

DigiClips Media Search Engine

sddec21-06

**Tyler Johnson, Samuel
Massey, Max Van de Wille,
Max Wilson**

Advisor: Ashfaq Khokar

Our Client

- DigiClips, Inc., a Colorado based media content analysis company
- Contacts:
 - Chairman: Bob Shapiro
 - Senior Software Engineer: Henry Bremers
- Constantly recording television news and radio in the Denver Metropolitan Area
- Provides a search engine so that recent news and radio broadcasts can be searched

Project Plan

- Problem Statement
- Functional Requirements
- Non-Functional Requirements
- Constraints & Consideration
- Market Survey
- Potential Risk & Mitigation
- Resource & Cost Estimates
- Project Milestones & Schedule

Project Plan

Problem Statement

- So far, only closed caption data is extracted alongside television recordings
- Closed captions data often misses words or phrases spoken within the broadcast
- Additionally, there is visible text shown on screen that is also not transcribed
- No closed captions or transcribed data is extracted alongside radio recordings
- Missing/untranscribed data leaves gaps in the searchable content of a broadcast

Project Plan

Functional Requirements

- Speech-to-text must convert mono and stereo audio recordings into plain text
- Video-to-text must detect multiple fonts/styles of text on bottom half of the recording frames
- All system results must match DigiClips database schemas
- All system results must have proper grammar and spelling
- All system errors must be recorded in the DigiClips error database

Project Plan

Non-Functional Requirements

- System will be built without using any costly API/cloud resources
- System will be built with documentation to explain usage
- System should scale with increased quantity of data
- System should reliably output accurate data in a reasonable amount of time

Project Plan

Constraints & Considerations

Constraints:

- Cannot utilize certain paid APIs for speech-to-text or optical character recognition
- Developed program must be able to run on a relatively underpowered computer
- Must run quickly to query data within 24 hours of recording

Considerations:

- The output text must be indexed by timestamp so that it can be linked to a video segment
- Assuming video input will be high enough quality for accurate processing

Project Plan

Market Survey

- Very few existing implementations of speech-to-text and video-to-text on television
- Most similar applications differ in key areas
 - Performing processing on a live feed
 - Grammar and spelling is not a concern for output
 - Output is not formatted to be searchable
 - Usages are not time-sensitive

Project Plan

Potential Risks & Mitigation

Risk	Probability	Mitigation
Speech-to-text processing inaccuracies	0.2	Extensively research speech recognition technology
Video-to-text service processing	0.5	Researching video OCR strategies and code optimization
Word misidentification	0.5	Testing throughout development
System Integration	0.5	Substantial planning ahead of time
Database connection	0.4	Communication with DigiClips

Project Plan

Resource & Cost Estimates

Resources:

- No additional resources required to complete project

Cost:

- This project will not incur any costs

Project Plan

Project Milestones & Schedule

Milestones:

- Complete speech-to-text system
- Complete video-to-text system
- Integrate the two systems into one
- Integrate with DigiClips database

Schedule:

- April 2021 - October 2021
- April 2021 - October 2021
- October 2021 - November 2021
- November 2021 - December 2021

Project Plan

Evaluation Criteria

- Achieve 80% accuracy on speech recognition and 95% coverage with microservice unit testing.
- Achieve 70% accuracy on video text recognition and 95% coverage with microservice unit testing.
- Process speech-to-text for a video file within 75% of the file's length.
- Process video-to-text for a video file within the length of the file.

System Design

- Functional Decomposition
- Detailed Design
- Hardware & Software Platforms
- Test Plan
- Prototype Implementations

