Learning Objectives

Learners will be able to...

- **Develop proficiency in conditional image generation techniques**
- **Demonstrate an understanding of prompt engineering for natural language processing**
- **Apply GPT-3 as a prompt generator in natural language processing tasks**
- **Evaluate the limitations of image-to-text applications and propose potential solutions**

info

Make Sure You Know

You are familiar with Python.

Limitations

This is a gentle introduction. So there is a little bit of Python programming. The information might not be the most up to date as OpenAI releases new features.

Understanding Prompt Engineering

A crucial aspect of generating relevant and visually appealing images using the DALL-E 2 API is crafting effective prompts. In this section, we will learn how to create more effective prompts to guide the DALL-E 2 API in generating the desired images.

- **Be clear and descriptive**: Use specific adjectives, nouns, and phrases to describe the desired image. For example, instead of “a dog,” try “a golden retriever playing in a park.”
- **Experiment with different prompt structures**: Change the order of words, use synonyms, or rephrase the prompt to achieve different results. For example, instead of “a futuristic city,” try “a city in the future” or “a city with advanced technology.”
- **Provide context**: Add context or additional information to the prompt to guide the model. For example, instead of “a sad robot,” try “a robot expressing sadness in its facial features.”
- **Test multiple prompts**: Generate images using different prompts and choose the one that produces the best results.

Combining DALL-E 2 API with GPT-3

For Text-to-Image and Image-to-Text Applications

In this section, we will discuss how to combine the DALL-E 2 API with GPT-3 to create powerful text-to-image and image-to-text applications. By leveraging the strengths of both models, you can build more versatile and interactive applications.

Text-to-Image Application

A text-to-image application translates a given text prompt into an image. You can use GPT-3 to refine the text prompts before passing them to the DALL-E 2 API for image generation. Here's an example:

Use GPT-3 to create a more descriptive text prompt based on user input:

```
import os
import openai
from PIL import Image,ImageOps
from io import BytesIO
import secret

# Set API key and prompt
openai.api_key = secret.key

user_input = "Create an image of a car"

# Generate more descriptive text with GPT-3
response = openai.Completion.create(
    engine="text-davinci-002",
    prompt=f"Create a more descriptive scene based on this user
        input: '{user_input}'.",
    max_tokens=50,
    n=1,
    stop=None,
    temperature=0.7,
)

descriptive_text = response.choices[0].text.strip()
print(descriptive_text)
```

The code above gets a user input, then uses GPT-3 to expand on that prompt. How to go about expanding on the prompt is by writing the following prompt : `Create a more descriptive scene based on this user input:.`

Now let's try creating and saving the image that we would generate from our new and more detailed prompt.

```
# Generate the base image
response = openai.Image.create(
    prompt=descriptive_text,
    n=1,
    size="512x512"
)

image_url = response['data'][0]['url']

img_data = requests.get(image_url).content
with open('base_img.jpg', 'wb') as handler:
    handler.write(img_data)
```

Once you found an image or prompt you are comfortable working with. We are going to save on some tokens and copy the prompt being printed and set our variable `descriptive_text` equal to it. From that point please comment out GPT code.

Below is an example of a descriptive text generated by the AI. Feel free to run it.

```
descriptive_text = "The car is a sleek, silver convertible with
        bright red leather seats. It's parked in front of a
        beautiful mansion with a fountain in the middle."
```

Temperature

In our code, we used the `temperature` with the value of `0.7`. Temperature is a parameter that influences the randomness of the image generation process. A higher temperature value results in more diverse and potentially more creative prompts, while a lower temperature value produces more conservative and focused results.

Controlling Image Attributes

Conditional Image Generation with DALL-E 2 API

Conditional image generation using DALL-E 2 API allows developers to control various attributes of the generated images by specifying them in the prompt. This powerful feature enables users to create images with desired characteristics, such as specific styles, colors, or content. In this page, we will explore how to control image attributes using the DALL-E 2 API, providing examples and explanations of how the prompts can be crafted to achieve specific results.

As we go through the examples. We are going to keep the libraries we import and our API key.

```
import os
import openai
from PIL import Image,ImageOps
from io import BytesIO
import requests
import secret

# Set API key and prompt
openai.api_key = secret.key
```

At the bottom of our page will be the code to generate, save and visualize our images.

```
image_url=(response['data'][0]['url'])
img_data = requests.get(image_url).content
with open('att.jpg', 'wb') as handler:
    handler.write(img_data)
```

We want to keep that. All the following examples should go in the middle of the two codes.

