ART AND MACHINE LEARNING

CMU 2023 SPRING

PROJECT 2

Lyrics2Image

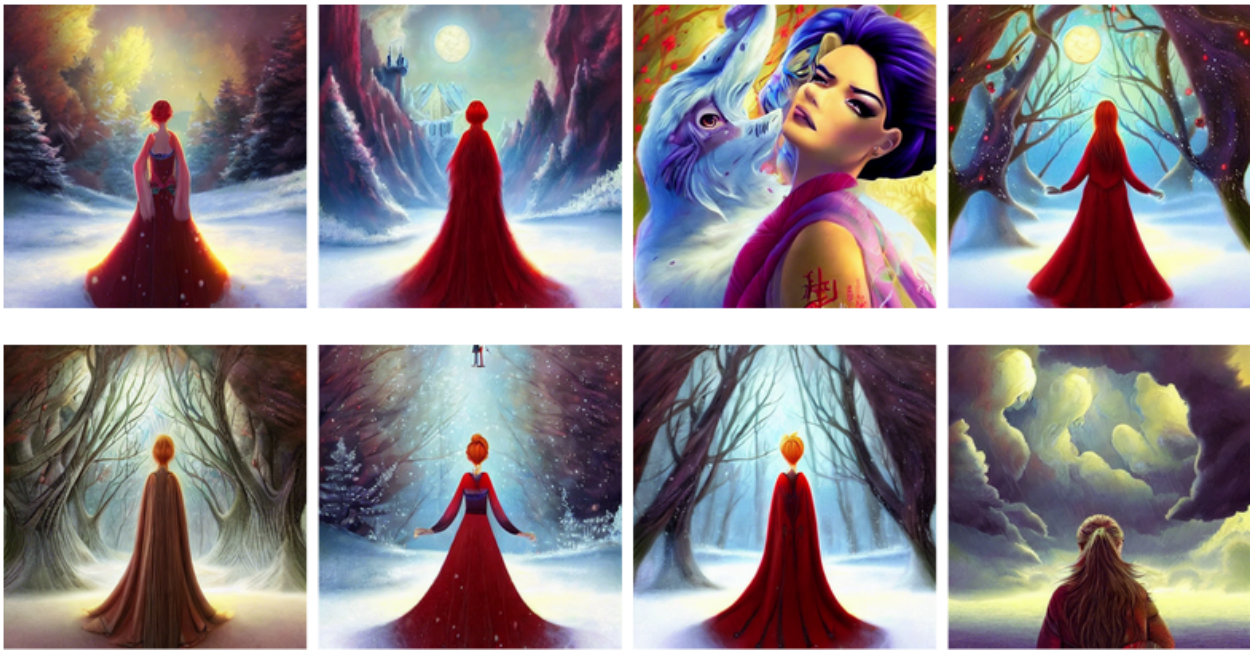


Figure 1.
Let It Go - Frozen

Yi-Hui Chou, Sharon Zhang, Winnie Chang

DESCRIPTION

The goal of this project is to generate consistent images given lyrics. Initially, we wanted to generate music videos, but the existing text-to-video research, including MAKE-A-VIDEO (Meta AI) [4] and Imagen (Google) [4], has some limitations. First, these models require a massive video dataset to get a decent result, but the companies did not release datasets or pre-trained checkpoints (it is more feasible to fine-tune from models pre-trained on videos than train from scratch). Moreover, the generated videos from these models are often only seconds long, which limits their application in music video production. Due to the time and computation constraints, we decided to shift our focus to "Lyrics to Images". This is similar to a storybook project, *The Adventure of Penelope the Porcupine and the Land of Whimsy,* where the developer uses Midjourney to generate a series of images given a story written by chatGPT. The main challenge of our project is that the lyrics are often vague and lack specific details, which is very different from the typical prompts for image generation. Our goal is to generate consistent images over time.

