

# MLDA Deep Learning Week Hackathon

## Problem #1: Tiktok Video Captioning Data Filtering Pipeline

---



### **Xponential Group:**

- Ng Zheng Xun (CSC Y4)
- Lyu XingXiao (EEE Masters)
- Kevin Hu (EEE Masters)

# Introduction to Problem Statement



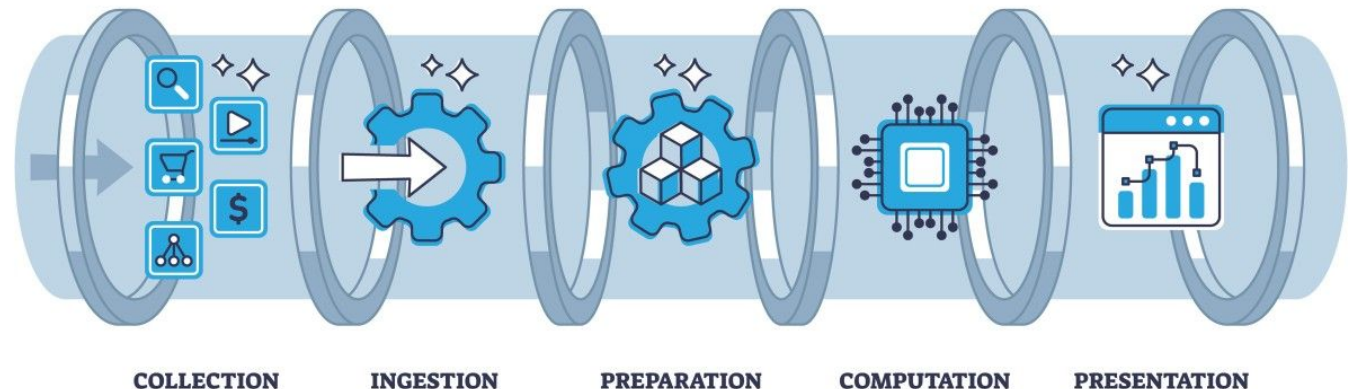
Given a Noisy Dataset of 1M Video-Caption pairs,

construct a pipeline to filter out the top 10% (10k pairs) in terms of quality

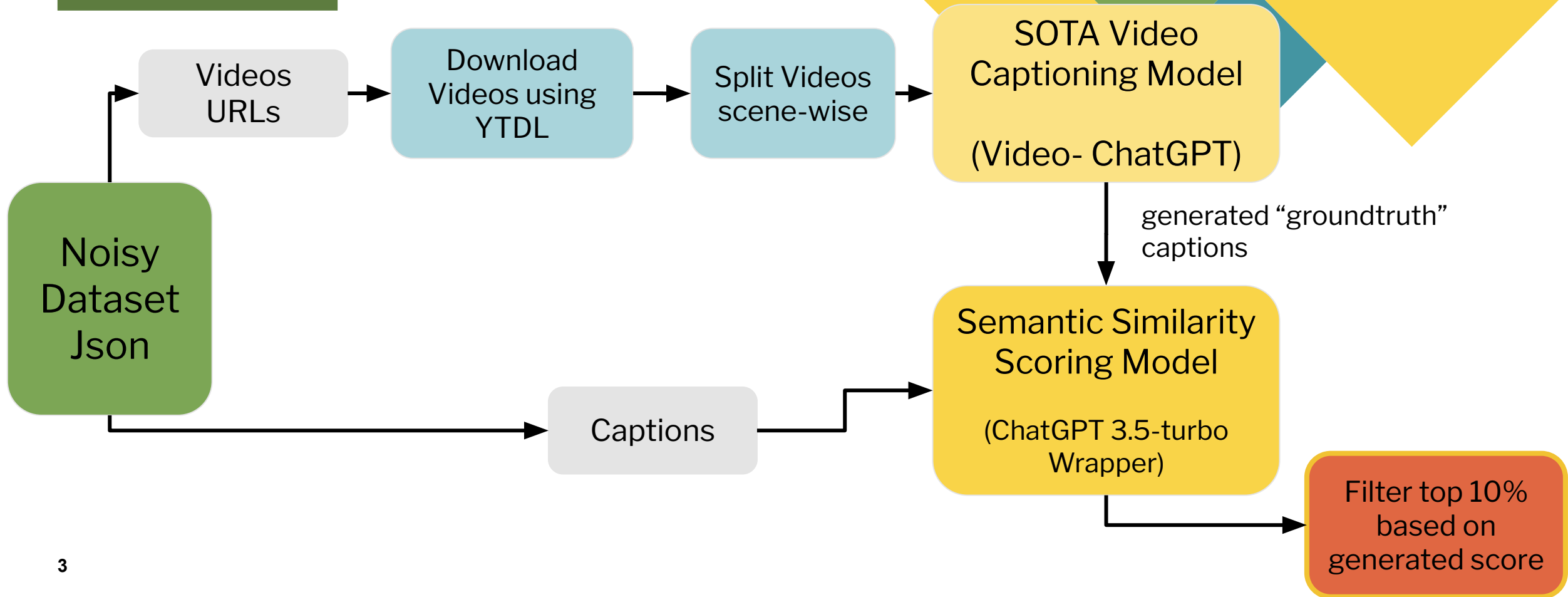
- crucial for improving model performance