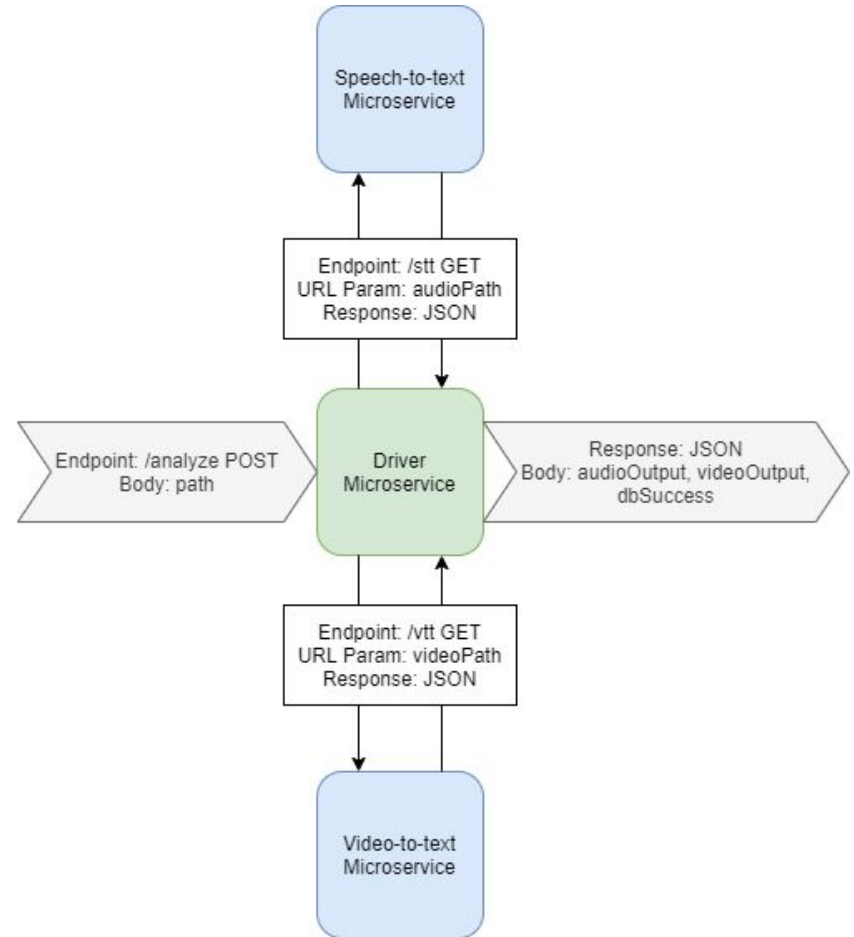
Functional Decomposition

- Detect speech in audio file and output transcript
 - Split input into chunks for individual processing
 - Process output for grammar and punctuation
 - Index output with timestamps
- Detect words and phrases shown on screen and output
 - Splitting video file into individual frames
 - Image pre-processing
 - Index text output with timestamps

System Design

Detailed Design - Overall

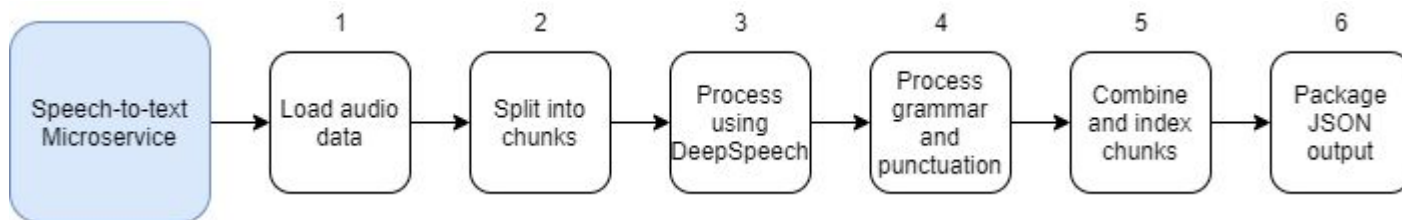
- Microservices
 - Driver Microservice
 - Speech-to-text Microservice
 - Video-to-text Microservice