Example 1: Controlling style

```
prompt = "A cubist painting of a cat"

response = openai.Image.create(
    prompt=prompt,
    n=1,
    size="512x512"
)
```

In this example, we specify the desired style of the generated image (“cubist painting”) along with the subject (“a cat”). By including the style in the prompt, DALL-E 2 will generate an image of a cat in the cubist painting style.

Example 2: Controlling color

```
prompt = "A red sports car on a blue background"

response = openai.Image.create(
    prompt=prompt,
    n=1,
    size="512x512"
)
```

In this example, we specify the desired colors of the generated image by including “red sports car” and “blue background” in the prompt. DALL-E 2 will generate an image of a red sports car with a blue background, as requested.

Example 3: Controlling content

```
prompt = "A futuristic city skyline with flying cars and
        skyscrapers"

response = openai.Image.create(
    prompt=prompt,
    n=1,
    size="512x512"
)
```

In this example, we specify the desired content of the image by describing a futuristic city skyline with flying cars and skyscrapers. DALL-E 2 will generate an image that matches the provided description, creating a scene

with the specified elements.

Controlling image attributes using the DALL-E 2 API is a powerful way to generate images that match specific requirements or artistic visions. By crafting prompts that include the desired attributes, developers can guide DALL-E 2 to create images with specific styles, colors, and content. Experimenting with different prompts and combinations of attributes can lead to diverse and creative results.

Image-to-Text Application

For an image-to-text application, you can use DALL-E 2 to generate a variety of images and then use GPT-3 to describe the images or provide captions.

Use DALL-E 2 API to generate an image based on a text prompt:

```
# Generate the base image
response = openai.Image.create(
    prompt="cute kitten playing with a ball of yarn",
    n=1,
    size="512x512"
)

image_url = response['data'][0]['url']
print(image_url)
img_data = requests.get(image_url).content
with open('kitten.jpg', 'wb') as handler:
    handler.write(img_data)
```

Feel free to copy and paste the image URL to compare. Before moving on please comment out the previous code. You can get the `image_url`from the try it button above if necessary.
Use GPT-3 to generate a description or caption for the generated image:

```
# Generate a description with GPT-3
response = openai.Completion.create(
    engine="text-davinci-002",
    prompt=f"Describe the following image: [img]{image_url}
        [/img]",
    max_tokens=50,
    n=1,
    stop=None,
    temperature=0.7,
)

image_description = response.choices[0].text.strip()
print(image_description)
```

It's important to be aware that image-to-text applications may not always provide the most accurate descriptions or captions. There are several reasons for this:

Lack of visual context: GPT-3 is a text-based model and does not have direct access to the actual image content. Instead, it relies on the image URL, which may not always provide enough context for the model to generate a precise description.

Ambiguity in images: Images can often contain ambiguous or abstract elements, making it difficult for the model to understand the exact content or context of the image. This can result in inaccuracies or vague descriptions.

Model limitations: While GPT-3 is an advanced language model, it is not perfect and has its limitations. It may not always be able to generate the most accurate or relevant descriptions, especially for images with complex or highly specialized content.

To improve the accuracy of image-to-text applications, you can consider using specialized computer vision models, such as **OpenAI's CLIP**, which is designed specifically for tasks involving both text and images. These models are better equipped to handle the challenges associated with understanding and describing visual content.

Coding Exercise

Generate Image with GPT and DALL-E API

Objective: Write a Python function `generate_image(prompt)` that uses OpenAI's GPT API to enhance a user's text prompt and DALL-E API to generate an image based on that enhanced text, then saves this image as `my_img.png`.

Function Requirements:

Your function `generate_image(prompt)` should accomplish the following steps:

1. Accept a user's prompt as an argument.
2. Refine this prompt by passing it to the GPT API.
3. Generate an image from the refined prompt using the DALL-E API.
4. Save the generated image as `my_img.png`.

Your Python function must look like the template below:

```python
def generate_image(prompt):
    """
    This function accepts a user's prompt, enhances it using the
        GPT API, then generates an image from the enhanced
        prompt using the DALL-E API, and saves it as
        `my_img.png`.

    Parameters:
        prompt (str): The initial user's prompt.

    Returns:
        None
    """

    # Step 1: Pass the user's prompt to the GPT API to get a
        refined prompt.

    # Step 2: Pass the refined prompt to the DALL-E API to get
        the generated image.

    # Step 3: Save the generated image as `my_img.png`.

    return None
```

Hint: You can test your code by calling the function with some prompts. **But** before submitting, make sure to delete the function call.