ideally optimised to reduce CO2 emissions

## DATA PIPELINE



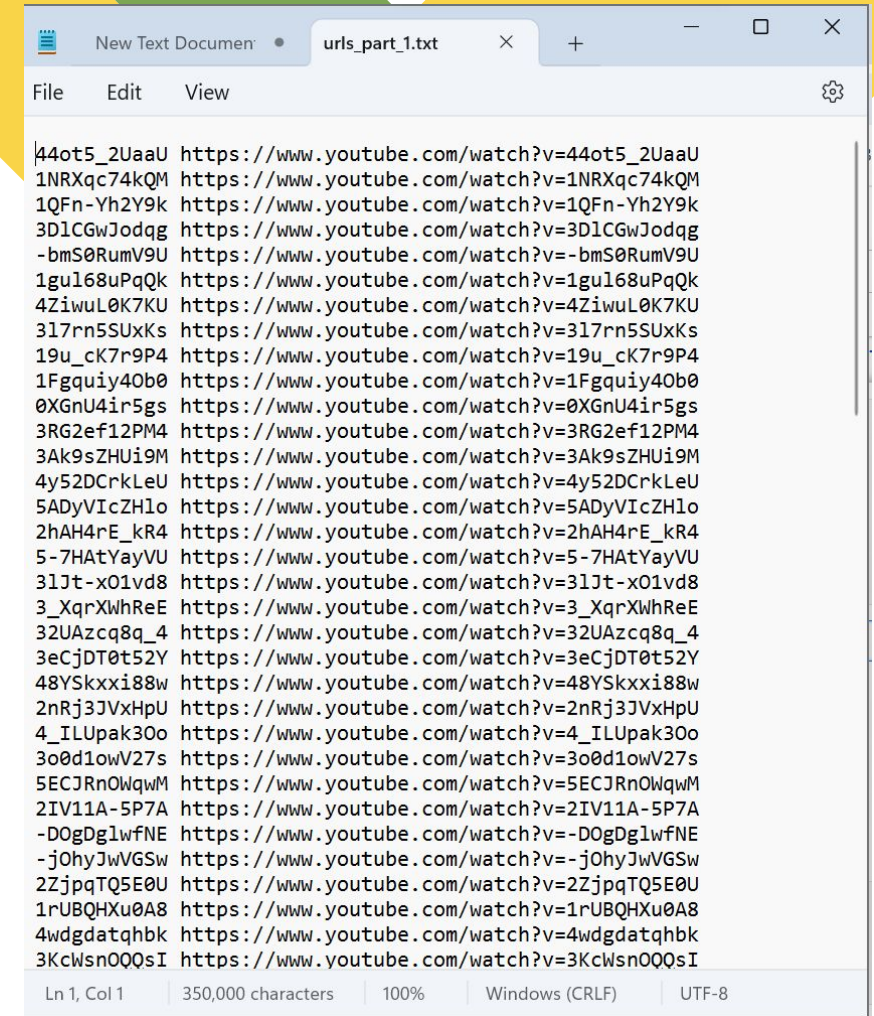
# Overall Pipeline



# Loading and Parsing Noisy Dataset Json



- Initial Idea: multiprocessing
  - ultimately not implemented due to time constraints, and the lack of GPU processing
- Extracting youtube links and video ID from json file, and saving it as 3 separate text files, for multiprocessing

