Dillon McCarthy
DS 552
GitHub: https://github.com/dmccarthy11/CaptionGen/
Streamlit: https://captiongen.streamlit.app/

# CaptionGen

## Introduction

CaptionGen allows users to generate social media captions for a specified input image. The model analyzes the image and then generates creative text to accompany the image on a post, streamlining the social media process. This fills a gap in existing content generation and offers small businesses and personal brands a faster process to generate social media posts, provide scalability for frequent posts, and consistency on brand.

## Model Architecture

This application uses pre-trained transformer architectures to convert images into text. BLIP (Bootstrapping Language-Image Pre-training) is responsible for initial conversion between images and captions by leveraging its vision-language training. These captions are then fed into GPT which refines and expands the text, injecting more detail, creativity, and relatable contexts into the captions. This integration allows for high-quality, contextually rich image captions suitable for social media platforms.

There are four distinct transformations in the model pipeline: BLIP encodes an image and converts it into pixel values, which are then processed to generate token IDs. These tokens are then fed through GPT and transformed into new token IDs for the enhanced caption. Finally, the tokenizer decodes the tokens back to text. Together, the first and last transformation layers convert data to numerical embeddings, making it usable by the inner two layers for the deep-learning framework.

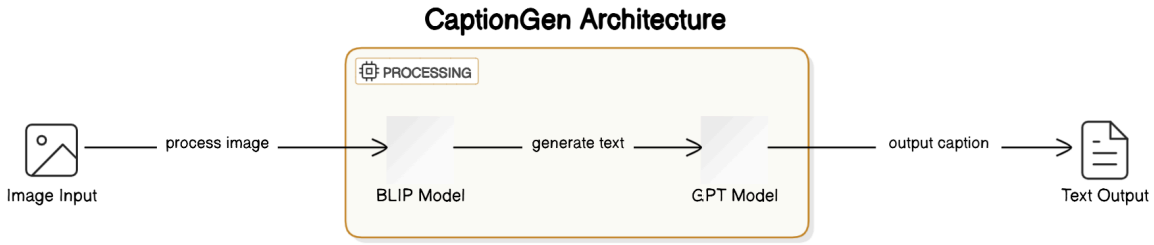



*Figure 1: Diagram of the application architecture*

BLIP and GPT are well established models using attention mechanisms to extract key components in the data. BLIP uses a vision transformer to capture images and impose cross attention layers shared with self-attention mechanisms from a text encoder and decoder (Li, J., Li, D., Xiong, C., & Hoi, S). The decoder is followed by the first decoder-only transformer block of the GPT model. This block uses masked self-attention with a feed-forward network and propagates the text embeddings through several subsequent decoder blocks before outputting

the token logits. Depending on the sampling parameters, a token is chosen and repeated until an EOS token or the maximum number of tokens is reached.

**Training Process**

The training process was the most challenging part of the application. While training can oftentimes be challenging due to compute resources and long convergence times, the limiting factor for training this architecture was the data. Fine-tuning was expected to be challenging, but this was aggravated by the lack of high-quality, accessible, and manageable datasets to use.

To begin the training process, datasets first needed to be identified. With a multimodal transformer architecture, three training opportunities were identified:

I. Image to Long-Text: fine-tune image captioning to directly transform image weights into social media-worthy captions
II. Short-Text to Long-Text: fine-tune a short caption into a longer, more creative caption catered for social media
III. Image to Short-Text: fine-tune image captioning to more descriptively identify objects within images

I. Image To Long-Text:

For ideal application performance, the first approach would be the most efficient and limit the loss of information with a direct image to text transformation. By fine-tuning on real social media images and captions, the model could have the best performance. Nonetheless, this would require a social media dataset containing both images and captions, and while many Twitter datasets exist, these are generally just text based. Instead, two vision datasets were identified: Instagram Influencer Dataset (Kim, Seungbae et. al.) and “takara-ai/image_captions” from Hugging Face. The Instagram Influencer dataset is a private dataset with access granted from the researchers. Files are saved to a Google Drive containing ~200GB of image-caption pairs spanning 64 multipart zip files. After several attempts to work with the dataset, including both Linux, Windows, and Python decompression tools, the multipart zips were unable to be loaded for training. It was determined that these files were not complete or required to be decompressed using the exact tool it was compressed with, and that information was not shared by the researchers.

Similarly, for the Hugging Face dataset, the size proved challenging to work with. The dataset was around ~300GB of image-caption pairs and were general across all domains, not just social media. Despite being stored on Hugging Face, the size exceeded Google Colab’s disk space for the free tier, and other techniques for prototyping with the dataset proved more challenging than its value. Altogether, combined attempts to work with these datasets intermittently absorbed almost two weeks of development.

