Multimodal Approach for News Image Caption Generation

This Project is a requisite for the 'Natural Language Processing' course, CS5803

1st Anirudh Joshi
Dept. Computer Science, IITH
CS23MTEHCH11002

2nd Harikrishnan Vishwanathan
Dept. Computer Science, IITH
CS23MTEHCH11008

3rd Koushik Maji
Dept. Artificial Intelligence, IITH
AI23MTECH11004

***Abstract*—News image caption generation is the task of generating caption, provided an input image and its associated article. The task is more demanding as it involves capturing the relevant context from the article along with the appropriate representation of the image. The paradigm of fine tuning pre-trained models to achieve high performance with little training drives us to explore its potential in this problem. We also explored a zero shot prompt based approach. We used many pretrained models such as ViT, gpt-2 and BERT, along with novel components to incorporate the article context. The following report provides in-depth explanation of all our tried methods to achieve the task of news image caption generation.**

***Index Terms*—ViT, gpt-2, BERT, multimodal**

I. Introduction

News image caption generation is the task of generating caption, provided an input image and its associated article. Generating better quality captions which are more informative and less generic in nature, it requires incorporating article context and image together to generate the caption. The dataset to train such captioning systems must include image of that scene, article context of that scene and the ground truth caption written by domain experts and journalists for that scene. We utilized the publicly available dataset, the GoodNews dataset which contains news articles from New York Times. We tried three different methodologies to generate captions, a zero-shot prompt based model, a vision encoder-text decoder model without multimodal cross-attention and finally a vision encoder-text decoder model with multimodal cross-attention. We discuss the problems faced while implementing these models, and their pros and cons.

II. Dataset Description

The **GoodNews** dataset is a collection of news articles and related images gathered using the New York Times API. It spans from 2010 to 2018, with a focus on the last 8 years due to the prevalent use of images in news articles during this period.

A. Dataset Composition

The dataset includes 466,000 images with captions, headlines, and text articles. These are divided into:

- **Training Set:** Comprises of 424,000 images.
- **Validation Set:** Contains 18,000 images.
- **Testing Set:** Consists of 23,000 images.

B. Unique Characteristics

- **Caption Length:** The average caption length in GoodNews is longer than in generic captioning datasets, indicating that news captions tend to be more descriptive.
- **Ground Truth Captions:** GoodNews provides a single ground truth caption per image, written by expert journalists.
- **Named Entities:** GoodNews captions contain a significant number of named entities, representing 20% of the words in the captions.
- **Part of Speech Tags:** GoodNews has a higher number of verbs, pronouns, and nouns than generic captioning datasets.

III. Preprocessing

The preprocessing of the **GoodNews** dataset involved several steps to ensure the data was suitable for our analysis.

A. Data Acquisition

Initially, the dataset was downloaded from an external source, as direct scraping from the New York Times is no longer permitted. The data was then verified according to our requirements.

B. Data Verification

We manually checked the links between articles and images. Some articles were associated with multiple images, necessitating the creation of a dictionary-like structure to maintain these relationships.

C. Data Structure

The data is structured in a dictionary and a pickle file for ease of access and manipulation. The dictionary, named `embed_dict`, contains the article IDs and their corresponding embeddings. Each key-value pair in the dictionary represents an article and its embedding. The keys are the article IDs and the values are the embeddings. The dictionary is then serialized and stored in a pickle file named `embed_dict.pkl`. Serialization is the process of converting the data structure into a format that can be stored and then reconstructed later in the same or another computer environment.

D. Conversion to JSON

In addition to the pickle file, the dictionary is also converted to a JSON file. Before this conversion, any PyTorch tensors in the dictionary are converted to lists to ensure compatibility with the JSON format. The resulting JSON file is named `my_dict_with_tensors.json`.

E. Tokenization

The text data was tokenized using a pre-trained tokenizer. Each article was prepended with a '[CLS]' token, tokenized, and then converted into indexed tokens. The indexed tokens were limited to the first 512 tokens for each article.

F. Embedding

The indexed tokens were then converted into PyTorch tensors and passed through a pre-trained model to generate embeddings. The embeddings were stored in a dictionary along with the corresponding article IDs for future use.

IV. APPROACHES

A. *Zero-Shot Prompt Based Model*

At first we were trying to implement a zero shot based model since tha dataset was too huge training a model from the scratch would have taken much longer time. hence we opted to compose several pretrained models, and produce some result with zero-shot setting.