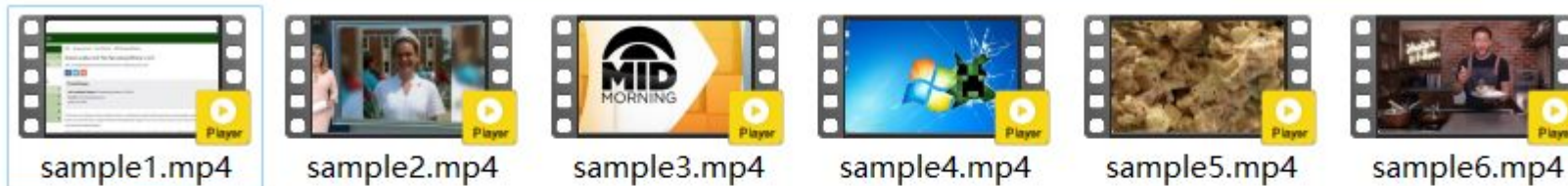
A screenshot of a text editor window titled 'New Text Document' with a tab labeled 'urls\_part\_1.txt'. The editor contains a list of 30 YouTube URLs, each preceded by a video ID. The status bar at the bottom shows 'Ln 1, Col 1', '350,000 characters', '100%', 'Windows (CRLF)', and 'UTF-8'.

```
44ot5_2UaaU https://www.youtube.com/watch?v=44ot5_2UaaU
1NRXqc74kQM https://www.youtube.com/watch?v=1NRXqc74kQM
1QFn-Yh2Y9k https://www.youtube.com/watch?v=1QFn-Yh2Y9k
3D1CGwJodqg https://www.youtube.com/watch?v=3D1CGwJodqg
-bmS0RumV9U https://www.youtube.com/watch?v=-bmS0RumV9U
1gul68uPqQk https://www.youtube.com/watch?v=1gul68uPqQk
4ZiwuL0K7KU https://www.youtube.com/watch?v=4ZiwuL0K7KU
317rn5SUXks https://www.youtube.com/watch?v=317rn5SUXks
19u_cK7r9P4 https://www.youtube.com/watch?v=19u_cK7r9P4
1Fgquiy40b0 https://www.youtube.com/watch?v=1Fgquiy40b0
0XGnU4ir5gs https://www.youtube.com/watch?v=0XGnU4ir5gs
3RG2ef12PM4 https://www.youtube.com/watch?v=3RG2ef12PM4
3Ak9sZHU9M https://www.youtube.com/watch?v=3Ak9sZHU9M
4y52DCrkleU https://www.youtube.com/watch?v=4y52DCrkleU
5ADyVicZH1o https://www.youtube.com/watch?v=5ADyVicZH1o
2hAH4rE_kR4 https://www.youtube.com/watch?v=2hAH4rE_kR4
5-7HatYayVU https://www.youtube.com/watch?v=5-7HatYayVU
31Jt-x01vd8 https://www.youtube.com/watch?v=31Jt-x01vd8
3_XqrXWhReE https://www.youtube.com/watch?v=3_XqrXWhReE
32UAzcq8q_4 https://www.youtube.com/watch?v=32UAzcq8q_4
3eCjDT0t52Y https://www.youtube.com/watch?v=3eCjDT0t52Y
48YSkxxi88w https://www.youtube.com/watch?v=48YSkxxi88w
2nRj3JVxHpU https://www.youtube.com/watch?v=2nRj3JVxHpU
4_ILUpak30o https://www.youtube.com/watch?v=4_ILUpak30o
3o0d1owV27s https://www.youtube.com/watch?v=3o0d1owV27s
5ECJRn0WqwM https://www.youtube.com/watch?v=5ECJRn0WqwM
2IV11A-5P7A https://www.youtube.com/watch?v=2IV11A-5P7A
-D0gDglwfNE https://www.youtube.com/watch?v=-D0gDglwfNE
-j0hyJwVGSw https://www.youtube.com/watch?v=-j0hyJwVGSw
2ZjppQTQ5E0U https://www.youtube.com/watch?v=2ZjppQTQ5E0U
1rUBQHxU0A8 https://www.youtube.com/watch?v=1rUBQHxU0A8
4wdgdatqhbK https://www.youtube.com/watch?v=4wdgdatqhbK
3KcWsn00QsI https://www.youtube.com/watch?v=3KcWsn00QsI
```

