Task 3 Report: Image Captioning with AI

Introduction

For Task 3 of my CodSoft internship, I used an AI model to generate a caption for an image I uploaded. The image features a blue BMW car parked next to a grassy area with autumn trees.

Image Description

The image shows a blue BMW car, likely a 3 Series model, parked on a paved surface beside a grassy area. The car has a sleek design with a black kidney grille, sharp LED headlights, and the BMW logo in the center. The background includes green grass and trees with autumn-colored leaves, suggesting a fall setting.

AI-Generated Caption

Using a pre-trained image captioning model in Google Colab, I generated this caption for my image:
"a blue car is parked on the street."

How I Did It

I used Google Colab to run a notebook with a pre-trained image captioning model. I uploaded my photo of the BMW car, ran the code, and the model produced a caption. The code and output are shown below.

Screenshots

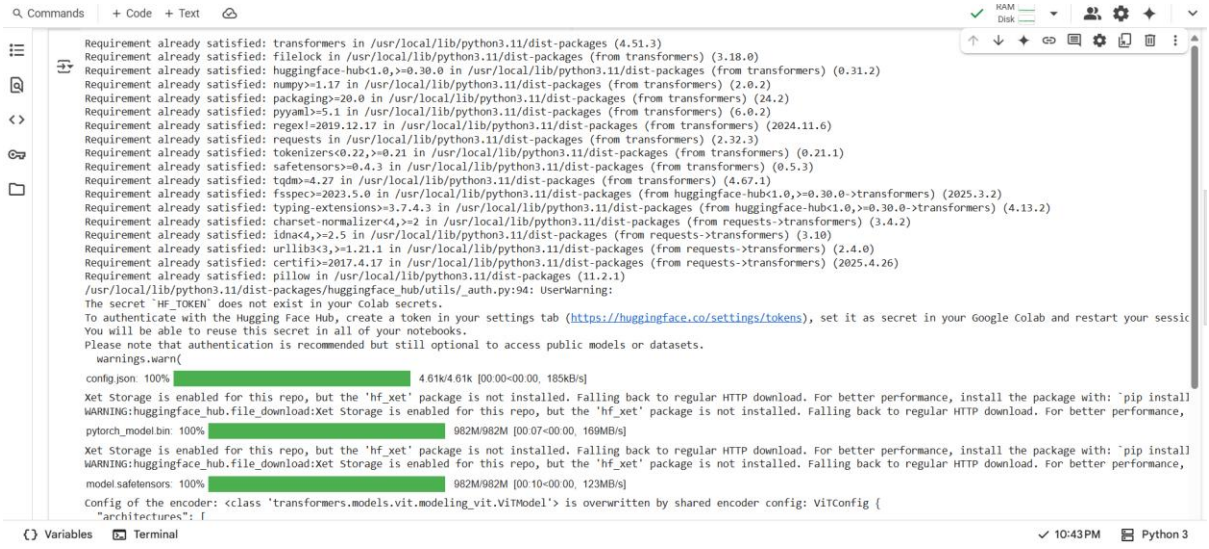
- **Figure 1: Screenshot of the code (Part 1).**
  This shows the first part of the code I ran.

```
# Step 1: Install the necessary tools (only need to run this once)
!pip install transformers
!pip install pillow

# Step 2: Load the tools we need
from transformers import VisionEncoderDecoderModel, ViTFeatureExtractor, AutoTokenizer
from PIL import Image
import requests

# Step 3: Set up the pre-trained image captioning model
model = VisionEncoderDecoderModel.from_pretrained("nlpconnect/vit-gpt2-image-captioning")
feature_extractor = ViTFeatureExtractor.from_pretrained("nlpconnect/vit-gpt2-image-captioning")
tokenizer = AutoTokenizer.from_pretrained("nlpconnect/vit-gpt2-image-captioning")

# Step 4: Define a function to generate captions
def generate_caption(image_path):
    # Open the image from your file
    image = Image.open(image_path).convert("RGB")

    # Prepare the image for the model
    pixel_values = feature_extractor(images=image, return_tensors="pt").pixel_values

    # Generate the caption
    output_ids = model.generate(pixel_values)
    caption = tokenizer.decode(output_ids[0], skip_special_tokens=True)

    return caption

# Step 5: Upload your image and generate a caption
from google.colab import files
uploaded = files.upload()  # This opens a window to upload your image

# Step 6: Get the caption for the uploaded image
```

- **Figure 2: Screenshot of the code (Part 2).**
  This shows the second part of the code.

