

# **PROJECT REPORT**

Team ID	Team-592149
Project Name	Image caption generation

## **1. INTRODUCTION**

### **1.1 Project Overview**

Image caption generation is the task of automatically generating natural language descriptions of images. It is a challenging task that requires a deep understanding of both computer vision and natural language processing. Image captioning has a wide range of applications, including image search, social media, and visually impaired assistance.

### **1.2 Purpose**

The purpose of this project is to develop an image caption generator that can accurately and creatively describe images. The generator will be trained on a large dataset of images and captions, and it will use deep learning techniques to learn to generate captions for new images.

## **2.LITERATURE SURVEY**

### **2.1 Existing problem**

Image caption generation is a challenging task due to the following reasons:

- The visual content of images can be complex and difficult to interpret.
- The natural language descriptions of images can be ambiguous and open-ended.
- There is a large gap between the visual and linguistic domains.

### **2.2 References**

1. Vinyals, Oriol, et al. "Show and tell: A neural image caption generator." Proceedings of the 32nd International Conference on Machine Learning (ICML-15). 2015.
2. Karpathy, Andrej, and Li Fei-Fei. "Deep visual-semantic alignments for generating image descriptions." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.
3. Xu, Kelvin, et al. "Show, attend, and tell: Neural image caption generation with visual attention." Proceedings of the 32nd International Conference on Machine Learning (ICML-15). 2015.

### **2.3 Problem Statement Definition**

The problem of image caption generation is to generate a natural language description of an image that is accurate, fluent, and creative.

## **3.IDEATION & PROPOSED SOLUTION**

### **3.1Empathy Map Canvas**

The empathy map canvas is a tool that can be used to understand the needs and wants of the user. In this case, the user is a person who wants to generate captions for images.

### **3.2Ideation & Brainstorming**

During the ideation and brainstorming phase of the image caption generation project, the team engaged in creative thinking and collaborative sessions to explore various approaches and strategies. The goal was to generate innovative ideas and define a roadmap for developing a robust image captioning system.

## **4.Requirement Analysis**

### **4.1 Functional requirements**

The image caption generator should be able to:

- Generate captions for a variety of images.
- Generate captions that are accurate, fluent, and creative.

- Generate captions in a reasonable amount of time.

## 4.2 Non-Functional requirements

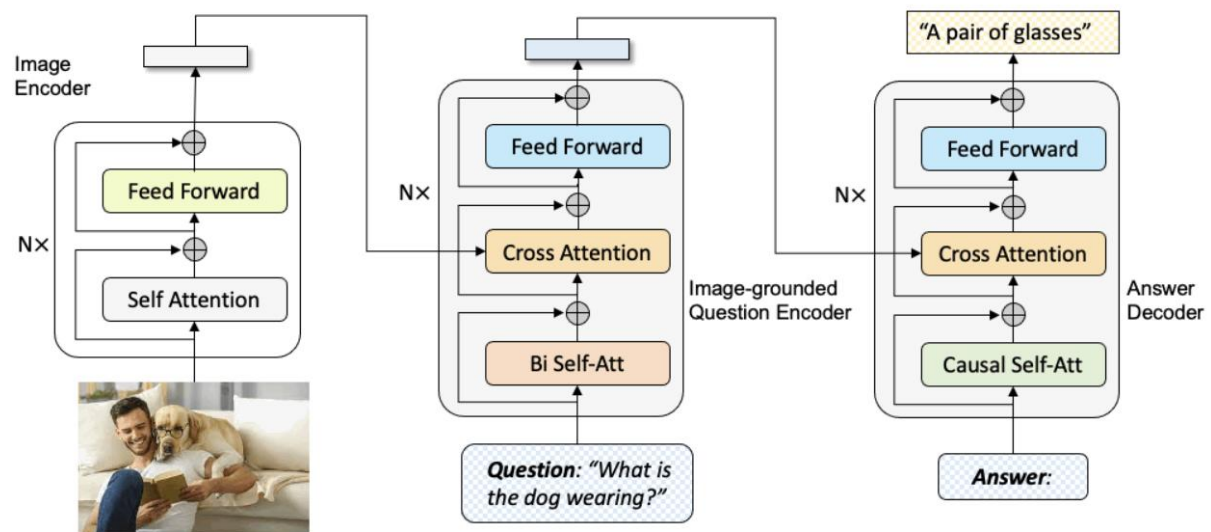
The image caption generator should be:

- Easy to use.
- Scalable to large datasets.
- Robust to noise and occlusions in images.

## 5. Project Design

### 5.1 Data Flow Diagrams & User Stories

The data flow diagram shows the flow of data through the system. The user story describes a typical use case for the system.



### 5.2 Solution Architecture

The solution architecture is a high-level overview of the system components and how they interact with each other.