# Automating the Download of Videos from URLs



- We used the **Python youtube-dl (YTDL) library** to do an automated download of the given URLs in mlda\_data.json.
- We download videos in a lower resolution of 360p to optimise computational load





# Splitting Downloaded Videos using Python script



- Clip videos from the downloaded video according to the timestamp shown in *mlda\_data.json* through the given programme by Tiktok.
- much less intensive due to the script provided.
- After some debugging and tweaking some variables, **`python cut_videos_mlda.py --metafile mlda_data.json --resultfile cut_part0.jsonl`**

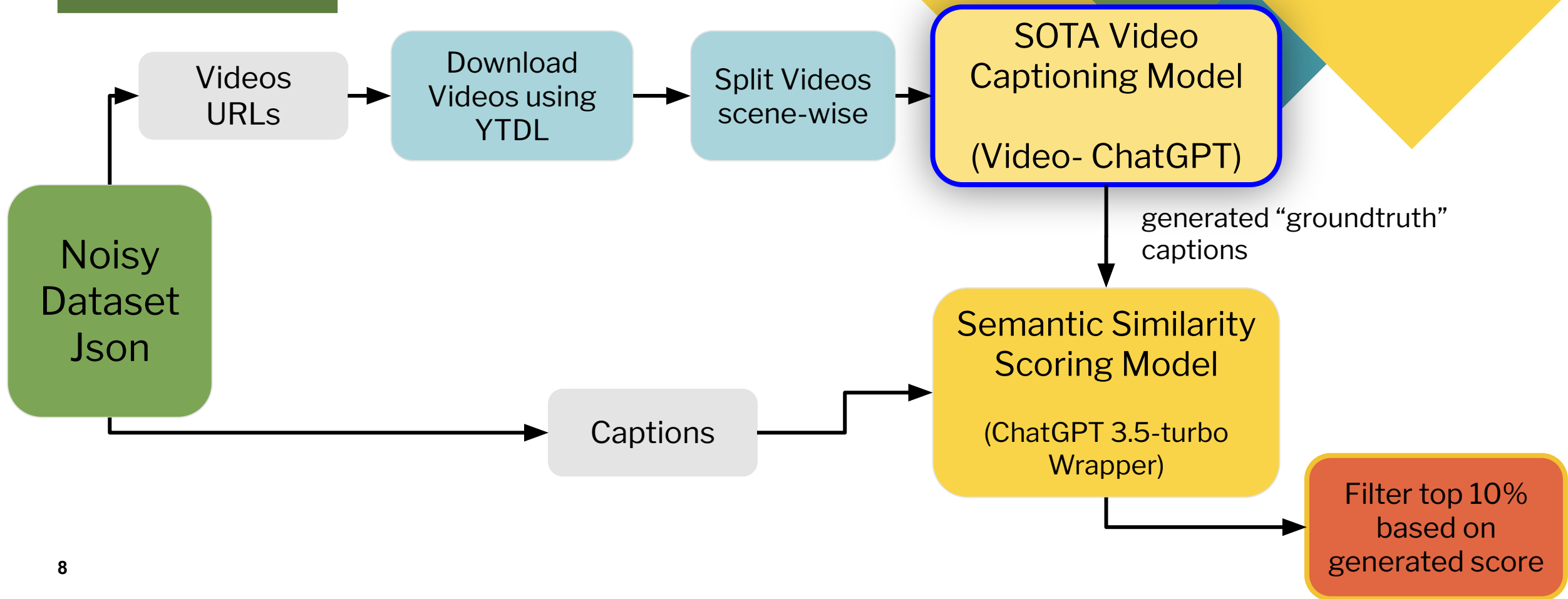
```
C:\Users\ngzhe\Anaconda3\lib\site-packages\numpy\.libs\libopenblas64__v0.3.21-gcc10_3-
warnings.warn("loaded more than 1 DLL from .libs:")
./metafiles\mlda_data.json
0%|
./tmp_clips\gpAkjSy_8iY\gpAkjSy_8iY.0.mp4
[ '00:00:00.030', '00:00:07.740' ]
download_videos\gpAkjSy_8iY.mp4
./download_videos\gpAkjSy_8iY.mp4
./tmp_clips\gpAkjSy_8iY\gpAkjSy_8iY.8.mp4
[ '00:01:46.479', '00:01:57.009' ]
download_videos\gpAkjSy_8iY.mp4