System Design

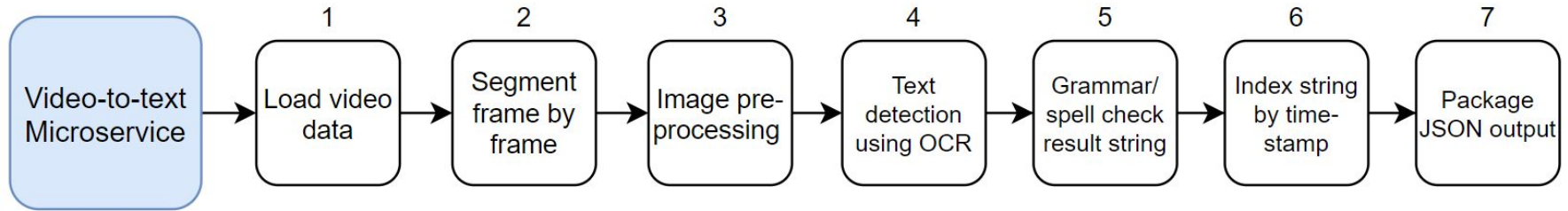
Detailed Design - Speech-to-text

- Load file into application
- Split into chunks on silence
- Run chunks through DeepSpeech model
- Add grammar and punctuation
- Combine chunks
- Output



System Design

Detailed Design - Video-to-text



- Perform pre-processing and text detection for every n frames
 - n is dependent on frames per second (fps) of input video
- Pre-processing includes cropping, thresholding, Gaussian noise reduction
- Timestamp of frame (in seconds) can be found by calculating frame number / fps

System Design

Hardware & Software Platforms

Programming Language:

- Python

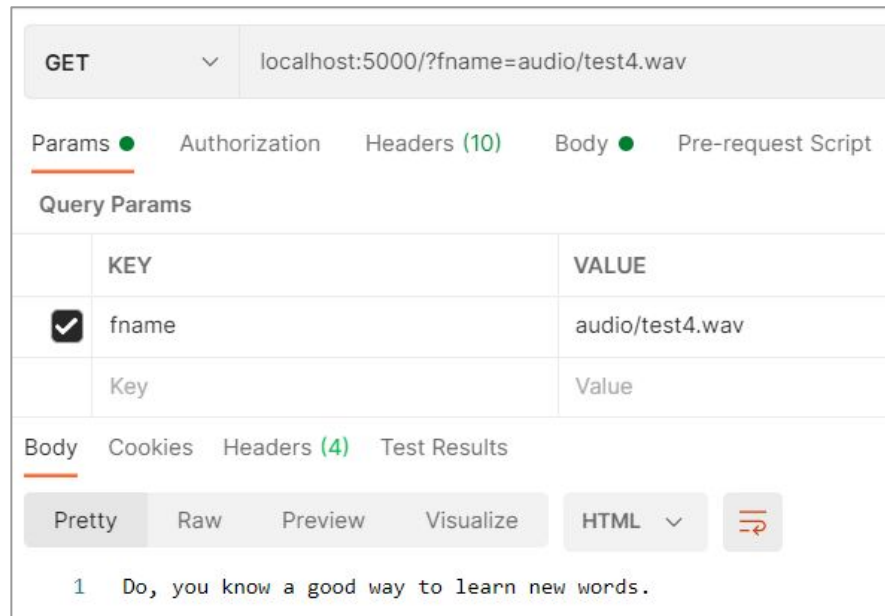
Frameworks and Libraries:

- Flask
- Tesseract
- OpenCV
- DeepSpeech
- pydub
- punctuator2
- Language-tool-python

System Design

Test Plan

- Unit Testing
 - pytest and unittest
 - Test smaller individual functions
 - Regression testing
- Interface Testing
 - Endpoint-based testing
 - HTTP requests
- Acceptance Testing
 - Working with client to ensure functionality
- Results
 - Use testing to validate features during development cycle