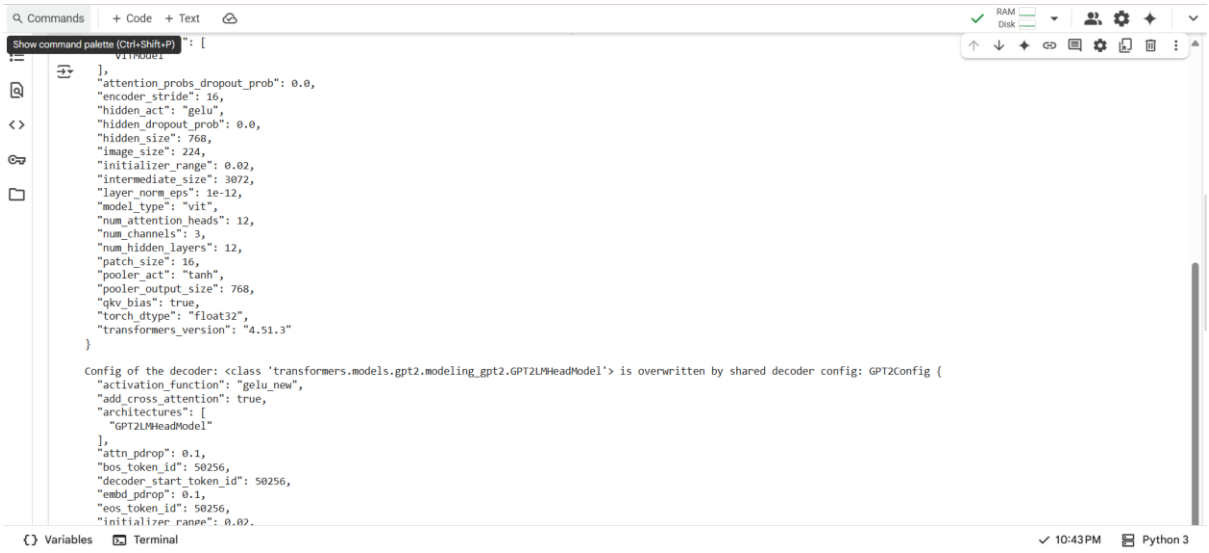
```
from transformers import VisionEncoderDecoderModel, ViTFeatureExtractor, AutoTokenizer
from PIL import Image
import requests

# Step 3: Set up the pre-trained image captioning model
model = VisionEncoderDecoderModel.from_pretrained("nlpconnect/vit-gpt2-image-captioning")
feature_extractor = ViTFeatureExtractor.from_pretrained("nlpconnect/vit-gpt2-image-captioning")
tokenizer = AutoTokenizer.from_pretrained("nlpconnect/vit-gpt2-image-captioning")

# Step 4: Define a function to generate captions
def generate_caption(image_path):
    # Open the image from your file
    image = Image.open(image_path).convert("RGB")

    # Prepare the image for the model
    pixel_values = feature_extractor(images=image, return_tensors="pt").pixel_values

    # Generate the caption
    output_ids = model.generate(pixel_values)
    caption = tokenizer.decode(output_ids[0], skip_special_tokens=True)

    return caption

# Step 5: Upload your image and generate a caption
from google.colab import files
uploaded = files.upload()  # This opens a window to upload your image

# Step 6: Get the caption for the uploaded image
for filename in uploaded.keys():
    caption = generate_caption(filename)
    print(f"Image: {filename}")
    print(f"Caption: {caption}")
```

- **Figure 3: Screenshot of the output (Part 1).**
  This shows the beginning of the output.



```
Requirement already satisfied: transformers in /usr/local/lib/python3.11/dist-packages (4.51.3)
Requirement already satisfied: filelock in /usr/local/lib/python3.11/dist-packages (from transformers) (3.18.0)
Requirement already satisfied: huggingface-hub<1.0,>=0.30.0 in /usr/local/lib/python3.11/dist-packages (from transformers) (0.31.2)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.11/dist-packages (from transformers) (2.0.2)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.11/dist-packages (from transformers) (24.2)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.11/dist-packages (from transformers) (6.0.2)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.11/dist-packages (from transformers) (2024.11.6)
Requirement already satisfied: requests in /usr/local/lib/python3.11/dist-packages (from transformers) (2.32.3)
Requirement already satisfied: tokenizers<0.22,>=0.21 in /usr/local/lib/python3.11/dist-packages (from transformers) (0.21.1)
Requirement already satisfied: safetensors>=0.4.3 in /usr/local/lib/python3.11/dist-packages (from transformers) (0.5.3)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.11/dist-packages (from transformers) (4.67.1)
Requirement already satisfied: fsspec>=2023.5.0 in /usr/local/lib/python3.11/dist-packages (from huggingface-hub<1.0,>=0.30.0->transformers) (2025.3.2)
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.11/dist-packages (from huggingface-hub<1.0,>=0.30.0->transformers) (4.13.2)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/dist-packages (from requests->transformers) (3.4.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-packages (from requests->transformers) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/dist-packages (from requests->transformers) (2.4.0)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.11/dist-packages (from requests->transformers) (2025.4.26)
Requirement already satisfied: pillow in /usr/local/lib/python3.11/dist-packages (11.2.1)
/usr/local/lib/python3.11/dist-packages/huggingface_hub/utils/_auth.py:94: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/settings/tokens), set it as secret in your Google Colab and restart your sessio
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or datasets.
  warnings.warn(
config.json: 100% 4.61k/4.61k [00:00<00:00, 185kB/s]
Xet Storage is enabled for this repo, but the 'hf_xet' package is not installed. Falling back to regular HTTP download. For better performance, install the package with: `pip install
WARNING:huggingface_hub.file_download:Xet Storage is enabled for this repo, but the 'hf_xet' package is not installed. Falling back to regular HTTP download. For better performance,
pytorch_model.bin: 100% 982M/982M [00:07<00:00, 169MB/s]
Xet Storage is enabled for this repo, but the 'hf_xet' package is not installed. Falling back to regular HTTP download. For better performance, install the package with: `pip install
WARNING:huggingface_hub.file_download:Xet Storage is enabled for this repo, but the 'hf_xet' package is not installed. Falling back to regular HTTP download. For better performance,
model.safetensors: 100% 982M/982M [00:10<00:00, 123MB/s]
Config of the encoder: <class 'transformers.models.vit.modeling_vit.ViTModel'> is overwritten by shared encoder config: ViTConfig {
  "architectures": [
```

- **Figure 4: Screenshot of the output (Part 2).**
  This continues the output.



- **Figure 5: Screenshot of the output (Part 3).**
  This shows more of the output.

- **Figure 6: Screenshot of the output (Part 4).**
  This shows the final output with the caption.

Reflection

This task taught me how AI can create captions from images. My caption was accurate but missed some details like the trees. I enjoyed using Colab and seeing the model work with my photo.