./download_videos\gpAkjSy_8iY.mp4
./tmp_clips\gpAkjSy_8iY\gpAkjSy_8iY.10.mp4
[ '00:02:14.739', '00:02:22.380' ]
download_videos\gpAkjSy_8iY.mp4
./download_videos\gpAkjSy_8iY.mp4
./tmp_clips\gpAkjSy_8iY\gpAkjSy_8iY.9.mp4
[ '00:01:57.009', '00:02:14.739' ]
download_videos\gpAkjSy_8iY.mp4
./download_videos\gpAkjSy_8iY.mp4
./tmp_clips\gpAkjSy_8iY\gpAkjSy_8iY.2.mp4
[ '00:00:23.789', '00:00:35.969' ]
download_videos\gpAkjSy_8iY.mp4
./download_videos\gpAkjSy_8iY.mp4
./tmp_clips\gpAkjSy_8iY\gpAkjSy_8iY.5.mp4
[ '00:01:00.600', '00:01:18.729' ]
download_videos\gpAkjSy_8iY.mp4
./download_videos\gpAkjSy_8iY.mp4
./tmp_clips\gpAkjSy_8iY\gpAkjSy_8iY.4.mp4
[ '00:00:55.140', '00:01:00.600' ]
download_videos\gpAkjSy_8iY.mp4
./download_videos\gpAkjSy_8iY.mp4
./tmp_clips\gpAkjSy_8iY\gpAkjSy_8iY.3.mp4
[ '00:00:35.969', '00:00:55.140' ]
download_videos\gpAkjSy_8iY.mp4
./download_videos\gpAkjSy_8iY.mp4
./tmp_clips\gpAkjSy_8iY\gpAkjSy_8iY.1.mp4
[ '00:00:07.740', '00:00:23.789' ]
download_videos\gpAkjSy_8iY.mp4
./download_videos\gpAkjSy_8iY.mp4
./tmp_clips\gpAkjSy_8iY\gpAkjSy_8iY.6.mp4
[ '00:01:18.729', '00:01:38.439' ]
```

