Image Captioning with Deep Neural Networks

CS 6350.002 | Anurag Nagar

Euphemia Wu
The University of Texas at Dallas
qxw180007@utdallas.edu

Saffat Ahmed
The University of Texas at Dallas
sia180003@utdallas.edu

I. ABSTRACT

The improvement of machine learning and artificial intelligence technology has dramatically affected our lives for the past decades. There have been many approaches utilizing Deep Learning models that hold considerable promise for projects requiring imagining recognition and generating captions. Describing images and providing relevant texts are considered to be challenging tasks in the realm of imaging science. These projects can benefit individuals with visual impairments by providing metadata information on the images used for search engines. The study of this area is still augmenting and improving, and the impact of advancements has the potential to significantly and positively impact various task-specific applications. This paper does not aim to reveal an in-depth study of imaging captioning with deep neural networks. Instead, it will provide necessary information on the basic model used for the project and a detailed analysis of the result. Further code improvement is also suggested at the end of the paper.

II. INTRODUCTION

In the digital era, images have become essential to people's daily lives. Its purpose is to display art, help people with visual disabilities, and communicate and convey information through news outlets, advertising, and social media. Often, it is easy for human eyes to extract the features in a simple image and understand its meaning. However, it could still be challenging for machines to unlock the rich semantic information embedded in images. In the field of machine learning, there has been an interesting research area targeting image captioning, which allows the model to provide descriptive captions when an image is given automatically. Problem Definition: *The problem involves extracting important features from an image and transforming them into meaningful information. After that, combining the visual cues with the given texts to generate relevant and clear captions.*

To solve the problem, it uses the concepts of Natural Language Processing (NLP) and

Computer Vision. As a subfield of artificial intelligence, NLP plays a big part in interpreting, understanding, and generating human languages through modeling. Computer Vision, on the other hand, serves to extract the semantic features of the visual word and provides meaningful information based on the extractions. Both concepts are significant in the process of image captioning.

Our study adopts the application of deep neural networks for the task, specifically convolutional neural networks (CNN) and long short-term memory networks (LSTM). Even though currently there are other popular methods, such as using a Vision Transformer (Vit) and Generative Pre-trained Transformer (GPT), we choose the CNN+LSTM-based approach because it contains more research and tutorials for our specific goal.

III. CONCEPTUAL STUDY AND ALGORITHMS

In our study of convolutional neural networks, we revealed that CNN has exceptional capability in feature extraction and image recognition. This idea can easily capture abstract features from images due to its ability to learn hierarchical structures of visual data. It can sense different layers within the visual pictures and find distinct details in each given layer. Image captioning generators can be efficiently implemented using CNN's ideas.

However, despite its powerful ability to extract image features, CNN is not ideal for processing text data. It often does not understand fully a given text and fail to capture the crucial information in sentences. In order to solve this issue, the researchers have decided to split the processing stage into two parts: visual data and text data. Text data will be processed and analyzed by a type of recurrent neural network (RNN) called Long Short-Term Memory networks. LSTM addresses the issue by efficiently capturing long-range dependencies and modeling sequential data. It plays a big role in generating relevant textual information based on given sentences, but it can also incorporate the visual features that CNN has extracted to develop vocabulary models.

The most basic image captioning generator incorporates the ideas of both CNN and LSTM. The synergy between them has formed a foundation of many image captioning systems. CNN serves as a robust feature extraction tool that can encode visual data into a hierarchical model, which can then be analyzed by LSTM to generate smooth and semantically relevant text captions. This pipeline and structural approach allows the language model to efficiently bridge the linguistic gap between text data and visual information, making the output captions more accurate and reliable.

In our initial implementation, in addition to using CNN+LSTM, we utilized the concept of pySpark MapReduce on the Google Colab platform to further improve the data processing procedure. For example, we chose the dataset of Flickr from Kaggle, which has a storage of more than 1 GB. It includes a caption.txt file and more than 8,000 images. Processing such large data

locally can be difficult, and pySpark helps us speed the process by allowing parallelizing of tasks across multiple nodes in a cluster, which is well-suited for a large image dataset. Our code downloads the Flickr8 dataset directly using the packages, and pre-processing the text and image data using mapReduce.