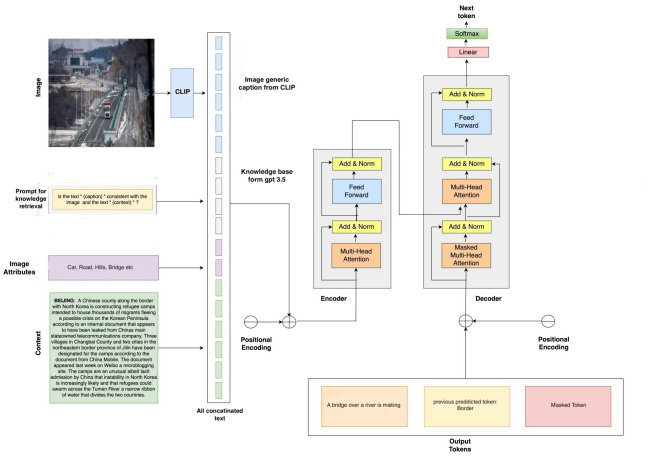

Fig. 1. Zero Shot Prompt Based Model

Our approach harnesses the power of large language models (LLMs) like GPT-3 and BERT for the task of image captioning. The methodology comprises several steps, each leveraging different models and techniques to extract and generate meaningful information.

1) Image Captioning and Attribute Extraction: We employ the CLIP model to generate generic captions for images, and the BLIP model to extract image attributes. These models provide a comprehensive understanding of the visual content, capturing both the overall context and specific details of the images.

2) Named Entity Recognition: We perform Named Entity Recognition (NER) on the news articles to extract all named entities. This step allows us to identify key individuals, organizations, locations, and other entities within the text.

3) Knowledge Extraction: For each named entity identified, we prompt a chat model (GPT-3.5) to provide information about the entity in the context of the article. The prompt is structured as "Give me information about this {named_entity} in the context of this {article} in short". This step generates a knowledge base for each named entity, providing context-specific information.

4) Data Concatenation: We concatenate the generic image captions, image attributes, knowledge base retrieved from GPT-3.5, and the original article. This forms a comprehensive input that encapsulates both visual and textual information, along with context-specific knowledge about the named entities.

5) Caption Generation: The concatenated data is passed to an encoder-decoder architecture comprising BERT as the encoder and GPT-3 as the decoder. The model is instructed to generate a caption for the image. This final step leverages the power of LLMs to generate descriptive and contextually relevant captions for the images.

The use of the chat GPT API for each named entity was a time-consuming process, taking about two minutes per request. Coupled with the daily request limit, this significantly slowed down our methodology and posed scalability challenges. We could have further optimize this process by reducing the number of named entity which will be fed to GPT by some count based method or using co-reference chain.

B. *Vision Encoder-Text Decoder Without Mulitmodal Attention*

Though we have large enough dataset but we focus on finetuning pretrained models on a smaller subset of our dataset, due to lack of compute resources. Hence, we used vision encoder decoder model available in transformers library, to initialize our encoder-decoder architecture.

This architecture gives us a lot of robustness, as in this architecture any pre-trained vision transformer can be used as an encoder, and any pre-trained language model can be used as a decoder.

Encoder inputs are the image features extracted from image using *AutoFeatureExtractor*, and we tokenize the ground truth captions using *AutoTokenizer*. We must pass the earlier initialized encoder and decoder models as an argument to *from_pretrained()* method of AutoFeatureExtractor and AutoTokenizer respectively.

Note that we are not doing multimodal attention in this model, meaning we are not attending to the article context to generate captions in this model.

1) Model Architecture: Our model architecture is shown in Fig. 1. We used pre-trained vision transformer (ViT) for image encoder, and pre-trained gpt-2 as text decoder model. We provide the specific details about our pre-trained models in TABLE I

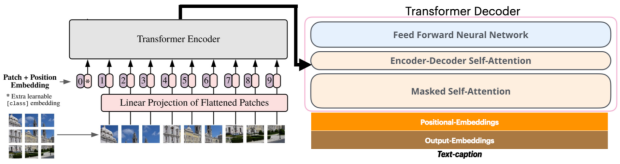

Fig. 2. Vision Encoder Decoder Architecture

TABLE I
PRE-TRAINED MODEL DETAILS

Components	Pretrained Model
vision encoder	"google/vit-base-patch16-224-in21k"
text decoder	"gpt2"

2) Workflow: The features extracted from image are passed in to vision transformer (ViT), which produces an output. The output produced by ViT then goes to encoder-decoder cross attention block in decoder and, simultaneously gpt-2 starts generating caption autoregressively. Initially a special start token is passed to gpt-2. This can be inferred from Fig. 1. We can now use the tokenized ground truth captions and compute cross-entropy loss, which would be backpropagated to finetune the model parameters.