# Splitting Downloaded Videos using Python script



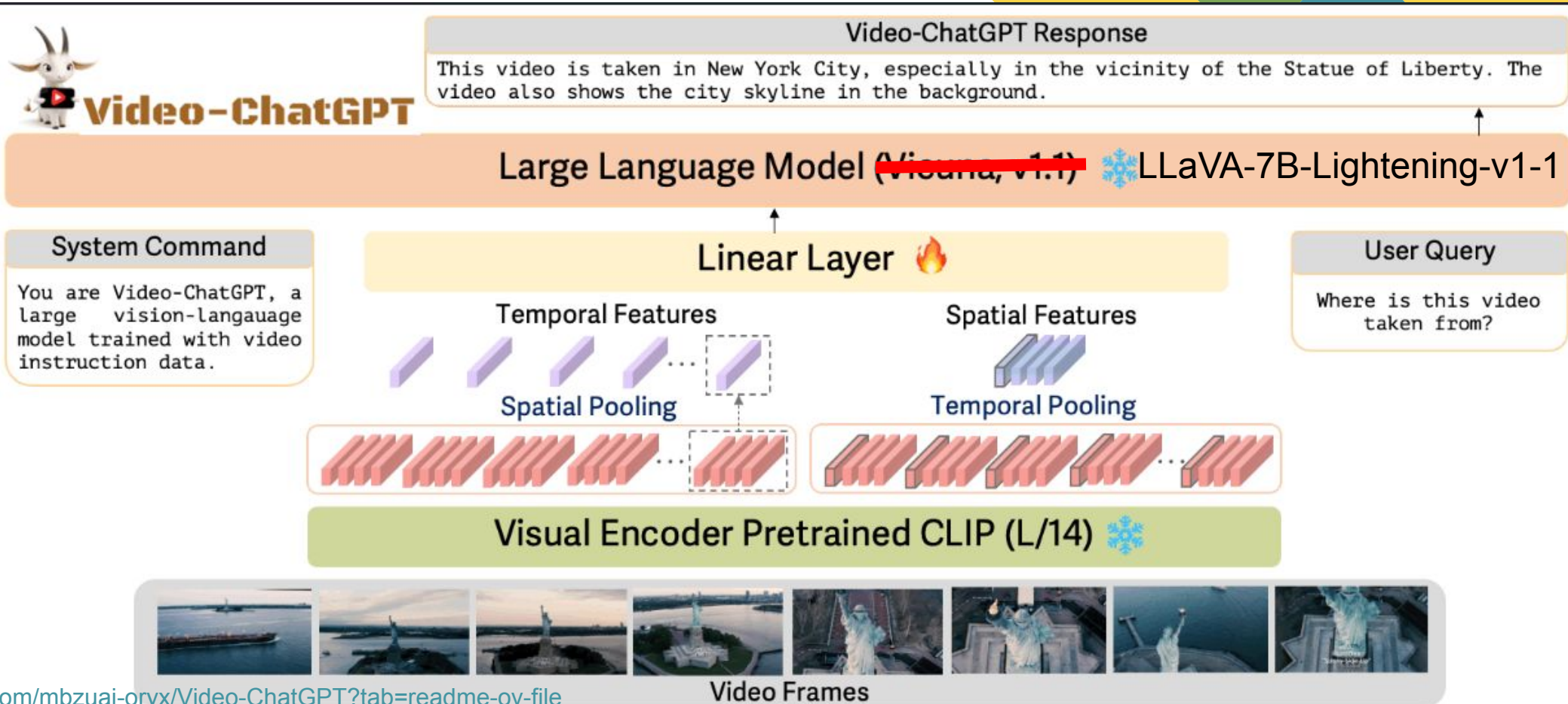
We take the FIRST FOUR videos from the above folder as our toy dataset

# Overall Pipeline





# Generate "groundtruth" Caption with SOTA Video Captioning Model



# Generate "groundtruth" Caption with SOTA Video Captioning Model

---



- Usage of State-of-the-Art (SOTA) Video Captioning Model **VideoChatGPT** to generate a groundtruth caption
- Model was chosen as it gives **SOTA and industry-leading results** on several video captioning benchmarks, such as the VideoInstruct benchmark

# Generate "groundtruth" Caption with SOTA Video Captioning Model



## Video-based Generative Performance Benchmarking on VideoInstruct



# Generate "groundtruth" Caption with SOTA Video Captioning Model

---



- Model can also do Question-Answering, but **we focus on the Video Captioning instead**
- - Interestingly, the original authors **approached the Video Captioning task as a variation of Question-Answering**, whereby the question is simply the provided noisy caption.

# Generate "groundtruth" Caption with SOTA Video Captioning Model

---



- There were **significant difficulties** in setting up a suitable environment, and to get the model running.
- However, we were able to overcome these difficulties, and adapting the model to run on a CPU environment



# Generate "groundtruth" Caption with SOTA Video Captioning Model

---



- Other models were considered, such as mPLUG-2 by Alibaba, but were ultimately not implemented due to time constraints. Ideally, we hoped to be able to do a comparison between the 2 models
- CLIP score was considered but as an **image-to-score metric**, it was **insufficient** and **unable to capture crucial context** or verbs (movement-based)