GET localhost:5000/?fname=audio/test4.wav

Params ● Authorization Headers (10) Body ● Pre-request Script

Query Params

	KEY	VALUE
<input checked="" type="checkbox"/>	fname	audio/test4.wav
	Key	Value

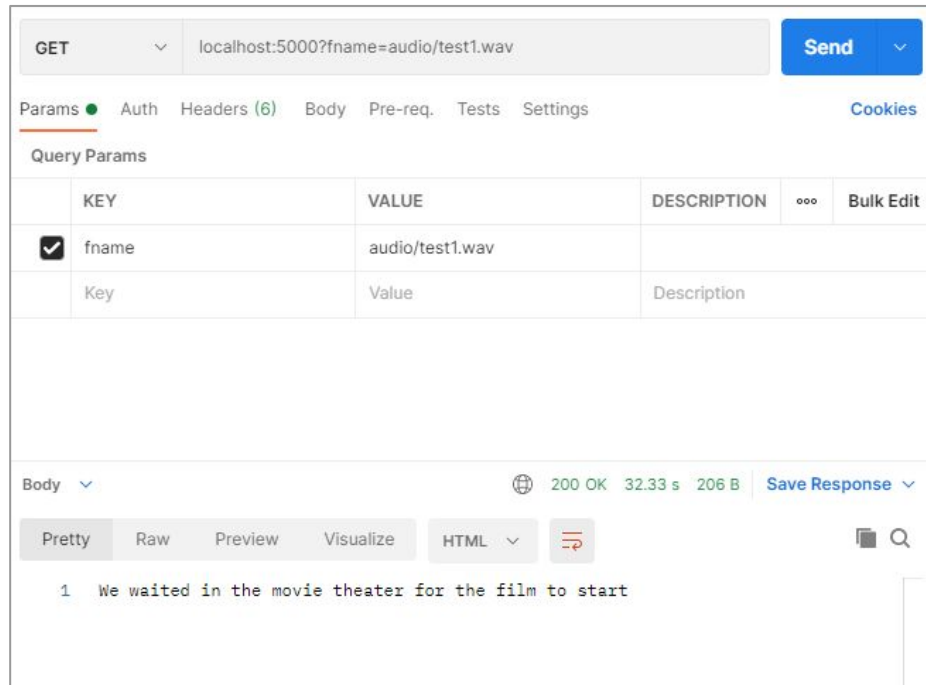
Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize HTML ▾

1 Do, you know a good way to learn new words.

Prototype Implementations - Speech-to-text

- Implemented simple flask web service
- API accepts a file path
- Outputs speech-to-text result



The screenshot shows a Postman interface for a GET request to `localhost:5000?fname=audio/test1.wav`. The request is successful, returning a 200 OK status with a response time of 32.33 seconds and a body size of 206 B. The response body is displayed in the 'Pretty' view as a single line of text: `1 We waited in the movie theater for the film to start`.

KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/> fname	audio/test1.wav			
Key	Value	Description		

System Design

Prototype Implementations - Video-to-text

- Implemented simple data pipeline for prototyping
- Program accepts a file path to video
- Outputs string captured by processed frame with timestamp index



Example output showing locations of identified text

```
0h:0m:13.65s - <& Couns Dr JENNIFER ASHTON ABC NEWS Chief Medical Correspondent
0h:0m:13.68s - <& Couns Dr JENNIFER ASHTON ABC NEWS Chief Medical Correspondent
0h:0m:13.71s - <& Couns Dr JENNIFER ASHTON ABC NEWS Chief Medical Correspondent
0h:0m:13.75s WHAT YOU SNEWS Couns Dr JENNIFER ASHTON ABC NEWS Chief Medical Corre
0h:0m:13.78s SN ye) Dr JENNIFER ASHTON ABC NEWS Chief Medical Correspondent
```

String output for previous figure with formatted timestamp

Conclusion

- Current Project Status
- Task Responsibilities & Contributions
- Next Semester Plan

Conclusion

Current Project Status

- Speech-to-text
 - Prototype functional
 - Basic Flask app
 - Processes audio file and returns text with grammar checking and punctuation
- Video-to-text
 - Early prototype
 - Tesseract running on video frames
 - Needs fine tuning for pre-processing, Tesseract parameters

Conclusion

Task Responsibilities & Contributions

- Tyler Johnson
 - Responsible for planning and implementing testing on project
- Samuel Massey
 - Responsible for assignment planning and research/work on speech-to-text
- Max Van de Wille
 - Responsible for documenting architecture changes and working on video-to-text
- Maxwell Wilson
 - Responsible as primary point of contact with client and working on speech-to-text

Conclusion

Next Semester Plan

- Complete speech-to-text microservice
- Complete video-to-text microservice
- Implement created elements into one product
 - Completing driver microservice
- Combine created project with DigiClips system
- Ongoing testing finalized via acceptance testing with client



Questions?

Appendix

- Gantt-Style Schedule