1. Concept

The original idea is inspired by multiple existing techniques. First, there are many existing text to images AI tools, but not many tools handling text+image to images or videos. Second, in machine learning for languages, sequential models such as LSTM are widely applied, yet models taking sequential text and image as input and generating the next image are not commonly seen. Third, current text to image tools could handle short text or single description fairly well, but could hardly digest a series of lyrics. The challenges mainly come from two aspects: 1. Lyrics are a series of text with or without obvious connection, hence it is difficult to summarize the lyrics into one gigantic description. 2. If lyrics are fed line-by-line into the tools, the output images might not be related at all and could have very different styles.

For the first challenge, we decided to use the GPT-3 model for prompt modification. OpenAI API provides various models for text summarization and modification, and we experimented with multiple instruction prompts to come out with the best performance prompt to re-write the lyrics into connected and logical sentences. Though the result is still a series of sentences, it is better than lyrics without obvious connections. We then feed the sentence line by line into the image generation model.

For the second challenge, based on the above observations, we came up with the idea to modify an existing machine learning model to have it intake lines of lyrics and generate a series of related images (or a video depicting all generated images in time series). The previously generated image should be taken as the input to the following step of the model, together with the following lyric text, to generate the next image based on the previous image and the new text. Meanwhile, the style of all generated images should be persistent.

We sought advice from Professor Jun-Yan Zhu and he directed us to the universal guidance algorithm [2]. The algorithm allows arbitrary guidance modalities to control diffusion models without retraining any use-specific components through including guidance from off-the-shelf auxiliary networks. building on the algorithm, an image generation model that preserves the aesthetic embeddings, if specified, but also

fulfills the text constraints, could be trained and fine-tuned. Their current model, however, only takes in one input text and a specified pre-trained aesthetic embedding, and does not handle sequential text inputs.

It is generally difficult to train a new diffusion model from scratch, hence we decided to build our idea on top of the universal guidance model. We adjust the CLIP to take into account the previous input image, and use the adjusted model to generate a series of images based on the input lyrics. Since the following images are all based on previous images, the style could be largely preserved along the sequence.

2. Technique

A bird's-eye view of our lyrics-to-images system is depicted in Figure 1 below. First, the lyrics are segmented and preprocessed to highlight important concepts or objects. Then, the preprocessed lyrics will be fed into the text-to-image model. The technical details will be elaborated in corresponding subsections.

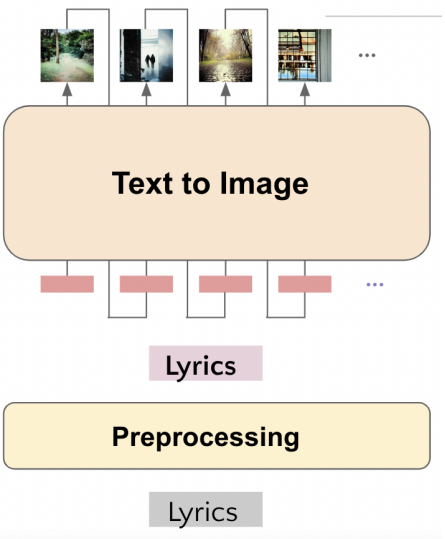


Fig. 1 Lyrics to Images System

Figure 2.

2-1. Lyrics Preprocessing

The lyrics are first preprocessed using OpenAI's Completions model[3] with the prompts "Separate the sentences and number:" followed by feeding that output to the prompt "Synonyms:\n Remove connecting words:" (see the Process section for a detailed example). The output of the second prompt is appended to the output of the first prompt and that becomes our text input to stable diffusion.

2-2. Text to Image

The augmented lyric sentences are then passed one by one into the text-to-image model. This part is inspired by [5]. The author pointed out that it is hard for users to describe their desired styles clearly, so he proposed aesthetic gradients, a method to guide the diffusion model toward custom aesthetics defined by the user from a set of images. More specifically, the text is fed into the CLIP text encoder to extract textual embedding, which will be fed into the Stable Diffusion model to generate an image. The author tries to modify the textual embedding by employing "aesthetic gradients (or style embedding)", which is the average of image features extracted by CLIP image encoder. Then, the CLIP text encoder is updated to maximize the dot product of the textual and style embeddings.

