PRDv1: LogMeIn Team

Contents

Project and Team Info

Project Info

Team Name	Coast Masters
Project Title	Presentation Trainer

Team Members

Name	Email
Isaiah Egan	ije@umail.ucsb.edu
Zachary Feinn (team lead)	zpfeinn@umail.ucsb.edu
Ryan Allen	rmallensb@gmail.com
Ryan Kemper	ryankemper@umail.ucsb.edu
Josue Montenegro	josuemy@gmail.com

Intro

Problem

Giving presentations is an essential skill in life, but high quality training is not available for everyone. The solution requires a shift in the way presentation skills are taught, as well as feedback for those who are preparing presentations. Therefore, the solution is to provide accessible and effective presentation training, as well as a feedback system for practitioners.

Innovation

We combine video, audio, and text analytics to give the user feedback on their presentation. This requires feature engineering--determining the important markers of a 'good' presentation.

We accomplish this by examining powerful historical presentations (e.g. Steve Jobs' launch announcements) and understanding what makes these presentations effective. We then translate this understanding into a set of measurable features which can be applied to any presentation. In this way we innovate both in data science and psychology/performance by understanding from chosen data what makes a presentation 'good' and translating those factors into digital terms.

We also innovate by constructing software which can measure these features from a video of a presentation. As these features are not yet determined, such software necessarily does not exist, so ours will be the first.

Finally, we innovate by taking presentation training out of the analog world. Currently, one can receive training only from communications classes, Toast Masters' meetings, and similar live or hybrid live/digital options. We innovate by creating a purely digital option that is accessible by the web. Therefore, anyone with access to the internet can learn to be an effective presenter.

Team Goals/Objectives

Our goal is a web based platform where users can practice presentations and receive actionable feedback. Our system should be a tool for users to systematically practice their presentation skills.

Stretch goals include persistent user data that allows users to track their performance and improvements over time. Additionally, we would like to include training modules where users are asked to perform a historical presentation, and their performance is then compared with that of the original presenter.

Soft goals include effective modularization of our software requirements, such that sub-groups within our team can work independently. We accomplish this by considering four main areas of development: web services, text analytics, audio analytics, and video analytics.

Background

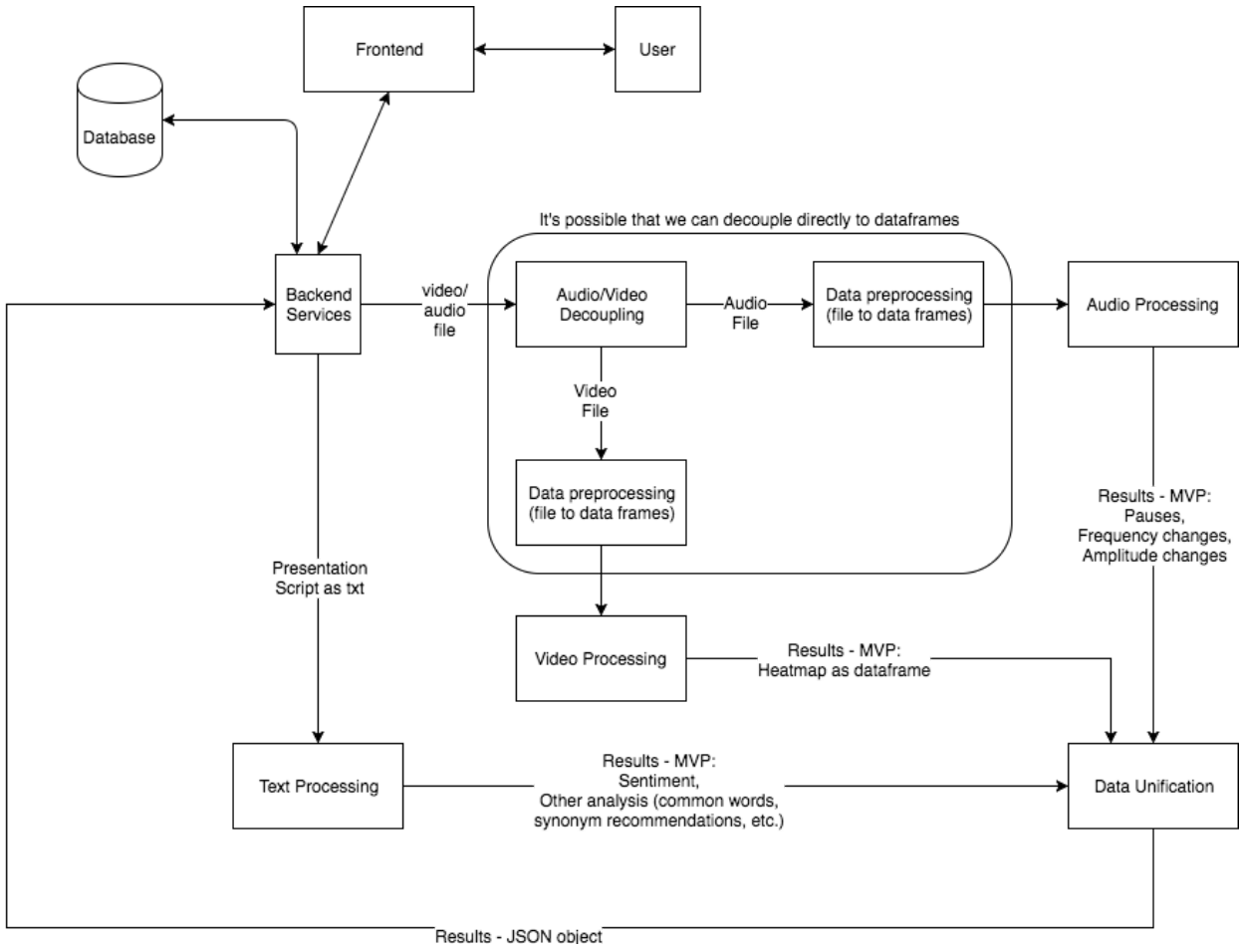
LogMeIn is a subsidiary of Citrix, a company which seeks to improve collaboration through web based software. Webcam presentations are essential in an increasingly online business environment. Therefore, our webcam training method fits nicely with LogMeIn's mission and vision for the future.