# Generate "groundtruth" Caption with SOTA Video Captioning Model



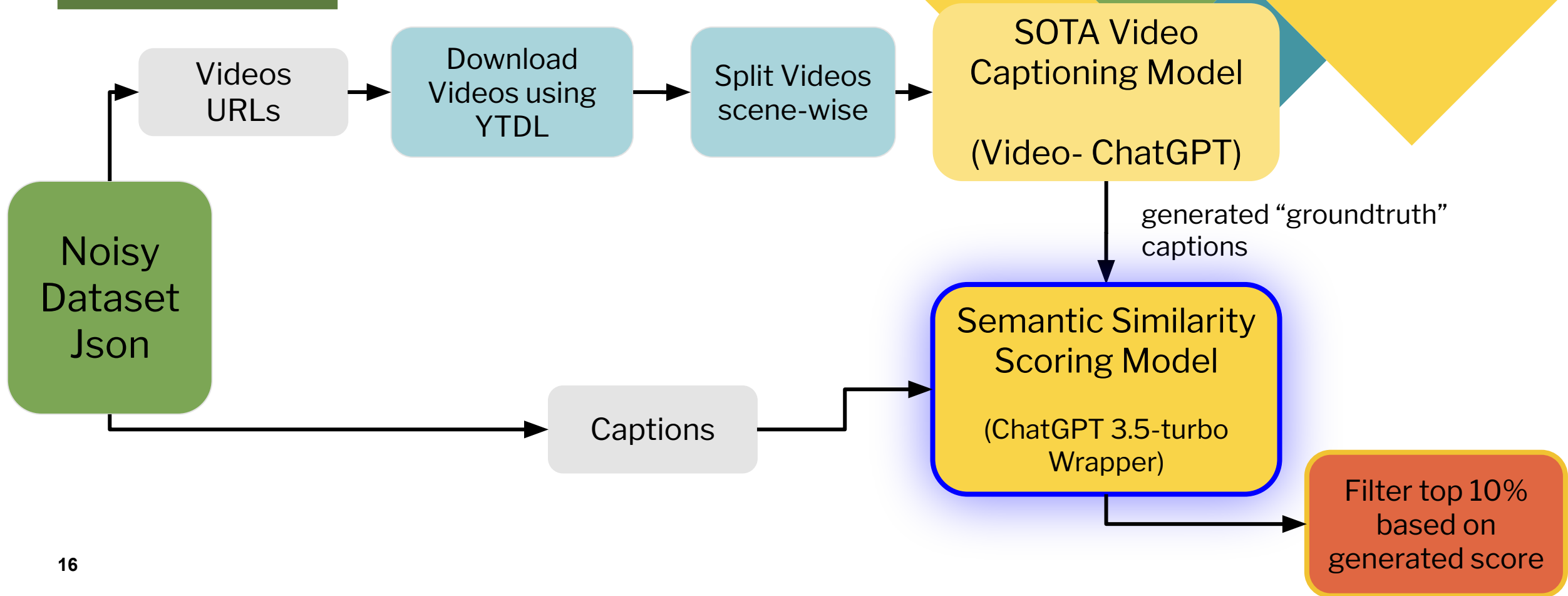
```
{  
  "clip": "gpAkjSy_8iY/gpAkjSy_8iY.0_0.mp4",  
  "caption": "a man in a suit and tie standing next to a news desk",  
  "pred": "The man in the suit and tie is standing next to a news desk, and he is talking  
to the camera. He is wearing a suit and tie, which suggests that he is likely a news  
reporter or a guest on a news show. He is talking about the news and providing information  
to the audience."  
},
```

```
{  
  "clip": "gpAkjSy_8iY/gpAkjSy_8iY.8_1.mp4"  
  "caption": "a street that has power lines"  
  "pred": "The video shows a street with po  
about the situation. He is wearing a suit and  
},  
}
```

```
  "clip": "gpAkjSy_8iY/gpAkjSy_8iY.8_2.mp4"  
  "caption": "a woman standing in the middl  
  "pred": "In the video, a woman is standin  
by water. She is holding a microphone and spe  
also speaking to the camera, and the news rep  
reporter is talking about the flood and the w  
reporter is also talking about the flood and  
},
```