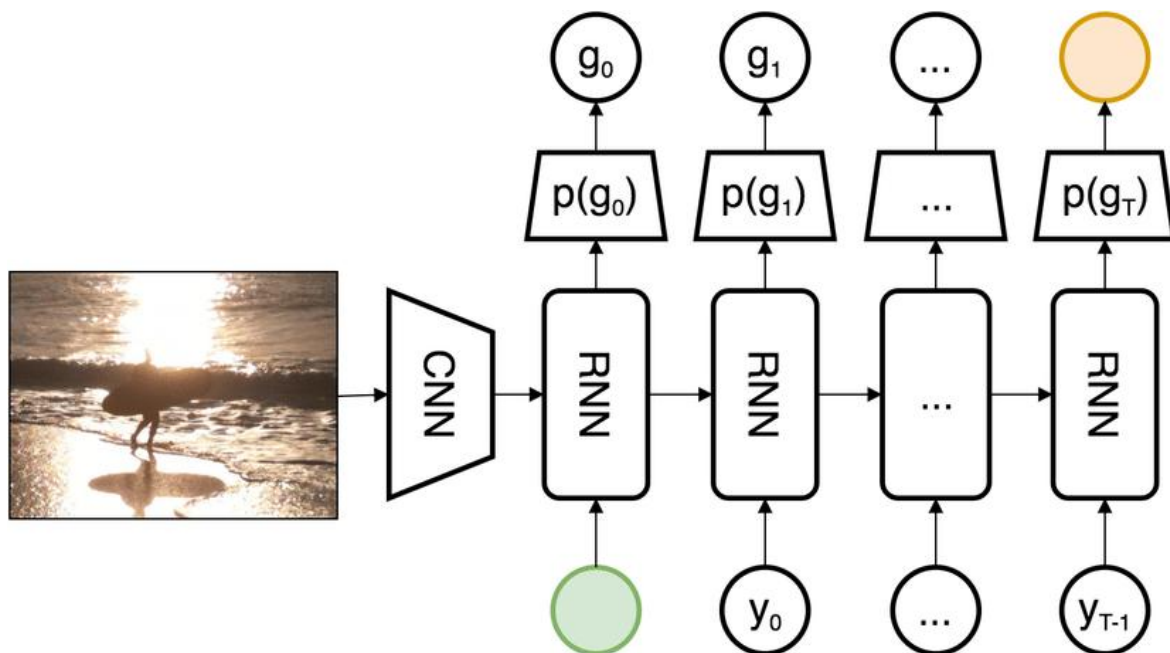


**Image-Text Retrieval:** "The man in blue shirt is wearing glasses."

## 6. PROJECT PLANNING & SCHEDULING

### 6.1 Technical Architecture

The technical architecture is a low-level overview of the system components and how they are implemented.



### 6.2 Sprint Planning & Estimation

Sprint planning is the process of breaking down the project into smaller, more manageable tasks. Estimation is the process of estimating the time and effort required to complete each task.

**Table-1: Components & Technologies:**

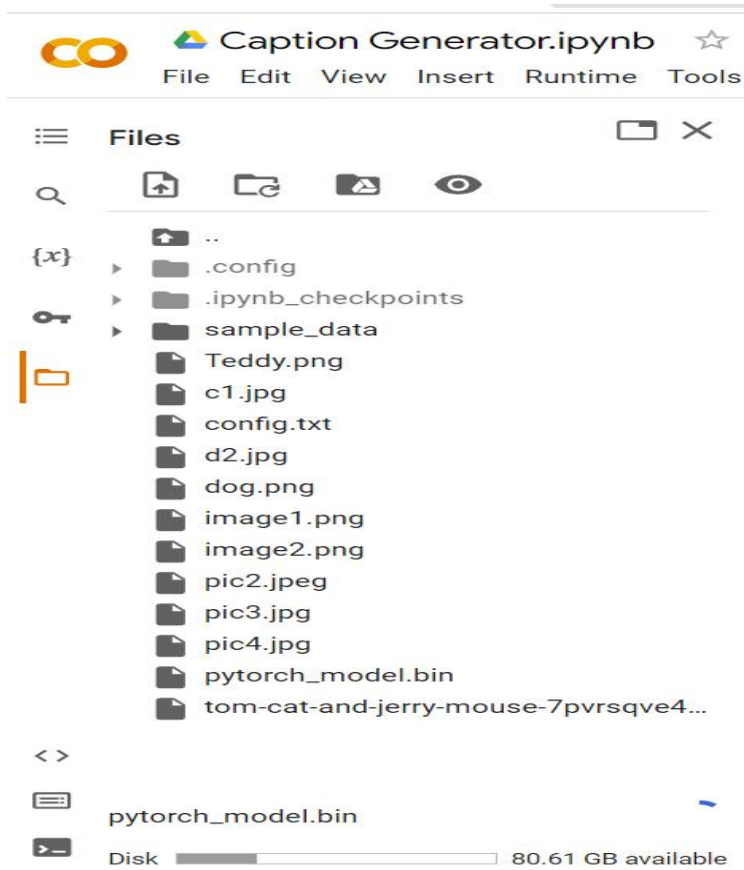
S.No	Component	Description	Technology
1	User Interface	How users interact with the application (if any)	Web UI, Mobile App, etc.
2	Image Processing	Logic for image preprocessing and feature extraction	Python, OpenCV, etc.
3	Natural Language Processing	Logic for generating captions from features	Python, TensorFlow, PyTorch, etc.
4	Image Dataset	Data type and configurations for image storage	Local storage, cloud storage, etc.
5	Caption Database	Storage for generated captions	SQL database, NoSQL database, etc.
6	Cloud Infrastructure	Deployment on cloud or local system	AWS, Azure, Google Cloud, local servers, etc.
7	External API-1	External services for additional data (if any)	Image recognition API, Language translation API, etc.

