ECSE-552 Final Project Organization Proposal

03/01/2022

Graham Smith

3 Larger Phases

- **Phase 1** – Layout Architectural Framework
  - Further phases depend on this
  - Higher coupling/dependencies between components
- **Phase 2** – Initial Experiments
  - Coarsely explore relationships between data/model
- **Phase 3** – Refinement of experiments/finalization
- Ideally asynchronous execution of experiments for phases 2/3
  - People can run experiments in parallel using common code
  - Allows for development given everyone’s different schedules
  - Experimental results can be posted to Slack for discussion/analysis

Phase 1 – Architectural Framework

- **Goal**: Determine interfaces/classes between constituent software components
- Write software to implement:
  - Model
  - Feature extraction/data preparation
  - Train/Test/Benchmark Model (i.e. execution time/accuracy)
- Ideally software parametrized in a way to facilitate experiments
  - Number of layers in model is an argument as opposed to hard-coded
  - Spectrogram parameter selection isolated to one component

Phase 2 – Initial Experimentation

- **Goal**: Identify limitations based on model/data/computational resources and rough parameter settings (i.e. 50 vs 500 layers)
- What are bottlenecks in execution?
  - Effect of network connection between Google Colab and data
  - How much data can we process in what amount of time?
  - How complex of a model can we train in 12 hours? 24 hours?
- Measure execution times
- Reliability of data
  - How many languages can we feasibly train for identification?
  - How much data is usable based on non-standardized recording process?

Phase 3 – Refined Experimentation

- **Goal**: Fine tune relationships between data/model (i.e. 50 vs. 60 layers)
- Mitigate data cleanliness issues if need be
- Optimize performance on languages model can discriminate
- Explore reasons why we excel/fail at particular languages
- Measure performance against other models

Network Issue w/ Data Stoarge

- Google Colab storage doesn’t persist → data needs to be uploaded each time
- Solution #1 – Store data on Gdrive and access Gdrive while training
  - Quota limits on per-user and per-file operation count and bandwidth quotas
  - Creates bottleneck where training loop could be stalled/waiting on data from network connection between Google Colab and Gdrive
- Better solution – Zip up data and transfer it all before training begins
  - More overhead upfront in terms of upload time
  - Data stored on drives associated with VM instance
  - Removes network connection from bottleneck
- Documented problem in Google Colab FAQ