# Overall Pipeline



# Generating a Semantic Similarity Score

---



- An NLP problem (comparison between 2 texts)
  - Several classical methods (n-grams, Character/Word Error Rate etc)
- Why Semantic Similarity? instead of Syntactic similarity?
  - There are multiple ways to express the same idea in natural language!
  - methods such as n-grams would not be useful / effective

# Generating a Semantic Similarity Score

- Model Used: A **ChatGPT 3.5 turbo** wrapper

*"You are an intelligent chatbot designed for evaluating the detail orientation of generative outputs for video-based question-answer pairs. "*

*"Your task is to compare the predicted answer with the correct answer and determine its level of detail, considering both completeness and specificity. Here's how you can accomplish the task:"*

*"-----"*

*"###INSTRUCTIONS: "*

*""- Check if the predicted answer covers all major points from the video. The response should not leave out any key aspects.\n"*

*"- Evaluate whether the predicted answer includes specific details rather than just generic points. It should provide comprehensive information that is tied to specific elements of the video.\n"*

*"- Consider synonyms or paraphrases as valid matches.\n"*

*"- Provide a single evaluation score that reflects the level of detail orientation of the prediction, considering both completeness and specificity."*



GPT **3.5** T U R B O



# Generating a Semantic Similarity Score



- [{"score": 5}, {"q": "a man in a suit and tie standing next to a news desk", "a": "a man in a suit and tie standing next to a news desk", "pred": "The man in the suit and tie is standing next to a news desk, and he is talking to the camera. He is wearing a suit and tie, which suggests that he is likely a news reporter or a guest on a news show. He is talking about the news and providing information to the audience."}]
- [{"score": 3}, {"q": "a street that has power lines down on it", "a": "a street that has power lines down on it", "pred": "The video shows a street with power lines down on it, and a man is talking about the situation. He is wearing a suit and tie and is standing in front of a news desk."}]
- [{"score": 2}, {"q": "a woman standing in the middle of a flooded street", "a": "a woman standing in the middle of a flooded street", "pred": "In the video, a woman is standing in the middle of a flooded street, surrounded by water. She is holding a microphone and speaking to the camera. The man in the suit is also speaking to the camera, and the news reporter is sitting in front of a desk. The news reporter is talking about the flood and the woman standing in the flooded street. The news reporter is also talking about the flood and the woman standing in the flooded street."}]

# Generating a Semantic Similarity Score

---

- After a few rounds of experimenting, we ran into `openai.error.RateLimitError`: You exceeded your current quota, please check your plan and billing details. For more information on this error, read the docs:  
<https://platform.openai.com/docs/guides/error-codes/api-errors>
- Decided to stop here for the running and usage of the models, as the **above results from the model shows that the pipeline is functional, and operates as intended.**



GPT **3.5** T U R B O

# Sorting of Results

---

- After each group of videos is rated, the scores and corresponding serial numbers are reordered, and the top 10% with the highest scores are screened out.



# Problems Solved

---



- Configured the required environments: pytorch, ffmpeg, pillow, etc.
- Without GPU, Video based performance benchmarking was modified for CPU processing and successfully produced results.
- Modified the format to meet the calling needs of different program modules.
- Submitted a functioning and complete data filtering pipeline integrating 2 SOTA models

# Efforts to reduce CO2 emissions

---



- **Optimize download resolution:**
  - On the premise of ensuring recognition accuracy, we try to reduce the resolution of the downloaded video to reduce the consumption of network bandwidth resources.
- **Using Pre-trained Models:**
  - We opt to not train a SOTA model from scratch which would consume significant computational resources



# Further Work/Optimisations



- UI / Web App to display progress/results so far
  - Current scores and serial numbers of all the clips may be dynamically updated / rearranged
  - User can view progress and terminate the data pipeline at any time
  - For easier usage, compilation of to .exe could be considered too
- Comparison between several Video Captioning and Semantic Comparison Models
  - Less of unfinished work, more of an optimisation
  - Always ideal to compare against industry benchmarks and other models (if possible computationally)



# Thank You! :)

---

Produced by Xponential group