SUBMITTED REPORT

Tan Chau

1. PROBLEM EXPLANATION

The purpose of this task is to build a machine learning solution that generates the description for the list of given images. Note that the list of the given images may contain only 1 image. To support the future machine learning operation, you also need to develop a solution to store the uploaded images, their metadata and the machine learning configurations.

After reading the problems, I realize that the problem of this task is building Image Captioning solution and deploy it to Streamlit framework connected to any database to store the uploaded images, their metadata and machine learning configuration.

2. THE IMAGE CAPTIONING PROBLEM

Image captioning is the process of generating a natural language description of an image. It is a task in the field of computer vision and natural language processing. The goal of image captioning is to generate a coherent and fluent sentence that accurately describes the image content.

A person riding a motorcycle on a dirt road.
Two dogs play in the grass.
A skateboarder does a trick on a ramp.
A dog is jumping to catch a frisbee.
A group of young people playing a game of frisbee.
Two hockey players are fighting over the puck.
A little girl in a pink hat is blowing bubbles.
A refrigerator filled with lots of food and drinks.
A herd of elephants walking across a dry grass field.
A close up of a cat laying on a couch.
A red motorcycle parked on the side of the road.
A yellow school bus parked in a parking lot.

Source: https://cdn-5f733ed3c1ac190fbc56ef88.closte.com/wp-content/uploads/2017/01/image-captioning.png

An image captioning system typically consists of two main components:

- An image feature extractor: This component is responsible for extracting features from the input image, such as object locations, sizes, and colors.
- A natural language generator: This component takes the image features as input and generates a natural language description of the image.
- The generated captions are typically evaluated using metrics such as BLEU, METEOR, ROUGE, and CIDEr.

My solution for this is to use the EfficientNet as the image feature generator and to use Transformer as the natural language generator.

3. EfficientNet and Transformer

I use EfficientNet to extract features of the image. Those features then will be fed into the Transformer.

a. EfficientNet

EfficientNet is a family of convolutional neural network (CNN) models that are designed to improve the accuracy and efficiency of CNNs. It was introduced in a paper by Google AI researchers in 2019. EfficientNet models are built on top of the MobileNetV2 architecture, but they are scaled up in terms of depth, width, and resolution. These models are designed to be computationally efficient, while still achieving high accuracy on image classification tasks.

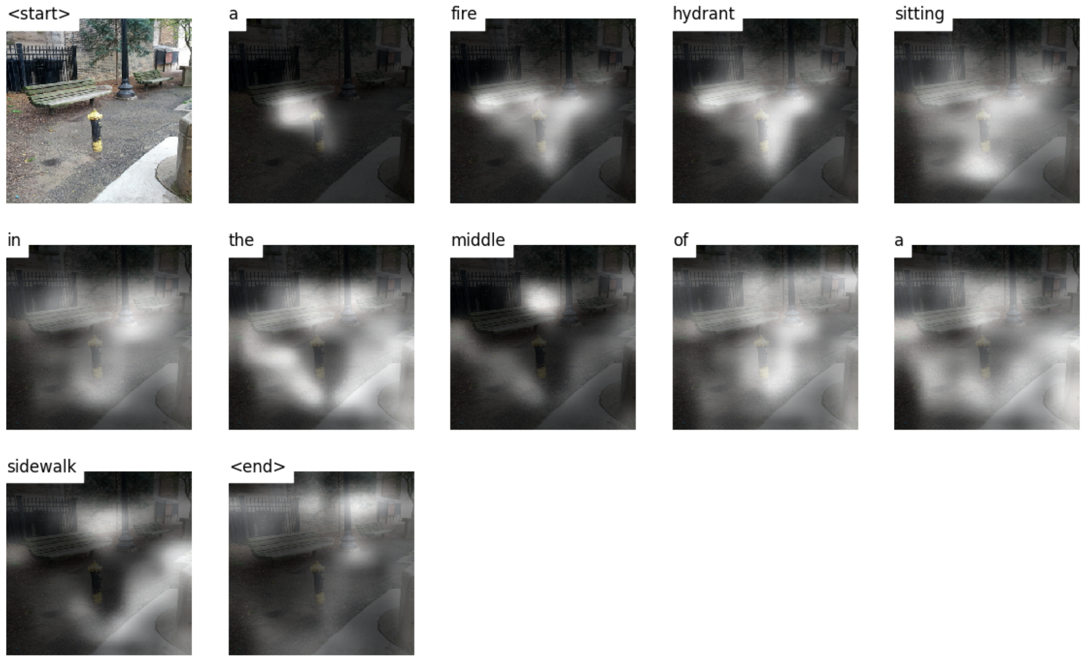
The key idea behind EfficientNet is to scale up the model in a smart and balanced way. This is done by using a combination of three scaling factors:

- Resolution scaling: Scaling the input resolution of the model by a factor, which increases the amount of information the model can process.
- Depth scaling: Scaling the depth of the model by a factor, which increases the capacity of the model.
- Width scaling: Scaling the width of the model by a factor, which increases the representational power of the model.

