Parallel massive dataset cleaning

Jianmeng Yu

Supervised by:
Bob Fisher

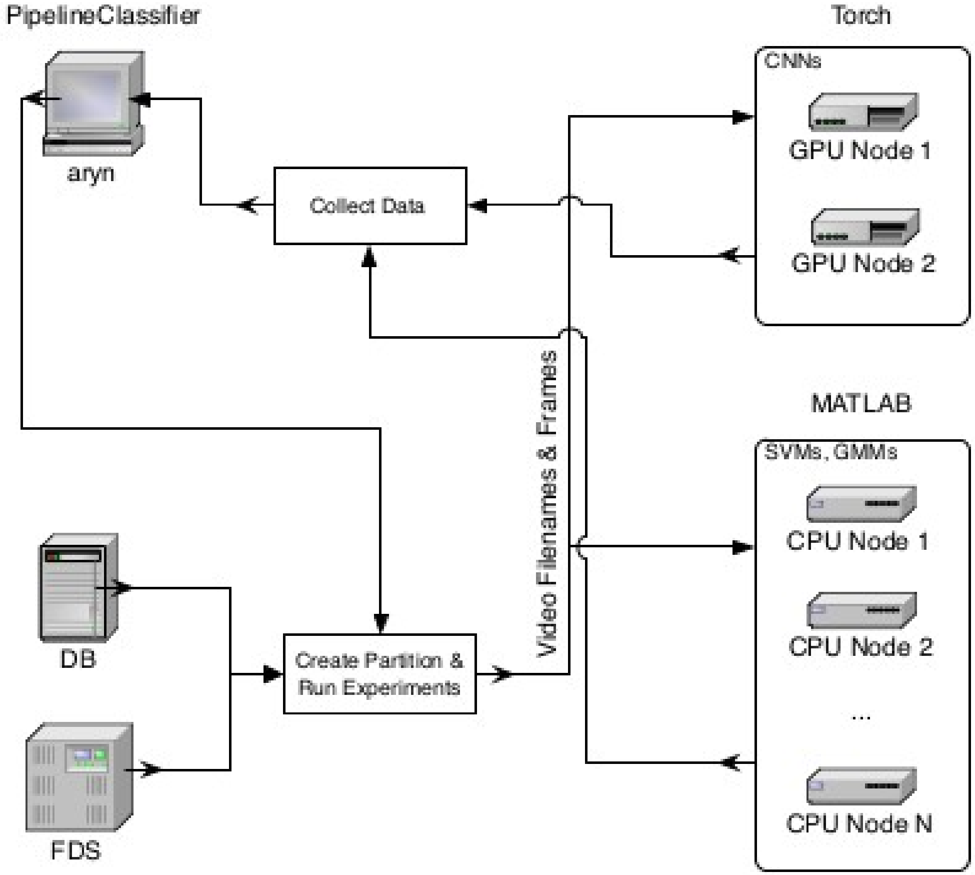
Project Recap

- What is the project about?
  - Translate the existing decision algorithms into Python.
  - Develop a framework to apply the algorithm at a massive parallel scale.
  - Apply the cleaning algorithm (task farming) and reconstruct the video files.
- Why parallel?
  - The original code take 2000 hours to finish.
  - There are about 839,000,000 frames in 396,000 video clips.
  - And 1,446,000,000 records in a 500 GB sql file.

