.o -I{Virtualenv python
header location} -L{Virtualenv python library location} -L{opencv library location}
-lopencv_calib3d -lopencv_contrib -lopencv_core -lopencv_features2d -lopencv_flann
-lopencv_gpu -lopencv_highgui -lopencv_imgproc -lopencv_legacy -lopencv_ml
-lopencv_nonfree -lopencv_objdetect -lopencv_ocl -lopencv_photo -lopencv_stitching
-lopencv_superres -lopencv_ts -lopencv_video -lopencv_videostab [if mac: -undefined
dynamic_lookup; if ubuntu: -lpython2.7] -o _MovementDetect.so
```

Example:

```
g++ -shared -fpic MovementDetect_wrap.cxx MovementDetect.o
-I/Users/localhost/Desktop/Projects/Working/Affectiva/affEnv/include/python2.7
-L/Users/localhost/Desktop/Projects/Working/Affectiva/affEnv/lib/python2.7
-L/usr/local/Cellar/opencv/HEAD-01e34b6/lib -lopencv_calib3d -lopencv_contrib
-lopencv_core -lopencv_features2d -lopencv_flann -lopencv_gpu -lopencv_highgui
-lopencv_imgproc -lopencv_legacy -lopencv_ml -lopencv_nonfree -lopencv_objdetect
-lopencv_ocl -lopencv_photo -lopencv_stitching -lopencv_superres -lopencv_ts
-lopencv_video -lopencv_videostab -undefined dynamic_lookup -o _MovementDetect.so
```

1. Create ~/.aws/config file with contents as follows (Note: You might need a .s3config too.)

```
[default]
aws_access_key_id = [Put access key here]
aws_secret_access_key = [Put secret key here]
```

```
[default]
region=us-east-1
```

1. Create a worker file. In the same directory as youtube_scraper.py, create a Worker_Key.key file. It should contain the following. General

```
Unique id
Whether it is the master computer
```

Example

```
b37829d7-fec3-4dcc-b76c-7f801c8acb32
False
```