II. Short-Text to Long-Text

The second approach was training a text-to-text model on enhancing a simple caption into a longer, more creative caption for social media. These short captions would be similar to what

BLIP would output, and GPT-2 could be fine-tuned to output the enhanced captions; however, this is not a common use case in research. Nonetheless, after heavy research, the Waterfront/social-media-captions was identified as an ideal match for the task. This dataset contains 45,400 pairings of "human" and "assistant" captions, where the assistant expands upon the human caption to make it suitable for social media. In fact, these human captions were near replicas of a pre-trained BLIP output. Unfortunately, despite various hyperparameters and fine-tuning techniques, the models did not evaluate well after training. Captions generated were off topic and hallucinated objects or people, and included an excessive amount of hashtags. The challenge here lay with the quality of the data itself, and without any images in the data, it seemed a lot of details were lost between the human and assistant captions. For instance, one such example was:

> "### Human: Write a social media photo caption based on this photo description: a woman in a red jacket and a dog ### Assistant: Rocketgirl and Rocketdog after the @houstonrockets dog walk this morning! #rockets"

The short caption does not mention anything about the Houston Rockets, yet the enhanced caption centers around them. As a result, hand-testing the model on this dataset gave only around 30% relevancy to the images, frequently generating captions like:

> "### Human: Write a social media photo caption based on this photo description: man finishing race in the street ### Assistant: Lévis parc du marque 🌊🤟✨🌭 #runnerspace #street #marque #nofilter #tourist #sunny #marquepage #racing #marquettes #classic #classiclifestyle #marquette #marquettesofinstagram #lollabarcelona #instagood #instagoodofthestre #instagood #instamood #loveyou #bae #men #tired #like #follow #tbt #tbtonly #myonlyhype"

This hand-picked example was far from an outlier in the outputs, and despite appearing as overfitting, showed up even after very minimal fine-tuning. As a result, it was determined that without the images in the original data collection too much information is lost between the training input IDs (input caption) and labels (output caption).

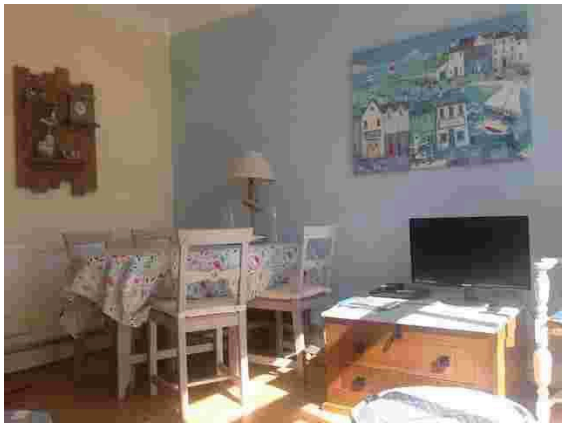
Instead, a unique approach was taken by using a synthetic dataset. In order to train GPT-2 for mapping short captions to long captions, GPT-4o was used to generate a synthetic dataset saved as a CSV file. GPT-4o is openly queryable, and it was asked to generate BLIP-like captions given an example and give an expanded, more creative caption. The model was only capable of producing around 1000 entries, but this enhanced fine-tuning nonetheless by transferring generalized text captions to comparable creative captions without losing image data because there was no image in the first place.

Training was completed using Google Colab to manually test the hyperparameters. Human evaluation showed the captions most relevant when using a learning rate of 5e-6, batch size of four, weight decay for regularization, and training over three epochs. Any more training and the model overfit the synthetic data and tended towards repeating captions.

## III. Image to Short-Text

Finally, with the capability to generate a creative caption provided a short one, the image to short-text process needed to be fine-tuned. Given the second half of the architecture captured

the social media aspect for the application, this approach is made easier by removing the need to cater towards social media datasets. Instead, the goal for training this portion was to capture as much detail as possible of objects within the image. BLIP is very effective at labeling images, but captioning with adjectives not so much. Datasets for this are plentiful, but to capture the most details Obscure-Entropy/ImageCaptioning_SmallParquets was identified as a strong dataset to enhance BLIP's captioning details. The dataset contains 1.5 million small image-caption pairs at around 34GB, and many rows contain several adjectives and descriptive identifiers of objects in the image, something the pretrained BLIP model does not always capture. One example from the dataset is as follows:



```
The image captures a cozy living room bathed in
natural light. Dominating the scene is a wooden
coffee table, adorned with a vibrant tablecloth
featuring a floral pattern. The table is surrounded
by four white chairs, each boasting a unique design
on their backs. Above the table hangs a large
painting depicting a serene coastal scene with
houses and boats, adding a touch of artistic flair
to the room. To the right of the table, a wooden
cabinet stands tall, housing a television set on its
top. The room's walls are painted in a soothing
shade of blue, complemented by white trimmings. A
wooden clock hangs on one wall, while a wooden shelf
on another wall holds various decorative items. The
room exudes warmth and comfort, inviting one to sit
down and enjoy the view.
```

The caption is extremely descriptive and emphasizes key features of the image that traditional BLIP does not. Fine-tuning with these image-caption pairs allows for stronger and more creative captions to be generated while the fine-tuned BLIP learns to associate the image embeddings with more descriptive and creative outputs. Furthermore, the images in the dataset are already small in size, but also encompass a wide range of sizes with 82.7% being less than 1k pixels in width and an additional 14.3% less than 2k pixels in width. This variety will mimic user input from the application, yet the smaller size will maintain an efficient training process.