Current Progress – Code Translation

Original Framework by Matthew

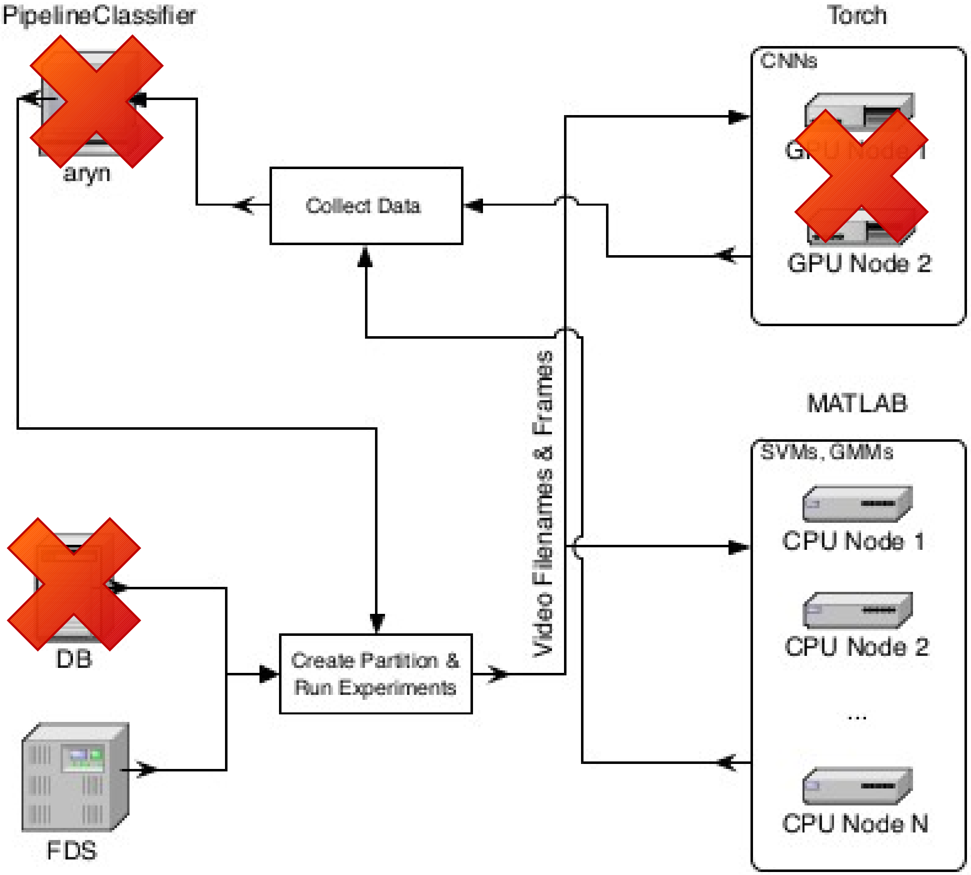
- Python Script with SQL to extract frame infos and store them on AFS.
- Partitioned data feed into trained classifiers using MATLAB interface.
- The output vectors are collected and processd into final classification result.



Current Progress – Code Translation

Difficulties

- The framework isn't fully implemented.
- The 500GB SQL is too large for the university servers.
- DICE machines have Intel HD 530 GPU, does not support CUDA.
- No documentation, the code checks if it's running on 'Aryn' instead of pulling libraries used.



Current Progress – Code Translation

Solution for not having a SQL server

- The 500 GB SQL file is the 'dump' file of the original database.
  - Including irrelevent tables like login logs.
- Instead of loading the entire SQL file into server and run 400,000 queries, A python script pipeline is used to extract and partition the required data from SQL.
- The extraction pipeline take about 300 hour to finish. After optimization it only took about 12 hours.
  - Most of the time is wasted on repeatedly open/close files.
- Loading a video and corresponding extracted SQL file in Jupyter Notebook took about 0.5 seconds on average now.

Current Progress – Code Translation

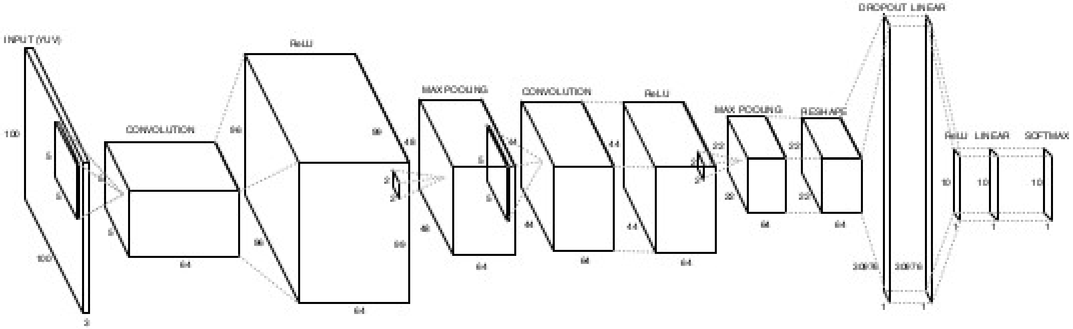
Solution for not having a NVIDIA GPU on DICE machine.

- The original CNN classification uses Torch, a lua library.
  - It uses NVIDIA CUDA backend which DICE doen't support.
- The CNN classifier will be translated into python using the TensorFlow library.
- Unfortunately I have no experience on Neural Networks, I am currently working on TensorFlow tutorials online.
- The CNN is already trained, only need to load weight.

