Summary of Multimodality and Large Multimodal Models (LMMs)

Daniel Hou (houd), Oh Jun Kweon (ohjun), Yuxuan Xia (xyuxuan)

Problem and Motivation

Traditionally, ML research has focused on models using a single mode, usually text. But the potential benefits of supporting multiple modes are significant. Introduced in 2021, CLIP made the major contribution of using a shared embedding space for text and image, resulting in a performant multimodal model.

The term multimodal model can generally refer to 3 types of models: (1) input and output are different modalities (2) inputs are different modalities (2) outputs are different modalities. The term LMM (large multimodal model) comes from the common architecture of connecting a visual encoder to an LLM to generate the output, but having being connected to a language model is not necessary to be considered multimodal. For example, CLIP is a multimodal model that takes image as input to produce text as output, and it is not connected to a language model. On the other hand, Flamingo is an LMM that connects a CLIP-like encoder to an LLM, taking text and image as input to produce text as output. Finally, note that multimodal models can be used for classification and generation; CLIP is a classification model, Flamingo is a generative model.

Related Works

BLIP2 (by Salesforce) creates more efficient vision-text by using fewer parameters than Flamingo. BLIPS2 utilizes a “Q-Former” to extract the most useful visual features of an image input and feed the input into a fully connected layer.

LLaVA (by UWisc, Microsoft) demonstrates the power of fine tuning existing models with a high quality dataset. Its underlying architecture is not new, but they fine-tune the models using a dataset that converts images to “symbolic representation” - this representation is a (caption, boxes) pair that uses bounded boxes on the images to provide a high quality caption.

MiniGPT-4 (open-source) also uses a similar underlying architecture, but fine-tunes the underlying models using a dataset that follows specific templates (eg. a chatbot that receives an image followed by a question about that image).

DeepSpeed-VisualChat (by Microsoft) makes an architectural improvement for multimodal inputs with multimodal causal attention. In this mechanism, images attend to themselves, but text contains self-attention as well as cross attention to include images.

Resource Consumption Table:

	Flamingo	BLIP2	LLaVa	MiniGPT-4	DeepSpeed-VisualChat
Training Data Size	N/A	129M images	158K language-image instruction-following data	5M image text pairs	222681 samples
Learnable Parameter Size	10.6B	1.2B	N/A	N/A	10M-100M
Training Resources	TPUv4 instances	16-A100(40G) machine	8 x A100 GPUs	4 x A100 GPUs	DeepSpeed Framework
Training TIme	15 days	9 days	18 hours	10 hours	N/A

Solution Overview

The Flamingo Large Multimodal Model consists of mainly two parts: an image encoder, which uses a CLIP-like architecture; and a Language Model, fine tuned from the existing Chinchilla, with additional layers added to incorporate visual data.

The image encoder is trained in conjunction with a text encoder using the contrastive objective from CLIP. The training is done on 2.1M image-text pairs. After the training, the text encoder part is discarded and the image encoder is frozen in place.

The language model of Flamingo is based on a previous language model from Google, Chinchilla, with its parameters frozen in place. To incorporate image data, Flamingo added two new components: Perceiver Resampler and GATED XATTN-DENSE layers. The Perceiver Resampler takes in a variable amount of feature outputs from the image encoder, and transforms them into a fixed number of visual output tokens, to be fed into the language model. The GATED XATTN-DENSE layers are interleaved between the transformer layers in the Chinchilla model. This layer uses a masked cross-attention followed by a feed-forward layer, both gated to provide better performance. The text input and visual output from the Perceiver Resampler layer are fed into this layer, and its output are fed into the transformer layer.

During inference, the interleaved images and text are first separated, where the position of images in text is marked by special tokens. The image is fed into the image encoder, followed by the Perception Resampler. The output and then fed together with text input into the modified LLM, which finally outputs text.

