AI Framework for a Full-Automatic Youtube Channel

Group 8 Written Report

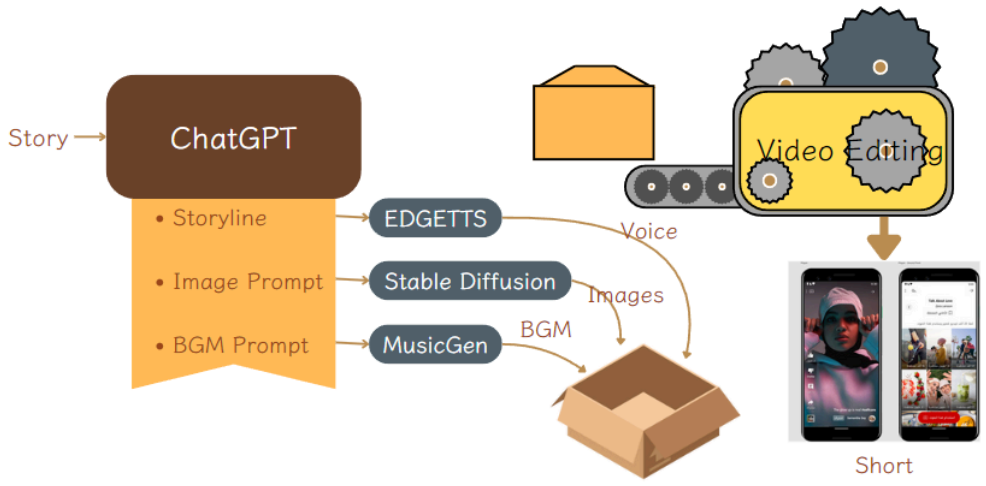
110612117 張仲瑜, 110550128 蔡耀霆, 110652032 許元瑞

Motivation

Through learning various generation models in the AI series courses, we've stepped into the world of AI artwork and witnessed the power and potential of it. As a result, we are curious about whether we can run a Youtube channel with contents automatically created by those models. We ended up deciding to implement a powerful framework for automating content creation, which can automatically generate videos with a given topic.

Technique used

- System Overview



- Text to Prompt
    - Pre-design system prompts to leverage the power of GPT3.5
    - By calling Openai's API with designed system prompts and input, we can get the storyline script and prompts for generating images and prompts for generating BGM.

    - We gain our prompt in a specific form, which ensures that the reply is processable and steady to generate the prompt for following procedures.
    - We also allowed the user to set the pre-prompt to make the video style meet their expectations, which means the final prompt for generating will be the concatenation of pre-prompt(by user) and prompts by GPT.
- Prompt to Image
    - Build a diffusers pipeline with a SDXL pretrained model to generate high quality pictures.
    - Leverage some pre-positive/negative prompts to refine our generated result. Based on our experience, this is an effective method to avoid unrelated outputs.
- Prompt to BGM
    - Build a musicgen pipeline with the smallest pretrained model and BGM prompt to generate the music.
    - We also did another experiment on another more complicated model with a function to generate the music with a reference work.
- Prompt to TTS and Subtitle
    - Utilize the human voice model provided by Microsoft Edge TTS (Text-to-Speech) to generate speech and subtitles. The main functionality is to convert input text into an audio file (MP3 format) and generate corresponding subtitle files with timeline (VTT format).
    - We originally designed a method to perform subtitles segmentation based on semantics, but the subtitles won't appear along with the audio and part of the subtitles may be

outside the screen. So we tried another method, that is, to display only a few words at one time to fix these problems.

- Video Editing
    - Using the Moviepy library to perform the editing of the video to combine the images, background music, narration, and subtitles together with simple transitions written by ourselves.

Experiments and Analysis

- Result Link: 🗀 **Experiment**
- Prompt engineering (ablation test) 🗀 prompt segmentation
    - prompt segmentation
        - Although the number of generated images meets the requirement with the prompt separation, the correspondence between the images and the scripts is apparently reduced due to the semantic loss during the segmentation.
        - Prompts before segmentation

1. The battle against the monster was intense and fierce.
2. Liam and his team fought courageously, utilizing their combined skills and magic to weaken the creature.
3. Although the monster was incredibly strong, their resolve remained steadfast.
4. Amidst the chaos of battle, Liam recalled his training and the advice of his mentor.
5. With one last, mighty spell, he and his team successfully overcame the monster.
6. The village was rescued, and the monster was defeated.