PYTORCH

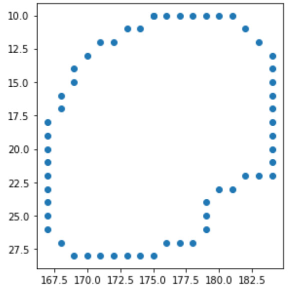
VS

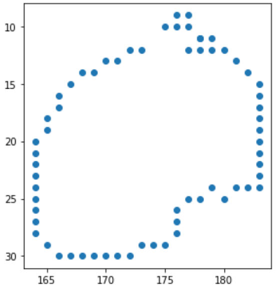
TensorFlow



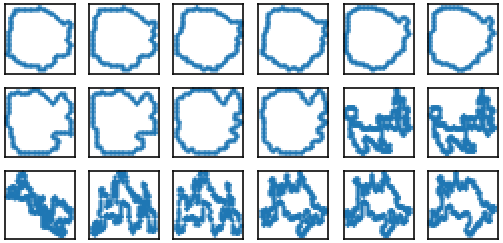
Current Progress – Preprocessing

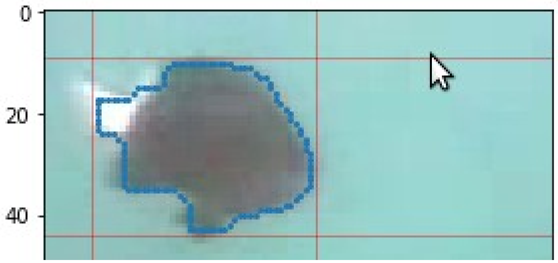
- The classification need a 'bb_cc' field from the sql, which means “bounding box chain code”, and it's stored in raw binary form.
- First attempt on extracting information from bb_cc yields a 'corrupted' bounding box.
  - Image on left is original contour
  - Image on right is corrupted





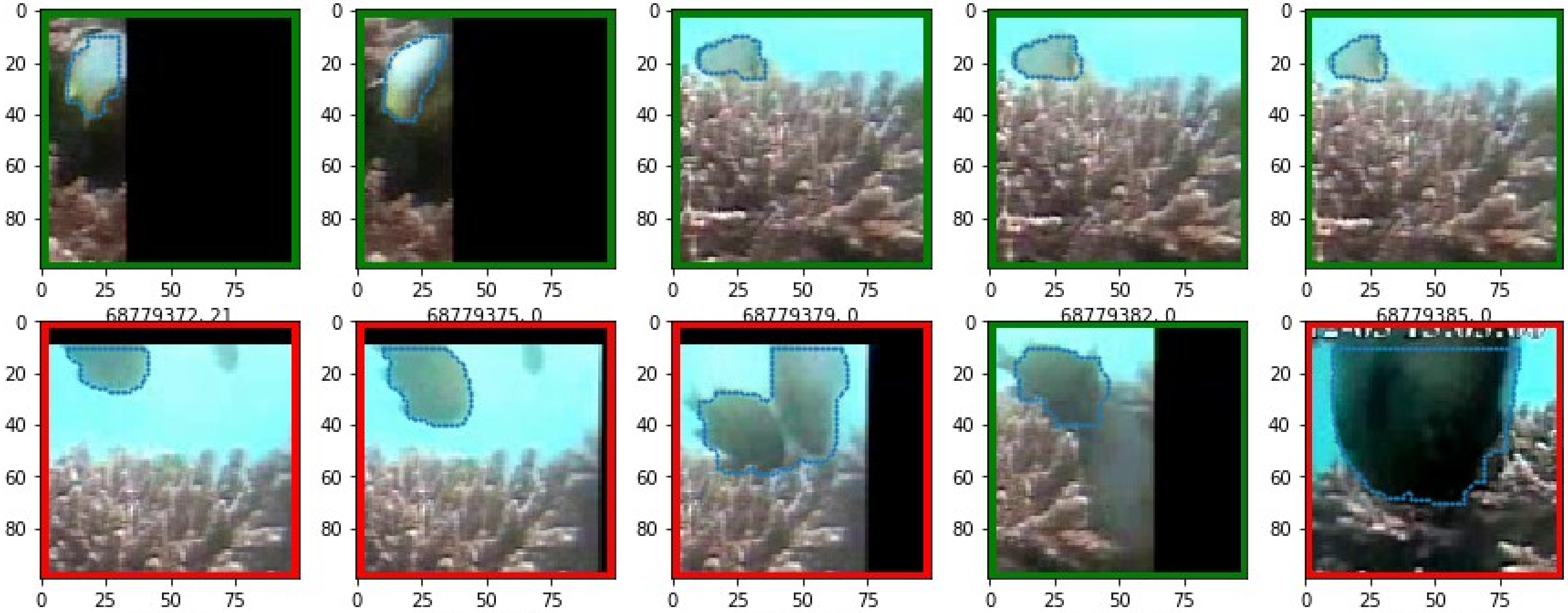
- By looking into the binary values of the data, the error is actually caused by the encoding of the binary files in sql.
  - Null bytes: “0000 0000” becomes \0: “0101 1100 0011 0000”