EfficientNet models have shown state-of-the-art performance on several image classification benchmarks, such as ImageNet, COCO, and others. They also have been applied to other tasks such as object detection, segmentation, and video classification.

b. Transformer

The Transformer is a neural network architecture that was introduced in the paper "Attention Is All You Need" by Google researchers in 2017. It is primarily used for natural language processing (NLP) tasks such as machine translation, text summarization, and language modeling.



Source: https://github.com/sgrvinod/a-PyTorch-Tutorial-to-Image-Captioning/raw/master/img/firehydrant.png

The Transformer architecture consists of an encoder and a decoder. The encoder takes in the input sequence and maps it to a high-dimensional representation. The decoder then generates the output sequence based on this representation. In Image Captioning problem. The encoder takes the features from EfficentNet and maps it to high-dimensional representation. The decoder then generates the captions based on this representation

One of the key components of the Transformer architecture is the attention mechanism. Attention allows the model to weigh the importance of different parts of the input when generating the output. Attention is computed using a multi-head mechanism, which allows the model to attend to different parts of the input simultaneously. The Transformer architecture also includes a positional encoding mechanism, which allows the model to understand the order of the input sequence.

4. **Streamlit**

Streamlit is an open-source library for building interactive machine learning and data science applications. It allows developers to create web applications quickly and easily, with minimal setup and configuration. Streamlit provides a simple, intuitive API for creating user interfaces and interactive visualizations, making it well-suited for prototyping and experimenting with machine learning models.

Streamlit allows developers to create web applications by writing Python code, which is then converted into a web application that can be run in a browser. One of the key features of Streamlit is its ability to automatically handle user input and update the application in real-time. This allows developers to easily create interactive applications that allow users to explore and experiment with data and models.

Overall, Streamlit is a powerful tool for building machine learning and data science applications, allowing developers to easily create interactive and user-friendly applications with minimal setup and configuration.

5. PostgreSQL

PostgreSQL is a powerful, open-source, object-relational database management system (ORDBMS) that is widely used for managing and querying large datasets. It is known for its reliability, robustness, and support for advanced features such as concurrency control, full-text search, and geospatial data processing.

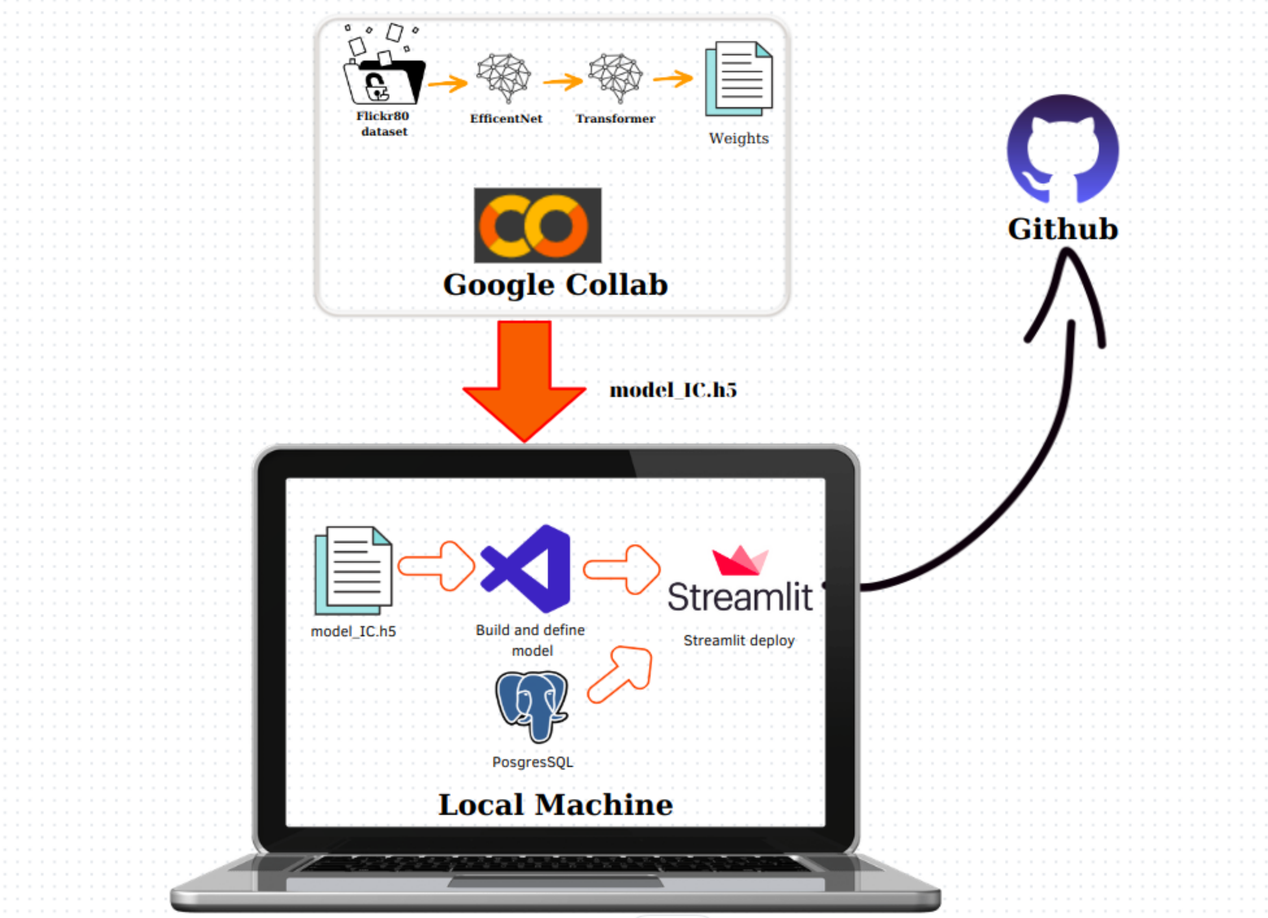
PostgreSQL is a relational database management system, which means that it stores data in tables and uses SQL (Structured Query Language) to manipulate and query the data. Each table in a PostgreSQL database is made up of rows and columns, and each column has a specific data type, such as integer, text, or date.