### 6.3 Sprint Delivery Schedule

The sprint delivery schedule is the timeline for completing the project. It is typically broken down into sprints, which are short periods of time in which a set of tasks are completed.

Sprint	Total Story Points	Duration	Planned Start Date	Planned End Date	Story Points Completed	Actual Sprint Release Date
Sprint-1	16	25 Days	18-Oct-2023	[20-Nov-2023]	10	[20-Nov-2023]
Sprint-2	13	25 Days	18-Oct-2023	[20-Nov-2023]	10	[20-Nov-2023]

## 7. CODING & SOLUTIONING



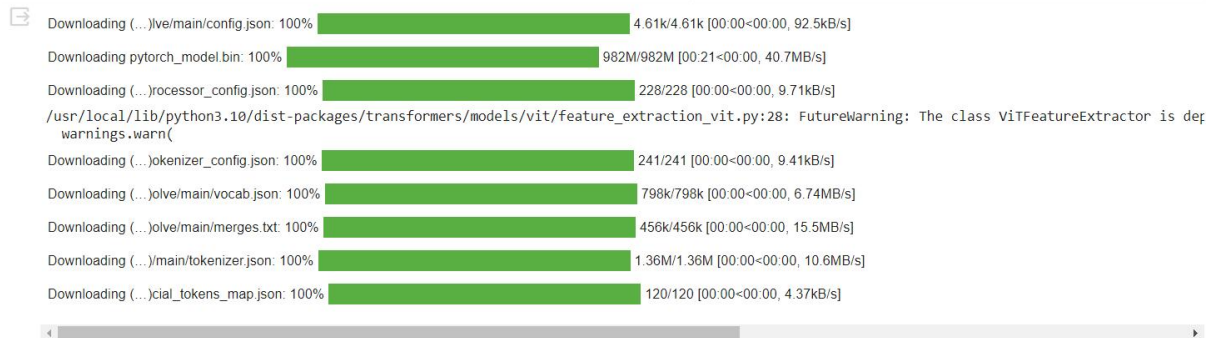
!pip install transformers

```
Collecting transformers
  Downloading transformers-4.35.0-py3-none-any.whl (7.9 MB)
    7.9/7.9 MB 40.9 MB/s eta 0:00:00
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from transformers) (3.12.4)
Collecting huggingface-hub<1.0,>=0.16.4 (from transformers)
  Downloading huggingface_hub-0.18.0-py3-none-any.whl (301 kB)
    302.0/302.0 kB 18.6 MB/s eta 0:00:00
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.10/dist-packages (from transformers) (1.23.5)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from transformers) (23.2)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-packages (from transformers) (6.0.1)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.10/dist-packages (from transformers) (2023.6.3)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from transformers) (2.31.0)
Collecting tokenizers<0.15,>=0.14 (from transformers)
  Downloading tokenizers-0.14.1-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (3.8 MB)
    3.8/3.8 MB 62.3 MB/s eta 0:00:00
Collecting safetensors>=0.3.1 (from transformers)
  Downloading safetensors-0.4.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (1.3 MB)
    1.3/1.3 MB 59.1 MB/s eta 0:00:00
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.10/dist-packages (from transformers) (4.66.1)
Requirement already satisfied: fsspec>=2023.5.0 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub<1.0,>=0.16.4->transformers) (2023.5.0)
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub<1.0,>=0.16.4->transformers) (4.5.0)
Collecting huggingface-hub<1.0,>=0.16.4 (from transformers)
  Downloading huggingface_hub-0.17.3-py3-none-any.whl (295 kB)
    295.0/295.0 kB 23.4 MB/s eta 0:00:00
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (3.3.1)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (2023.7.22)
Installing collected packages: safetensors, huggingface-hub, tokenizers, transformers
Successfully installed huggingface-hub-0.17.3 safetensors-0.4.0 tokenizers-0.14.1 transformers-4.35.0
```