Current Progress – First Stage Cleaning

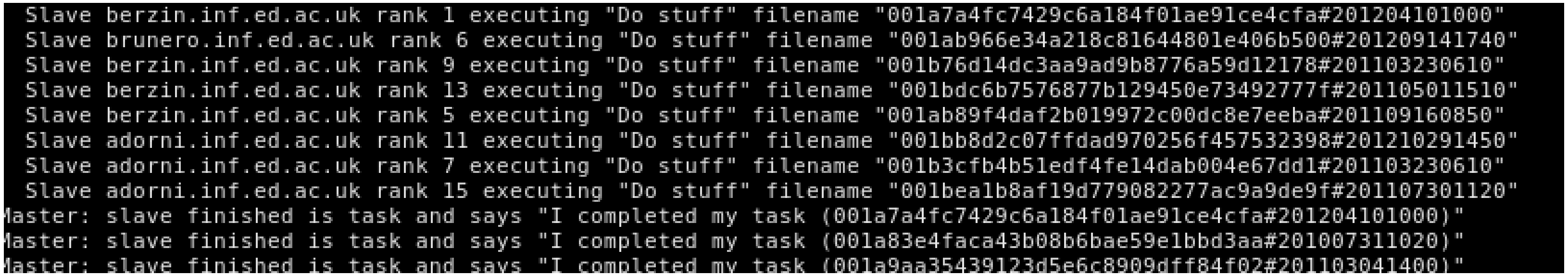
- The first stage of the cleaning uses a Frame Edge Indicator Function (FEIF)
  - It checks number of points on the contour touches the edge.
  - It removes detections with high edge point count.
- This effectively removes the 'partial fish' false positives.



Current Progress – Task Distribution

“MPI for Python” package

- Allows message passing between python process.
- Assigning the process with the highest ID as 'master' process, and all other process as 'slave', The master assigns task to idle slave whenever they are available.
- For testing, a dummy mpi4py program is used to extract the 'ID' field from summary files, using 4 different DICE machine.



```
 Slave berzin.inf.ed.ac.uk rank 1 executing "Do stuff" filename "001a7a4fc7429c6a184f01ae91ce4cfa#201204101000"
 Slave brunero.inf.ed.ac.uk rank 6 executing "Do stuff" filename "001ab966e34a218c81644801e406b500#201209141740"
 Slave berzin.inf.ed.ac.uk rank 9 executing "Do stuff" filename "001b76d14dc3aa9ad9b8776a59d12178#201103230610"
 Slave berzin.inf.ed.ac.uk rank 13 executing "Do stuff" filename "001bdc6b7576877b129450e73492777f#201105011510"
 Slave berzin.inf.ed.ac.uk rank 5 executing "Do stuff" filename "001ab89f4daf2b019972c00dc8e7eeba#201109160850"
 Slave adorni.inf.ed.ac.uk rank 11 executing "Do stuff" filename "001bb8d2c07ffdad970256f457532398#201210291450"
 Slave adorni.inf.ed.ac.uk rank 7 executing "Do stuff" filename "001b3cfb4b51edf4fe14dab004e67dd1#201103230610"
 Slave adorni.inf.ed.ac.uk rank 15 executing "Do stuff" filename "001bea1b8af19d779082277ac9a9de9f#201107301120"
Master: slave finished is task and says "I completed my task (001a7a4fc7429c6a184f01ae91ce4cfa#201204101000)"
Master: slave finished is task and says "I completed my task (001a83e4faca43b08b6bae59e1bbd3aa#201007311020)"
Master: slave finished is task and says "I completed my task (001a9aa35439123d5e6c8909dff84f02#201103041400)"
```

- By printing the Processor ID, it shows that the task is actually distributed to different machines.

Current Progress – Task Distribution

How do I know which machines are available?

- I recorded the location and name of every DICE machine in labs of Appleton Tower, floor 4-6. (about 200 machines)
- A "marauder's map" bash script to ping every machine on the list. If a connection is made, then ssh into it to see who's using the machine, if I am the only user, then mark it as available.
- Ideally, most of the DICE machines will be available during Christmas.

Future Tasks

- Translate the feature extraction algorithms and the 3 remaining Classifier: SVM, GMM, and CNN.
- Put the classification code into the dummy task distributor program.
- Actual cleaning of the data, rebuild the AVI files for the project.
- Create more test dataset to evaluate of the quality of cleaning.

- Possibly implement the unfinished divide-and-conquer classifier in original pipeline.

Timetabling

- December Exam Period / Christmas
  - Finish the translation of the code.
  - Process the data using the idle DICE machines.
- January and onwards.
  - Evaluate the cleaning, and start working on reports.