PostgreSQL supports many advanced features, including:

- ACID (Atomicity, Consistency, Isolation, Durability) transactions, which ensure that database operations are atomic, consistent, isolated, and durable.
- Concurrency control, which allows multiple users to access the database simultaneously without conflicts.
- Full-text search, which allows for powerful and efficient text-based searches of the database.
- Geospatial data support, which allows for the storage and manipulation of geospatial data such as points, lines, and polygons.

PostgreSQL is a highly extensible database management system, it has a large and active community of developers that have created a wide variety of extensions and add-ons for it, such as for full-text search, spatial data, and data visualization. PostgreSQL can run on various operating systems, including Windows, macOS, and Linux. It is widely used by organizations and businesses of all sizes, from small startups to large enterprises. It is also used in many applications such as data warehousing, e-commerce, data analytics, and more.

6. Workflow.



In my workflow, there were two main stages that I followed in order to build my Image Captioning solution.

The first stage was to train the model and obtain the weights. I used Google Collaboratory as my platform for this stage, as it provides a convenient and easy-to-use environment for training machine learning models. I defined the model architecture, prepared the data and trained the model to obtain the weights.

The second stage was to deploy the model to a web application using Streamlit. I defined the model and fit the downloaded weights from Google Collaboratory to the model. Then I used Streamlit to demo the performance of the model and integrated it with a PostgreSQL database on my local machine to meet the requirement of storage. This stage was focused on making the model accessible to users through a web interface, allowing them to interact with the model and see the results in real-time.

The github of my project is here: https://github.com/TomatoFT/Image-Captioning . You can access it to know how to set up my project and watch the demo in file README.md

7. Drawbacks and Development Directions.

Drawbacks

There are some drawbacks in my project.

- The model does not work too well because I just use the Flickr80 dataset (Which is just a fine dataset with only 80000 images and labels). Because the deadline for the project is just 3 days, I can't train it on the larger dataset (COCO is an example). The reason for it is the time training is too long and the GPU of Google Collab gives me is limited.

- I can't find the way to write an API for my model and choose to download the weights file and put it in the backend of the application. I try to find some ways to use FastAPI but it takes me a lot of time so I have to change the approach to complete the project before the deadline.

Development Directions

I have some development directions from this project

- The model may work better when it is trained with the larger data (COCO is an example) and trained with a machine with GPU.
- Add the login/logout feature to personalize the image upload data.
- We can add the "Write your caption" to collect more data.

8. Conclusion

In this project, I have completed the take-home challenge for the technical part. In this project built an Image Captioning solution by following a two-stage workflow. The first stage involved training the model and obtaining the weights using Google Collaboratory as the platform. The second stage was to deploy the model to a web application using Streamlit, integrating it with a PostgreSQL database for data storage. Then I push my project to Github. The github of my project is here: https://github.com/TomatoFT/Image-Captioning . You can access it to know how to set up my project and watch the demo in file README.md

I am looking forward to receiving your reply and feedback to my project as soon as possible. I Hope you have a nice day.

9. Material

Here is some of the material I use

EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks,
https://arxiv.org/abs/1905.11946
Attention Is All You Need, https://arxiv.org/abs/1706.03762
Image Captioning:
https://github.com/sgrvinod/a-PyTorch-Tutorial-to-Image-Captioning
https://www.youtube.com/watch?v=aaP7JJZuvGs
https://keras.io/examples/vision/image_captioning/
PostgreSQL
https://www.youtube.com/watch?v=-LwI4HMR_Eg&t=1s
Connect Streamlit to PostgreSQL
https://docs.streamlit.io/knowledge-base/tutorials/databases/postgresql

_______________END_______________