3) Sequence-to-Sequence Trainer: Seq2SeqTrainers are algorithms which are specifically made to train these encoder-decoder kind of architectures, where encoder takes a sequence and decoder generates a sequence, hence the name Seq2Seq. The Seq2Seq trainer helps us in streamlining our training loop for finetuning our Seq2Seq model, provides flexibility for customization, and optimizes the training loop, so that we need not focus on optimizers, learning rate and other training related specific things. We can directly import this trainer from transformers library. Hence we create a Seq2Seq trainer, and train our model.

4) Results: The TABLE II provides the validation loss and rouge scores. We can see that the validation loss is decreasing, so we are able to finetune this model.

TABLE II
WITHOUT MULTIMODAL ATTENTION RESULTS

Ep	Val. Loss	Rouge1	Rougel	Rougelsum
1	0.675264	7.5660	6.8113	6.7714
2	0.672297	7.6028	7.3166	7.3080
3	0.640696	8.7877	7.9329	8.0105

Ep. : Epoch, Val. Loss: validation loss

5) Limitations: The vision encoder-decoder model without multi-modal crossattention produced very generic captions, because it did not incorporate article context. The model introduced in the next section does multi-modal cross attention, and we will show that it performs better.

C. Vision Encoder-Text Decoder With Mulitmodal Attention

The overall architecture remains same, we have a vision encoder and a text decoder. But in addition to encoder-decoder cross attention, we also add attention mechanism between different modalities of data, which are image and article text. We use the *AutoFeatureExtractor* to obtain the image features, and *AutoTokenizer* to tokenize the ground truth captions. Additionally, we use BERT to get article embeddings. The TABLE III provides details of our pretrained models used.
Note that there is an architectural change from our previous model, we created a new custom vision encoder class which emcapsulates the pretrained ViT. We explain more about it in our model architecture and workflow subsection.

TABLE III
PRE-TRAINED MODEL DETAILS

Component	Pretrained Model
article embeddings	"bert-base-uncased"
vision encoder	"google/vit-base-patch16-224-in21k"
text decoder	"gpt2"

1) Motivation: Our main idea and model architecture was inspired from [1]. The architecture proposed in the paper is shown in Fig. 3. It used ResNet as the image encoder, they built their own transformer encoder and decoder and they trained their model using GoodNews dataset after preprocessing the dataset according to their requirements. We use the idea of multimodal attention in our architecture.

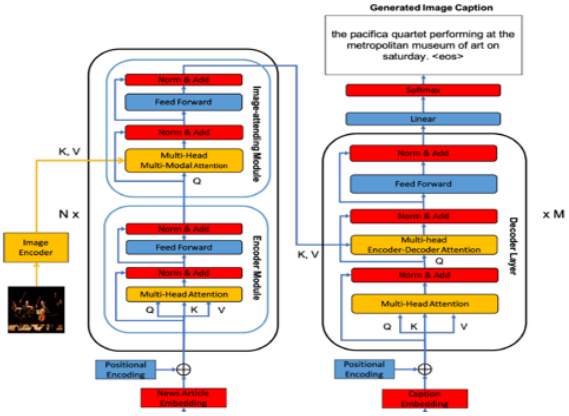

Fig. 3. Vision Encoder Decoder Architecture With MultiModal Attention

2) Model Architecture: The custom encoder block encapsulates pretrained ViT and multimodal attention. The flow of information is depicted in Fig. 4. The custom encoder and gpt-2 decoder are further encapsulated in a vision encoder decoder block which we train. It takes as input the image along with the article and generates a caption.

3) Workflow: We first prepare the article embeddings and image features. The article embeddings are obtained by passing first 512 words of each article to a pretrained BERT model. BERT would give us the contextualized embeddings of dimension 768.
Image features are obtained using *AutoTokenizer*, which is using the ViT feature extractor. We saved these article embeddings and their corresponding image features in a dataset dictionary, we saved this in a pickle file as well. Now our

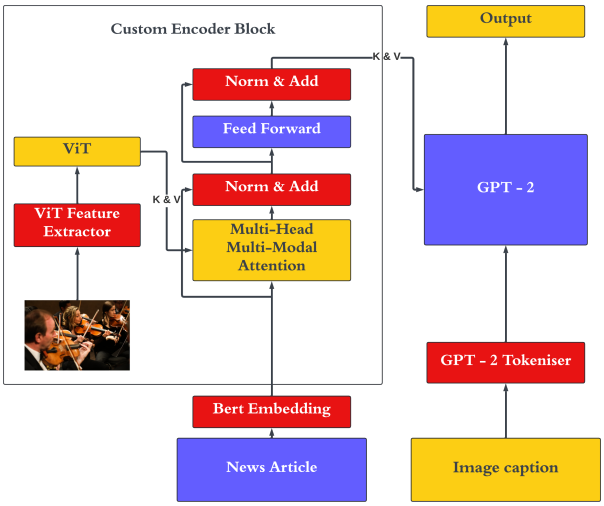

Fig. 4. Vision Encoder Decoder Architecture With MultiModal Attention

dataset contains tokenized images and their corresponding article embeddings, being the values for different keys in the dictionary.