$$txt\_embed = CLIP_{txt}(text)$$

$$style\_embed = \sum_{i=1}^{K} CLIP_{img}(image_i)$$

$$updated\_style\_embed = \alpha \times style\_embed + (1 - \alpha) \times content\_embed$$

We propose image generation with content feedback by extending his research to satisfy our needs. That is, we update the style embedding every timestep. The first image is generated given a defined style embedding. Then, the following generated images will be conditioned on both the lyrics and the updated style embedding, which is a mix of the given style embedding and the "content" embedding (average of k previously generated images). Here, k and α are hyperparameters. Note that both the Stable Diffusion model and the CLIP image encoder are frozen, the only model we need to fine-tune is the CLIP text encoder, which makes it highly parameter-efficient. A series of images can be generated within a few minutes.

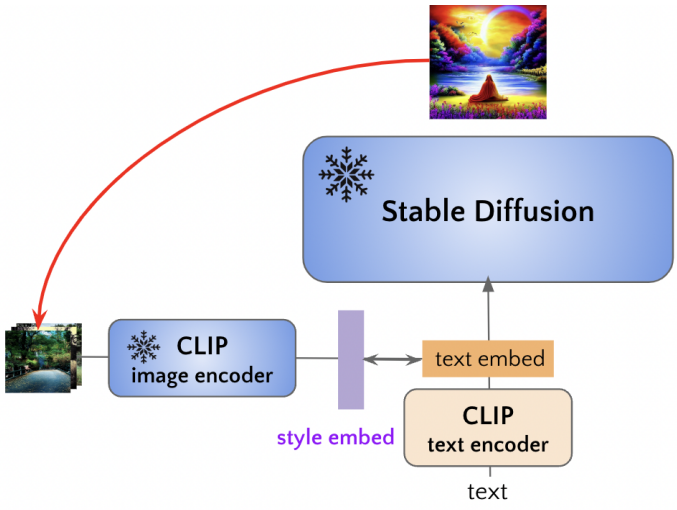


Fig. 2 Our adjusted Text-to-Image generation system with content feedback.

Figure 3.

3. Process

3-1. Lyrics Preprocessing

Initially, we just generated images by blocking the lyrics every two lines. However, one drawback of such a method is that we risk generating images based on partial details because connected lyrics were split. To remedy this, we decided to use OpenAI's GPT3 Completions model[3] to automatically block lyrics according to what makes a "sentence" to ensure that relevant details are connected to each other.

```
Separate into sentences and number:

In this world you tried

Not leaving me alone behind

...
```

```
1. In this world you tried not leaving me alone behind, there's
no other way.
```

The Completions model was pretty good at blocking the relevant lyric lines into sentences, however there were 2 things we noted. First, the model wasn't always stable. Sometimes the model would just spit the lyrics right back out line for line and other times it would clump all the lines together into one long run-on sentence, especially when the prompt was just "Separate into sentences:". The second is that what the model outputs as a sentence might be considered as 2 or more sentences by a human annotator. However, having multiple sentences is better than missing out on details and the model's output did provide significantly better images compared to if the lyrics were just blocked two lines at a time.

Our next step took inspiration from Jeong et al. (2023)[1] where they first preprocessed their input prose by having their model first "rewrite" prose into a more descriptive, caption-like format and then simplifying that result through summarization. Attempting this with our song lyrics, however, did not yield very good textual outputs, instead adding more details.

```
Describe the scene:

Seven A.M., the usual morning lineup

Start on the chores and sweep 'til the floor's all clean
```

```
It is seven in the morning and the sun is just starting to rise
```