For training, both the encoder and the LLM are frozen, and only the Perceiver Resampler and GATED XATTN-DENSE layers are trained. Flamingo uses 4 datasets: two image text pair

datasets with 2.1B pairs combined, one 27M video text pair dataset and one dataset with 43M websites with interleaved images and text. The Flamingo took 15 days of compute time to train.

Flamingo comes in different size variants, largest being 80B parameters. Most of the parameters come from the LLM, with the largest being 70B parameters.

Limitations

Flamingo has some limitations, some of which are inherited from its reliance on an LLM. First, scalability is difficult because the number of parameters is bottlenecked by the LLM parameters. Pretraining the LLM with more parameters would require a lot of compute. Furthermore, the design of Flamingo requires 1 cross attention layer per LLM block, meaning that the compute would scale linearly with the size of the LLM. Second, Flamingo suffers from hallucinations, directly inherited from the underlying LLM. Third, Flamingo demonstrates a lack of spatial understanding, requiring hinting to understand complex scenes. Finally, we observe that the state-of-the-art models struggle to count objects in images, though it is unclear whether this is inherently a problem with the architecture of the training dataset quality.

Future Research Directions

Currently, most multimodal systems work mainly with text and images. However, in the future, systems may need to incorporate other modalities. Examples of this include audio, video, and even 3D models. An area of potential work is looking into creating a shared embedding space for all possible modalities, so that LMMs can perform a wider range of tasks and potentially perform better.

Another area of future work is centered around the “completion” nature of Flamingo. Flamingo was trained for completion not dialogue, meaning that Flamingo completes a task based on a prompt but is not able to uphold an interactive conversation with or follow directions from users. Current work in this area involves instruction tuning to allow for LMMs to better process instructions from users.

On the training side, Flamingo had to pretrain its vision encoder, Perceiver Resampler and GATED XATTN-DENSE layers from scratch, which is compute-intensive. An area of research involves looking into how to more efficiently train multimodals by using less training from scratch.

Lastly, one final area of potential work is cerating models that can produce multimodal outputs. One notable use case of this feature is the ability to explain a complex idea with text and diagrams. One option for incorporating multimodal outputs include having models create a textual intermediate output that can then be used to generate other actions. Examples of this idea include outputting HTML that can be compiled into web pages with formatting, links, and images or outputting Latex code that can be compiled as data tables. Another option for incorporating multimodal outputs include producing multimodal tokens as an intermediate output. Salesforce has reported using this technique where each token has a tag/label to denote what modality it is (text, image, etc.) and then each token will be fed into a tag-specific model (text tokens go into language model, image tokens go into image model, etc.).

Summary of Class Discussion

The class discussion centered around 2 main topics: (1) how to represent video and (2) the tradeoffs between embedding each modality independently vs using a shared embedding.

Flamingo represents videos as a sequence of images that are flattened then projected onto the image embedding space. Most frames in a video are similar, so we discussed whether it would be more efficient to sample down videos to a certain fps. The speakers noted that there is existing research investigating the optimal sampling frequency for video. There was another question about whether it would be possible to pipeline the Flamingo process of converting a video into an image embedding. This could be interesting to explore, with potentially large performance benefits. Finally, we noted that representing videos as simply a sequence of images may not be ideal, since we lose audial information. Google's Gemini claims to interleave video, audio, and text as input, but the details on how the architecture supports this has not been made public.

The models we have seen today rely on frozen, pre-trained image encoders and LLM decoders. The second discussion centered around whether it is better to have a separate embedding for each modality (that can be projected onto a shared space) or represent different modalities using a shared embedding from the start (instead of relying on frozen, pre-trained components). Having a separate embedding for each modality allows us to use existing, pre-trained models, which eliminates the cost of training the component from scratch. However, training a model that uses shared embeddings from the start may potentially produce better results. The discussion turned to the tradeoff between the cost of training vs inference. We discussed that, depending on the context (eg. a researcher working on a project vs OpenAI creating a new GPT), different approaches would be more suitable. Based on this discussion, a more formal investigation into the tradeoff between training and inference could be interesting.