After loading the parquets and shards, the dataset still was too large for the free disk on Google Colab, so the dataset was streamed for fine-tuning the learning rate. After manual evaluation, a learning rate of 2e-6 was used for longer training on Turing with a linearly decaying scheduler to stabilize fine-tuning.

**UI Implementation**

For interfacing with the application the front-end was developed using Streamlit. This open-sourced framework seamlessly integrated with the models and was a quick way to build and share the application. The framework is specialized for generative AI applications and is quickly becoming an industry standard for quick development and sharing.

The user-interface needed to be simple and easy to use for this application, so the app first centered around input from the user. When first launching the app with "streamlit run app.py", the models will need to be downloaded. These are done immediately before displaying the main page of the app, but this only needs to be done when it is first deployed. Once loaded, an image upload was added in the center of the UI, and the image was then passed through the platform architecture to generate a caption. Prior to displaying the caption, the text is passed through utilities to clean the caption and make it presentable and readable, removing any whitespace and headers included in the direct model output. This includes phrases like "the image is" or extra whitespace. Optionally, hashtags and emojis can also be eliminated from the displayed caption using Regex pattern matching.

To keep the UI simple, optional parameters such as inclusion of hashtags and emojis were placed in the sidebar on the left of the screen and enabled by default. These were joined by other model inputs as well, including the GPT model to use, the social media platform to target, and the style to use, either engaging, funny, or professional. Model parameters temperature, top-p, and max output tokens were also included in a dropdown for simplicity and can be adjusted as needed by the user, but the default settings are recommended for best and consistent performance.

**Deployment Steps**

Streamlit offers its Community Cloud platform for hosting and sharing Streamlit applications. Once CaptionGen was fully functioning locally, the deployment process was initiated to share the app. Several steps were identified to deploy and publish the code:

1. Remove API key from code
2. Create BLIP and GPT models with their access token in Hugging Face
3. Upload BLIP and GPT tensor weights to Hugging Face using access token
4. Adjust code to use Hugging Face model instead of loading locally
5. Test loading model from Hugging Face with Streamlit
6. Upload remaining code to GitHub (Can't store large files like the models)
7. Verify requirements.txt for Streamlit to install dependencies
8. Deploy new Streamlit App with GitHub repo and OpenAI secret
9. Monitor Streamlit Community Cloud App install all dependencies and build the app
10. Share the URL and let others start captioning their social media posts!

Before deploying, the code needed some modifications before it was ready for production. Local testing was done using an API key, so this key was removed from the code and placed in a Streamlit secret. The pre-trained models needed to be loaded elsewhere as well. The fine-tuned BLIP and GPT exceeded the file size limit of the basic GitHub storage tier, so rather than load locally these models had to be stored on external cloud storage and downloaded at runtime into the Streamlit app. Since the repository already used the transformer library for loading the models, Hugging Face was the ideal candidate for model storage. The models were created under the user `dmccarthy1145` and then uploaded using Hugging Face CLI. From there, the application was modified to load the models from that user with the transformer library which seamlessly integrated into the code and was validated locally.

With the models offloaded, the rest of the repository for the Streamlit app could be uploaded to GitHub, as well as some supporting training files and development work. Streamlit Community Cloud looks for a `requirements.txt` file when deploying a repository, so all dependencies were gathered and verified using Python 3.9 and a virtual environment. Finally, an App was created and linked to the GitHub repository and API key. After several minutes building and installing dependencies, the app was finally published under https://captiongen.streamlit.app/.

**Conclusion**

Altogether CaptionGen offers a streamlined method to rapidly generate creative captions for images with consistent styles. GPT-4o integrates extremely well with the fine-tuned BLIP model and is highly recommended for use. The application creatively engages readers through interesting captions, enhancing the visual storytelling experience and making content more compelling and relatable. Whether for social media, marketing, or personal projects, CaptionGen ensures that each image is accompanied by a captivating and contextually relevant caption, elevating the overall impact of the visual content. Go see for yourself!

**References**


Alammar, Jay. The Illustrated GPT-2 (Visualizing Transformer Language Models). https://jalammar.github.io/illustrated-gpt2/. 2025

Kim, Seungbae and Jiang, Jyun-Yu and Nakada, Masaki and Han, Jinyoung and Wang, Wei. Multimodal Post Attentive Profiling for Influencer Marketing. *Proceeding of the Web Conference,* pg 2878-2884, 2020.

Li, J., Li, D., Xiong, C., & Hoi, S. BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation. arXiv preprint arXiv:2201.12086, 2022.

Waterfront/social-media-captions, Johannes. *Hugging Face*. https://huggingface.co/datasets/Waterfront/social-media-captions, 2025.