        - Prompts after segmentation

1. Intense battle scene, warriors fighting, monster roaring, magic spells, fierce combat.
2. Liam and his team strategizing, teamwork, magic spells casting, determination, bravery.
3. Monster's strength, ominous presence, formidable foe, towering creature, menacing eyes.
4. Chaos of battle, swords clashing, explosions, magical effects, dust and debris.
5. Training flashbacks, mentor guiding Liam, learning moments, wisdom imparted, flashback scenes.
6. Final spell casting, powerful aura, glowing magic, concentration, climax of battle.
7. Villages in danger, fearful villagers, burning buildings, distressed people, call for help.
8. Monster defeated, triumphant moment, celebration, victory poses, relief and happiness.
9. Rescued village, grateful townsfolk, rebuilding efforts, community coming together, hope restored.
10. Vanquished monster, defeated creature, lifeless foe, epic aftermath, peace restored.

    - pre-promp 🗀 preprompt
        - Applying this technique reduces the chances of generating unrealistic images, and is able to generate the image according to users' requests .
        - For example, we put a manga tag in a positive pre-prompt. We can get six images in manga style out of ten images. However, if we get rid of the pre-prompt, there will be no images in manga style due to no limitation. As for negative pre-prompt, we suggest many unacceptable situations, like: people with more than two hands, face distortion ... .We expected that those unrealistic images should not be generated, but the effect on outcome is not so obvious. We supposed that it was a defect of this diffusion model, when there are too many requirements some of them may be ignored.

- Music model selection and pipeline
    - reference music 🗀 BGM_reference_music

      We aim to test the effect of the reference music in this experiment. It turns out that the results are closely related to

the given music. The melody and the tempo will be similar given the same reference music, and the style will be slightly different according to the given prompt.
Furthermore, using the reference music will reduce a lot of the GPU memory usage.

- BGM prompt regularization 🗀 BGM prompt regularization
  In this section, we aim to optimize our BGM (Background Music) prompt. The original version of the prompt was rough and unclear. To improve it, we will establish more specific guidelines to regulate the prompt, including aspects such as style, tempo, BPM, emotion, and atmosphere.

  Original prompt:

```
[Prompts for BGM]
1. Enchanting and mystical instrumental music to capture the ambiance of the magician school.
2. Upbeat and inspiring music to reflect Liam's excitement and eagerness to learn.
3. Mysterious and suspenseful music to accompany scenes of Liam encountering difficulties with spells.
4. Reflective and introspective music to convey Liam's moments of doubt and contemplation.
5. Empowering and uplifting music to symbolize Liam seeking guidance and support to overcome challenges.
```

  Here is the example prompt:

```
[Prompts for BGM]
Style: Lofi
Tempo: Slow
BPM: 80-90
Emotion and Atmosphere: Reflective, contemplative, with a hint of nostalgia and uncertainty
```

  It is obvious that the BGM with regulated prompt has better and steady outcomes.

- Time and Memory
  - 20 steps for image generation and 15 second BGM with 16 floating points accuracy are the limitations of a 15 GB Tesla T4 GPU. It takes about 15 seconds to generate an image, and takes about 45 seconds to generate a BGM.
    To create a 1 minute short, we need about 8.5 minutes.

Results

- **Demo channel 1:** https://www.youtube.com/@aidreamingfirm/shorts
- **Demo channel 2:** https://www.youtube.com/@StoryTeller-VRJ/shorts

Conclusion

Summary:

This assignment involves creating an automated AI pipeline capable of generating AI shorts. Each component can be treated as a subtopic: using a storyline to generate prompts, then using prompts to create background music (BGM), text-to-speech (TTS), and images. Finally, these elements are combined using Python's moviepy library. We put every effort to enhance the quality of our shorts, prompting us to think about how to optimize the output at each stage and how modifications can lead to more stable and high-quality results in the subsequent processing by the model.

Future work

1. Generate small dynamic video clips instead of static images.
2. Higher computing power to use a better music model.
3. Apply various video transitions in the video.
4. Leverage web crawlers to automatically find the topic and upload to YouTube to achieve full-automation.

Thoughts

Through this assignment, we have caught up with the latest tools to complete the generation tasks mentioned above. Due to the limitation of

computing power, the tools we chose this time are relatively simple, easy to integrate, and not resource-intensive.

To improve the outputs of each submodule, changing the models is the most effective method. However, due to limitations mentioned above, we decided to integrate the prompt techniques to boost the performance, and eventually we are quite satisfied with the current output, though there is still much room for continuous improvement. It is definitely a great experience for us to learn how to achieve a goal by modifying and optimizing the existing codes.

Contributions

110612117 張仲瑜 –

prompt generation, text to image, video cliping, report writing

110652032 許元瑞 –

text to speech, text to subtitle, code integration

110550128 蔡耀霆 –

text to music, video cliping, report writing

Reference

1. **stabilityai/stable-diffusion-xl-base-1.0 · Hugging Face**
2. **MusicGen - a Hugging Face Space by facebook**
3. **edge-tts · PyPI**
4. **moviepy · PyPI**
5. **https://openai.com/index/openai-api/**