Modifying the parameters of the Completions model did nothing to improve the outcome, so we ruled this method out. We also tried "Summarize the lyrics" instead of "Describe the scene" to get caption-like text, but this also did not yield good results. Instead, we then tried to have the Completions model pick out the main points, actions, objects, synonyms, etc. Some prompts we tried were "Describe the keywords", "Illustrate the action", "Rewrite as a stable diffusion prompt", "Paint the scene", "Main action of the scene below", "Rewrite in fantasy style", "Synonyms", and "Remove connecting words". The best results came from the combination of "Synonyms:" and "Remove connecting words".

```
Remember when the twinkling stars at night told you reindeer
were in flight and jolly Santa Claus was on his way?

Synonyms:

Remove connecting words:

Twinkling, stars, night, reindeer, flight, jolly, Santa, Claus,
way.
```

Fig. 3 Before (left) and after (right) the preprocessing, Can You Remember by Kellie Coffey

3-2. Text to Image

At first, we tried to adapt the paper: Universal guidance for diffusion models [2] to our project. In this paper, the authors propose a universal guidance algorithm that enables controlling diffusion models by arbitrary guidance modalities, including Human Identity, Segmentation Maps, Object Location, Image Style, and CLIP. It provides a GitHub repo, however, the codebase is not complete (the backward guidance is not implemented), so we follow the algorithm in the paper to implement this. Given the lyrics "*I walked through the door with you, the air was cold. But something about it felt like home somehow.*" and a random scenery photo as the input style, the model generates the following:

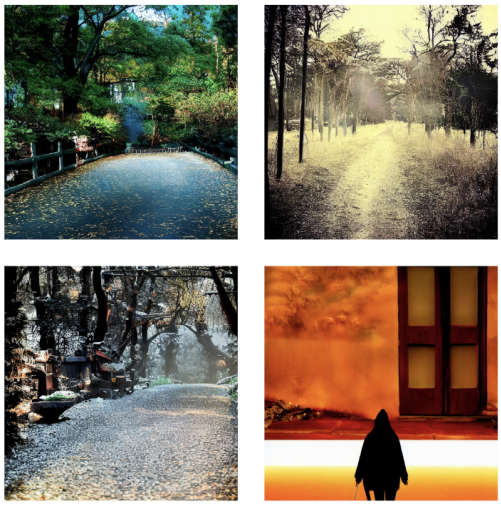


Fig. 4(a) forward guidance

Fig. 4(b) forward+backward guidance

It is hard to decide which one is better. My personal favorites are the top-right image for Fig.4(a), the top-right, and the bottom-left image for Fig.4(b). Because they capture the bleakness and vibe of this song (All too well by Taylor Swift).

But it takes hours even with solely forward guidance to generate an image. If we added the implemented backward guidance, it takes days for image generation.

To speed up the experimental cycle, we decided to use another paper [5], where the technical parts are already covered in Section 2.2. If we use the original codebase, i.e. without content feedback, the generated images in Fig.5(a) do not have a similar style, let alone be consistent. On the other hand, images depicted in Fig.5(b) have a more consistent style, and they also align with the text better. We ignore the aesthetic side for now, as we will discuss further in the following experiments.

Fig. 5(a) Generated images without *content* feedback. The lyrics are from "when will my life begin".

Fig. 5(b) Generated images with *content* feedback. The lyrics are from "when will my life begin".

We experiment with lots of songs, some work well while some don't have a consistent style. We suspect that aesthetic embedding does not play a huge part in some songs. We decided to reinforce it with written style, i.e. specify the style at the end of each line of lyrics. The result is shown in Fig.6. We can clearly see that Fig.6(b) is more consistent and aligned with the song when we add the style in text.