Assumptions

Our product will be targeted toward American English speakers, due to constraints on what speech-to-text libraries can accomplish. Additionally, we assume that certain factors of a presentation are universally considered 'good' in this market. For example, we assume that a consistent rate of speech is better than a highly varied rate of speech.

System Architecture Overview

Our system architecture reflects our goal of modularization. There are several main components, web services, A/V decoupling, audio processing, text processing, and video processing.



Requirements

User Stories

Below we include a sample of ten user stories. Some of these relate to larger features in our noted modules which will be included in our MVP. Those user stories are annotated with relevant Github commits and information. As with our previous work, user stories are divided up by the relevant module.

Audio Processing

- As a user, I would like to see how much I fluctuate in pitch while I speak, so I can see if I change pitch too much or too little
  Deliverable: Show a univariate, 2-D graph of their pitch during the speech.

- As a user, I would like to practice my rate of speech during a presentation so I can see if I am talking too fast.
  Deliverable: Show the current rate of speech at any given time during the presentation.

- As a user, I would like to see my average wpm for the entirety of my speech so that I can know if I am speaking too quickly or too slowly
  Deliverable: A visible metric (their average WPM) for the entire speech

- As a user, I would like to see how loudly I speak during a presentation so that I am well understood.
  Deliverable: Show a univariate, 2-D graph of their volume during the speech

- As a user, I would like to see my rate of speech during different parts of the presentation
  Deliverable: Show the positions of each word in their speech, and show the WPM rate within each cluster of speech

Relevant Github commits for above user story

Commit Name	Description
Simple onset time detection	Explored Librosa's onset detection as an initial attempt for speech rates
Added some JSON exports, calculates the wpm for the entire audiofile	Calculates WPM for audio file. Allows for exportability of data to web interface in JSON format
Simple plot of KDE	Began working on one half of 1-D clustering algorithms using SciKit-Learn KDE modules. Plot output of module using matplotlib to ensure module is working as desired

Video Processing:

- As a user, I want to know how well I use space during my presentation, so that I can verify I'm not standing in one place for too long.
  Deliverable: A bar graph indicating user's movement in each area of the video frame.

Relevant Github commits for above user story

Commit name	Description
Add openCV samples I've played with	Explored OpenCV to learn how to track user in video
Recognize face expression and get coordinates	Record an x coord stamp of the user's face, every 4 sec.
Divide video window in three sections	Split the video frame in 3 areas so that I can sort my x coords. by area.
Succesfully combined openCV and GoogleAPI to grab x coords	Use openCV to track faces, and if that fails use Google's APIs to recover face coords. This will minimize API latency and costs.
Plot graph of movement distr.	After sorting coordinates per area, I plot a graph that shows the user's space usage throughout the whole presentation.

Text Processing:

- As a user I would like to see the frequency of the words used in my speech so that I can easily see if I am being too repetitive.
  Deliverable: A word cloud of the top 10 most used words and their word count (disregarding words such as 'the' 'and' 'is' etc).

- As a user I would like to see the semantics of my speech so that I can make sure my tone is correct for my intended audience.
  Deliverable: A radio graph representing the social and emotional tones of the text.

Relevant Github commits for above user story

Commit name	Description
Basic NLTK Sentiment Analysis	Explored the possibility of using NLTK for our sentiment analysis
Added a Frequency Analyzer to Present Commonly Used Words	This is going to be used to show the user which words he/she is over using and provide the backbone for providing synonyms
Updated watson and included more test files	We decided to not use NLTK and to use IBM Watson's tone analyzer to perform sentiment analysis
Incorporated watson analysis and word frequency into classes	The initial backbone for the text analysis class has been made and it includes methods to create a tone analysis as well as getter methods to get desired outputs (json files, dictionaries)

Web Services:

- As a user, I would like to be able to upload my presentation to a web interface so that I can get feedback on my performance.
  Deliverable: an upload webpage.

- As a user, I would like to be able to view results on a single page so that I can easily make decisions about how I can improve.
  Deliverable: a results webpage.

Appendices

Technologies Employed

- Audio Processing
    - Librosa
    - Watson API
- Video Processing
    - OpenCV
- Text Processing
    - Watson API
    - NLTK
- Web Services
    - Flask (Python)
    - Ffmpeg