Now we pass the image features to the pretrained ViT model to obtain the image embeddings. Each image embedding is of dimension 768, which is consistent with article embeddings size obtained from BERT. The sequence length is fixed to 197.

Now at this stage both modes of our data, image and article are ready for multi modal attention, so we perform multi head multi modal attention between these embeddings. For the multi modal attention we consider article embeddings as keys and values, and the image embeddings obtained from ViT as queries. Thus the ViT embeddings are enhanced by attending to the article embeddings, which results in the information contained in the article to be integrated.

Finally the output of multihead multimodal attention is passed to encoder-decoder cross attention block in gpt-2 decoder, to generate the caption autoregressively. Cross-entropy loss is computed and, we use this loss to finetune the network.

4) Sequence-to-Sequence Trainer: We discussed the advantages of using Seq2Seq trainer in earlier section. But two of the major challenges we faced while implementing the model, were due to Seq2Seq trainer.

So one of them was to add a custom vision encoder inside the vision encoder decoder wrapper without violating the requirements of Seq2Seq trainer. Further, the use of the VisionEncoderDecoder as a wrapper for the proposed architecture is mainly to serve the Seq2seq trainer which requires a single model to train. The trainer has many other requirements to be able to train on and is mainly built for fine tuning pre trained models as it is. It needs a lot of adjustments in order for a custom model to be used.

The second major challenge was to actually pass the article embeddings along with the image features because the Seq2Seq trainer expects a single sequence as input and a single sequence as output. we confirmed that the Seq2Seq trainer checks the config files of the encoder and allows for only one main input. Therefore, to incorporate the custom encoder, the inputs must be packaged into one before passing it to the model.

Further we need to split the packaged input again in the forward function, to pass only the image features to ViT. The outputs obtained from ViT, then can be used to attend to article embeddings. again split in the forward function to pass the pixel values to the ViT and attend to the Bert embeddings later.

5) Results: The TABLE IV provides the validation loss and rouge scores. We can easily infer from the table that the validation loss is decreasing, hence we were able to finetune the model.

Note that the rouge scores obtained are better than the model without multimodal attention. But still we cannot advocate the correctness of our idea, because we chose smaller training and validation sets due to constraints on compute power.

TABLE IV
WITH MULTIMODAL ATTENTION RESULTS

Ep	Val. Loss	Rouge1	Rougel	Rougelsum
1	0.7765	7.8731	7.2970	7.3423
2	0.7386	9.9183	8.5117	8.6058
3	0.7384	9.8163	8.4867	8.5998

Ep. : Epoch, Val. Loss: validation loss

V. CONCLUSION

So in our work we tried to attempt the challenging task of news image caption generation, which requires integrating different modalities of data to generate quality captions. We tried three different approaches to achieve this task, and our final attempt was to make a custom vision encoder which utilizes pretrained models like BERT, ViT, gpt-2 and multimodal attention. There are still many challenges and we were not able to sufficiently test out our model due to compute heavy nature of our architecture. We were not able to address the problem of named entity insertion.

VI. FUTURE WORKS

We can split the layers of the ViT, to incorporate mullti-modal attention at each layer of ViT. Thus further enriching the embeddings and incorporating the required information from the article.

REFERENCES

[1] Yang, Z., Okazaki, N. (2020, December). Image caption generation for news articles. In Proceedings of the 28th international conference on computational linguistics (pp. 1941-1951).

[2] JBiten, A. F., Gomez, L., Rusinol, M., Karatzas, D. (2019). Good news, everyone! context driven entity-aware captioning for news images. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 12466-12475).

[3] IWang, Y., Xu, N., Tian, H., Lv, B., Duan, Y., Li, X., Liu, A. A. (2023, July). Knowledge Prompt Makes Composed Pre-Trained Models Zero-Shot News Captioner. In 2023 IEEE International Conference on Multimedia and Expo (ICME) (pp. 28779-2884). IEEE.