Automatic Illustration of Text via Multimodal Interaction

Stergious Aji - 2546916A

December 15, 2022

1 Status report

1.1 Proposal

1.1.1 Motivation

Video production is a very time consuming process and many parts of it can be easily automated like the making of illustrated music lyric videos, educational content or podcasts. This can greatly improve accessibility, not to mention, provide better mediums for teaching and learning. An automatic videographic tool would aim to sequence relevant images, that are based on the textual content present in the audio, in a timely manner. However, at the moment, there lacks a generally accepted, objective procedure at evaluating the results of such a system. User surveys can provide widely subjective and differing opinions on the relevancy of the images and their timeliness. Additionally, each time the system is changed, new user surveys must be conducted to re-evaluate it. For this reason, the system's output should be gauged against an immutable ground truth which would yield standardised metrics and can therefore be compared and repeated.

1.1.2 Aims

The aims of this project will be to develop a software framework to make user corroborated ground truths of automated videographic content. The user will have to select images that they believe are most relevant to the given audio chunks, out of a set that the system retrieves. This can then be objectively and systematically evaluated against other automated videography systems on the same audio sources. This evaluation aims to provide standardised performance metrics for any general automated videography tool.

1.2 Progress

- Language and Web framework chosen: System will be built using Python and the Django Web Framework.
- Main videography generation pipeline implemented with pytube API to take YouTube URLs as input.
- Pipeline made more automated by using Shazam API to recognise songs and artist names and MusixMatch API to retrieve its corresponding synced lyrics.
- Dataset to retrieve text and images from selected: Wikipedia-based Image-Text dataset (WIT).
- Background research conducted on solely text indexing and querying.
- Text indexing and querying implemented using PyTerrier.
- Background research conducted on solely image vectorising using its Wikipedia caption.
- Started development of the prototype Django Web Application.

1.3 Problems and risks

1.3.1 Problems

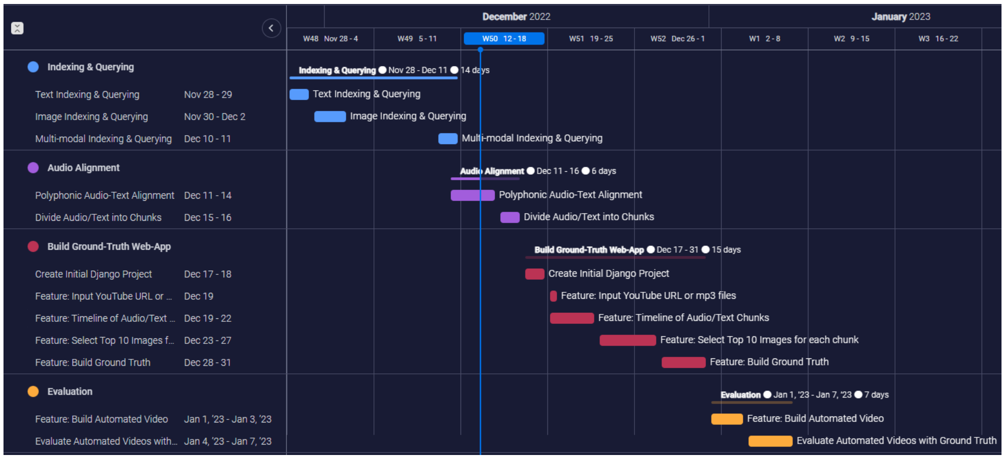
- Initial compatibility issues experienced with the JDK version used by PyTerrier and the one installed in my computer. Luckily once problem was identified, it was easily fixed.
- Researching an efficient way of indexing images and text in a multimodal space. Currently researching into CLIP to solve problem.
- Implemented Shazam API is not fully robust as in very rare cases, it fails to recognise well-known songs. This can be solved by running the search again, although, need to look into a better solution.
- Need to find a way to systematically download the images from the WIT dataset which currently only contain image URLs.
- Solely using text to index and query images do not provide very relevant images. Need to experiment with more vectorisers as currently I am only using TF-IDF.

1.3.2 Risks

- Sufficient space is needed to store the static collection of images from the large WIT dataset.
  Mitigation: Systematically collect a subset of images as there are many that can be considered useless for this application.
- The system pipeline still relies on external APIs to download videos, recognise songs and retrieve synced lyrical data which can hinder the system if any one fails.
  Mitigation: Instead of retrieving audio from downloaded YouTube videos, an option to upload MP3 files should be implemented as well.
- At the moment, it is not clear how the evaluation metrics will be calculated. No clear mitigation currently but will be looked into, in the future.

1.4 Plan

The following Gantt Chart in Figure 1 shows my plan to develop the software for my project during the Winter weeks. This work will then continue through Semester 2 and the deliverables are outlined below.
Winter



● Indexing & Querying	
Text Indexing & Querying	Nov 28 - 29
Image Indexing & Querying	Nov 30 - Dec 2
Multi-modal Indexing & Querying	Dec 10 - 11
● Audio Alignment	
Polyphonic Audio-Text Alignment	Dec 11 - 14
Divide Audio/Text into Chunks	Dec 15 - 16
● Build Ground-Truth Web-App	
Create Initial Django Project	Dec 17 - 18
Feature: Input YouTube URL or ...	Dec 19
Feature: Timeline of Audio/Text ...	Dec 19 - 22
Feature: Select Top 10 Images f...	Dec 23 - 27
Feature: Build Ground Truth	Dec 28 - 31
● Evaluation	
Feature: Build Automated Video	Jan 1, '23 - Jan 3, '23
Evaluate Automated Videos with...	Jan 4, '23 - Jan 7, '23

December 2022
January 2023
W48 Nov 28 - 4
W49 5 - 11
W50 12 - 18
W51 19 - 25
W52 Dec 26 - 1
W1 2 - 8
W2 9 - 15
W3 16 - 22
Indexing & Querying ● Nov 28 - Dec 11 ● 14 days
Text Indexing & Querying
Image Indexing & Querying
Multi-modal Indexing & Querying
Audio Alignment ● Dec 11 - 16 ● 6 days
Polyphonic Audio-Text Alignment
Divide Audio/Text into Chunks
Build Ground-Truth Web-App ● Dec 17 - 31 ● 15 days
Create Initial Django Project
Feature: Input YouTube URL or mp3 files
Feature: Timeline of Audio/Text Chunks
Feature: Select Top 10 Images for each chunk
Feature: Build Ground Truth
Evaluation ● Jan 1, '23 - Jan 7, '23 ● 7 days
Feature: Build Automated Video
Evaluate Automated Videos with Ground Truth

Figure 1: Project Development Gantt Chart

Semester 2

- Weeks 1-2: Polish web solution to function for input MP3 files and YouTube URLs.
  - **Deliverable:** Working web application that outputs generated videograpy data for inputted audio sources.
- Weeks 3-5: Create ground truth data on previously agreed songs.
  - **Deliverable:** Ground truth data for the selected songs.
- Weeks 6-8: Research and work on calculating performance metrics of automated videography tool.
  - **Deliverable:** Performance metrics for the generic automated videographic tool for the selected songs using the ground truths.
- Weeks 8-10: Write up of dissertation.
  - **Deliverable:** First draft submitted to supervisor.