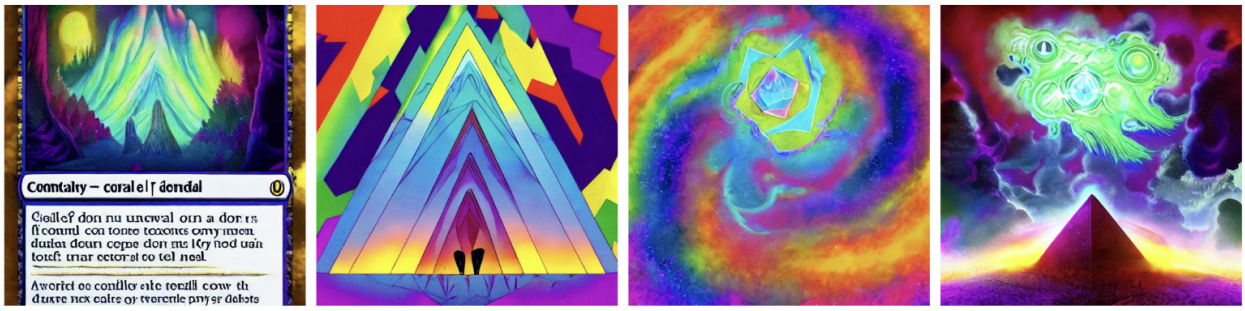


Fig. 6(a) Generated images without written style in each line of *Let it Go*.

Fig. 6(b) Generated images with written style in each line of *Let it Go*. In this case, the added style is "hyperrealistic".

4. Reflection

As mentioned in the Process section, the Completions model isn't the best at blocking the lyrics in the most optimal way. Although the model is relatively good at blocking into sentences, it seems to be worse at determining sentence boundaries. In other words, a single sentence under the Completions model might actually be 2 or 3 if annotated by a human. Furthermore, when playing around with the model during the experimentation phase, it did seem somehow that the model is easily confused when most of the lyrics are standalone line by line. The model will still try to concatenate these standalone lines. Another drawback we found using the Completions model, also mentioned in the Process sections, is that it isn't very stable with the prompt outputs. For sentence blocking, occasionally the model will return the lyrics as is the line for line, and other times it will return the lyrics as one giant run-on sentence. For keyword extraction (the second part of preprocessing), sometimes we get extraneous words or we'll get actual synonyms for some of the words in the sentence. Using only "Synonyms:" or "Remove connecting words:" doesn't provide good results, but when used together it usually does fine with extracting keywords. Occasionally, we get two lines of output where the extraneous line is filled with synonyms that don't appear in the actual sentence. A possible next step could be to build a more stable system that can generate sentences and keywords more reliably. However, one thing to note is that even if we improve the ability to generate minimal-spanning sentences from lyrics and extracted keywords, it may not necessarily improve the image generation process despite the improvement we did see. This falls in the realm of prompt engineering and a literature review on existing studies into prompt engineering for AI-generative models show that it is difficult to find one strong template to fit all purposes.

Another interesting thing we observed about the results was that our model tends to generate the same female figure in relatively similar poses across different sentences even when the sentences don't contain any information about a female figure. A potential guess is that the corpus used to train the model could be unbalanced and dominated by certain figures or features. Another possible explanation is that some gender biases, either from datasets or from the model itself, are inherited by the downstream applications using the model. One possible solution is to fine-tune the model using a new image corpus with less or barely any similar images to the commonly seen female figure, and run the tuned model with the same lyrics and compare the difference if any.

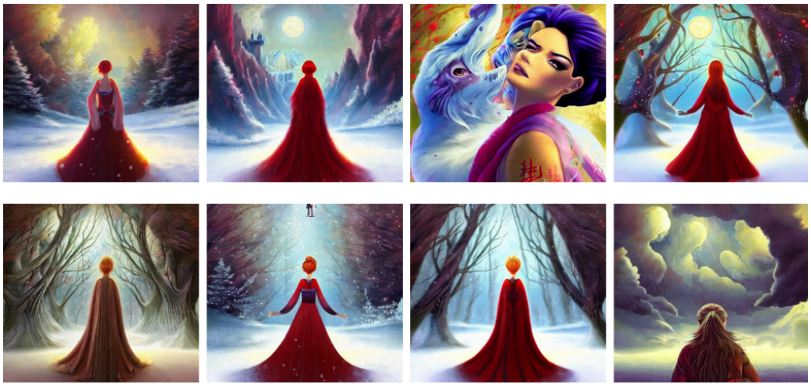
RESULT

Below are the final results generated by the edited diffusion model, with the input lyric being processed by our preprocessing method, as described above. To make sure the model is applicable to several kinds of lyrics, we selected multiple songs of different musical styles, including Let It Go from Frozen (power ballad), Can You Remember by Kellie Coffey (country music), Memories by Within Temptation (rock), You

Raise Me Up by Westlife (operatic pop) and Love Story by Taylor Swift (mid-tempo country pop). The results are displayed per song.