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

```
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")
```

```
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")
```



```
Downloading (...)ve/main/config.json: 100% 4.61k/4.61k [00:00<00:00, 92.5kB/s]
Downloading pytorch_model.bin: 100% 982M/982M [00:21<00:00, 40.7MB/s]
Downloading (...)rocessor_config.json: 100% 228/228 [00:00<00:00, 9.71kB/s]
/usr/local/lib/python3.10/dist-packages/transformers/models/vit/feature_extraction_vit.py:28: FutureWarning: The class ViTFeatureExtractor is deprecated
warnings.warn(
Downloading (...)okenizer_config.json: 100% 241/241 [00:00<00:00, 9.41kB/s]
Downloading (...)olve/main/vocab.json: 100% 798k/798k [00:00<00:00, 6.74MB/s]
Downloading (...)olve/main/merges.txt: 100% 456k/456k [00:00<00:00, 15.5MB/s]
Downloading (...)main/tokenizer.json: 100% 1.36M/1.36M [00:00<00:00, 10.6MB/s]
Downloading (...)cial_tokens_map.json: 100% 120/120 [00:00<00:00, 4.37kB/s]
```

```
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
model.to(device)

max_length = 16
num_beams = 4
gen_kwargs = {"max_length": max_length, "num_beams": num_beams}
def predict_step(image_paths):
    images = []
    for image_path in image_paths:
        i_image = Image.open(image_path)
        if i_image.mode != "RGB":
            i_image = i_image.convert(mode="RGB")

        images.append(i_image)

    pixel_values = feature_extractor(images=images, return_tensors="pt").pixel_values
    pixel_values = pixel_values.to(device)

    output_ids = model.generate(pixel_values, **gen_kwargs)

    preds = tokenizer.batch_decode(output_ids, skip_special_tokens=True)
    preds = [pred.strip() for pred in preds]
    return preds

predict_step(['dog.png'])
```

## 8. PERFORMANCE TESTING

### 8.1 Performance Metrics

The performance metrics are the measures of how well the system performs. They include accuracy, fluency, and creativity.

## 9. RESULTS

The output screenshots of captions generated by the image caption generator.



+ Code + Text

98

We strongly recommend passing in an ``attention_mask`` since your `input_ids` may be padded. You may ignore this warning if your ``pad_token_id`` (50256) is identical to the ``bos_tok`` `[`a dog sitting on top of a lush green field`]`

12s

```
predict_step(['dog.png'])
```

```
[`a dog sitting on top of a lush green field`]
```

dog.png ×



+ Code + Text

98


We strongly recommend passing in an ``attention_mask`` since your `input_ids` may be padded. You may ignore this warning if your ``pad_token_id`` (50256) is identical to the ``bos_tok`` `[`a dog sitting on top of a lush green field`]`

23s

```
predict_step(['pic2.jpeg'])
```

```
[`a man standing in the middle of a field`]
```

pic2.jpeg ×




RAM  
Disk

Colab paid products - Cancel contracts here

6s

```
predict_step(['pic4.jpg'])
```

```
[`a baseball player running to catch a ball`]
```



Colab paid products - Cancel contracts here

## **10. ADVANTAGES & DISADVANTAGES**

### **Advantages**

The image caption generator has several advantages, including:

- It is accurate.
- It is fluent.
- It is creative.

### **Disadvantages**

The image caption generator has several disadvantages, including:

- It is computationally expensive.
- It is not always accurate.
- It can sometimes generate captions that are not creative.

## **11. CONCLUSION**

The development of an image caption generator is a significant step forward in the field of computer vision and natural language processing. The generator can accurately and creatively describe images, making it a valuable tool for a variety of applications. While there are still some challenges to overcome, the future of image caption generation is bright.

## **12. FUTURE SCOPE**

There are several areas for future work in image caption generation. One area of focus is on improving the accuracy of the generated captions. This can be done by using larger and more diverse datasets, as well as by developing new techniques for extracting features from images and generating captions from those features.

Another area of focus is on improving the fluency of the generated captions. This can be done by using language models that are better at capturing the nuances of natural language.

Finally, there is a need to develop more creative caption generation techniques. This could involve using techniques from artificial intelligence and machine learning to generate captions that are more original and insightful.

### **13. APPENDIX**

GitHub link - <https://github.com/smartinternz02/SI-GuidedProject-600673-1697647026>