For each song, we only use part of the entire lyrics (at least two sections) for the image generation, to avoid repetitive sections and to control the length of the input text.

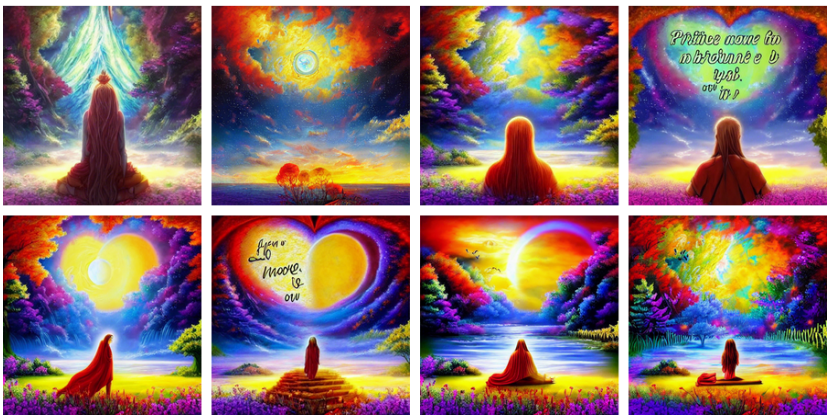
Let It Go - Frozen



```
The snow glows white on the mountain tonight
Not a footprint to be seen
A kingdom of isolation
And it looks like I'm the queen
The wind is howling like this swirling storm inside
Couldn't keep it in, heaven knows I tried
Don't let them in, don't let them see
Be the good girl you always have to be
Conceal, don't feel, don't let them know
Well, now they know
Let it go, let it go
Can't hold it back anymore
Let it go, let it go
Turn away and slam the door
I don't care what they're going to say
Let the storm rage on
The cold never bothered me anyway
```

Can You Remember - Kellie Coffey

Memories - Within Temptation



You Raise Me Up - Westlife

When I am down and, oh my soul, so weary
When troubles come and my heart burdened be
Then, I am still and wait here in the silence
Until You come and sit awhile with me
You raise me up, so I can stand on mountains
You raise me up, to walk on stormy seas
I am strong, when I am on your shoulders
You raise me up to more than I can be

Love Story - Taylor Swift

Add prompt: Storybook style
Oil painting, hyperrealistic does not work well with "love story"...

```
We were both young when I first saw you
I close my eyes and the flashback starts
I'm standin' there
On a balcony in summer air
See the lights, see the party, the ball gowns
See you make your way through the crowd
And say, "Hello"
Little did I know
That you were Romeo, you were throwin' pebbles
And my daddy said, "Stay away from Juliet"
And I was cryin' on the staircase
Beggin' you, "Please don't go, " and I said
Romeo, take me somewhere we can be alone
I'll be waiting, all there's left to do is run
You'll be the prince and I'll be the princess
It's a love story, baby, just say, "Yes"
```

CODE

We submit the code in the zip file. A public GitHub repo (https://github.com/sophia1488/Lyrics-to-images) for your reference.

REFERENCE

[1] Hyeonho Jeong, Gihyun Kwon, and Jong Chul Ye. 2023. Zero-shot Generation of Coherent Storybook from Plain Text Story using Diffusion Models

[2] Arpit Bansal, Hong-Min Chu, Avi Schwarzschild, Soumyadip Sengupta, Micah Goldblum, Jonas Geiping, and Tom Goldstein. 2023. Universal guidance for diffusion models.

[3] Completions API by OpenAI. https://platform.openai.com/docs/api-reference/completions

[4] Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, Devi Parikh, Sonal Gupta, and Yaniv Taigman. 2022. Make-a-video: Text-to-video generation without text-video data.

[5] Victor Gallego. 2022. Personalizing text-to-image generation via aesthetic gradients.