After the Kaggle packages and files are downloaded, our code proceeds to load the necessary libraries to pre-process the data. For example, we import packages such as Tokernizer and StopWordsRemover to process text data. In addition to that, we use the VGG16 model from the Keras library for processing image data. VGG16 is a pre-trained machine learning model that is commonly used in imaging science for object detection, feature extraction, and image classification. In our implementation, we use it for image extraction specifically as it simplifies the task we have to do. However, our implementation of using pySpark failed to work as it was unable to produce a machine learning model we needed. We later used some tensorflow packages and were able to come up with a useful model.

Image captioning is a sequence-to-sequence task, and we use an encoder-decoder framework for the model to train such a task. The encoder analyzes the extracted image features, and the decoder uses the result to generate relevant captions. Our model's architecture ensures that the encoder consists of a fully connected layer when processing the image features, and the decoder includes an LSTM layer, embedded layer, and dense layer when generating the image captions. The code then trains the model in batch format, compiled with some loss function and optimizer. It uses a specific number of epochs for the training samples.

For the evaluation part, We use the BLEU (Bilingual Evaluation Understudy) score because it performs well when measuring how closely a model-produced text matches the human-generated text. It tests how similar the texts are and gives the approximated percentage values. We also use our model to generate descriptive captions for new images at the end, where the code takes a random picture from the internet and displays the corresponding text. This is a small test for us to see if the model can produce correct captions for images that are not in the dataset.

IV. RESULTS

Here are some sample images and their corresponding texts from the input data:

Figure 1: samples from the input data

Our model has produced result like below picture:

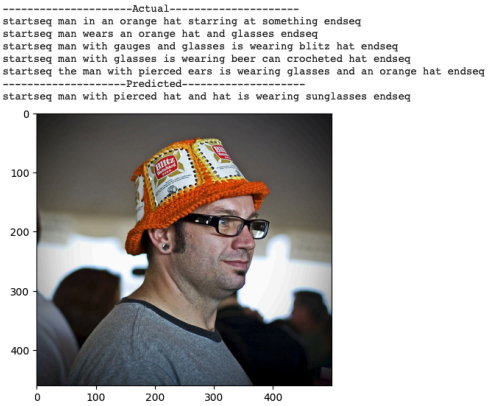

Figure 2: sample result

As you can see, the predicted result generated by the model indeed describes the distinct objects in the picture. It recognizes "man" and "hat". However, it's not entirely accurate as it mistakes a regular pair of glasses as a pair of sunglasses. In addition to that, the result falsely claims that the "hat is wearing sunglasses" while the man is the one wearing the sunglasses.

V. ANALYSIS

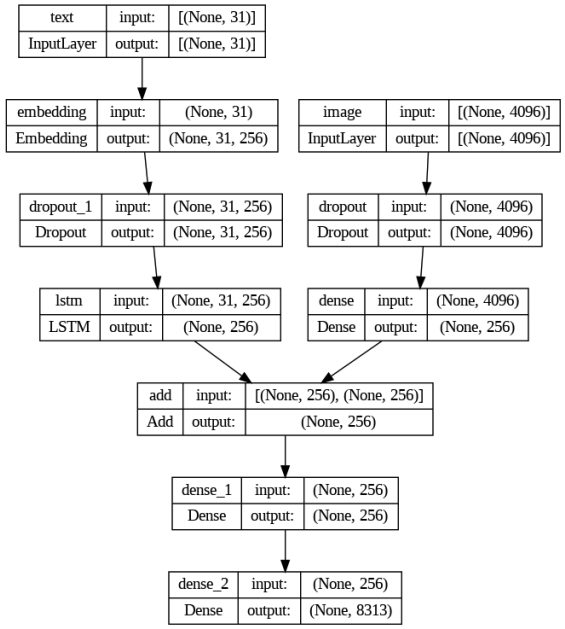

Figure 3: model architecture

Our model defines the architecture of RNN. Inputs1 is the input layer for the image features that specify the input tensor to 4096, which means there is a feature vector of length 4096. Fe1 applies dropout regulation to the input data, and fe2 transforms the image features into another representation. Input2 is used for textual data and specifies the input tensor to max_length, which is the maximum length of the caption sentences. This step is necessary as we must add paddings for captions that don't have max length. Se1 and se2 both serve the purpose of dropout regularization, and se3 is the LSTM layer with 256 units. It analyzes the text data and captures contextual information. Plot_model command is used to show the architecture in a picture format so that it is easier to understand.

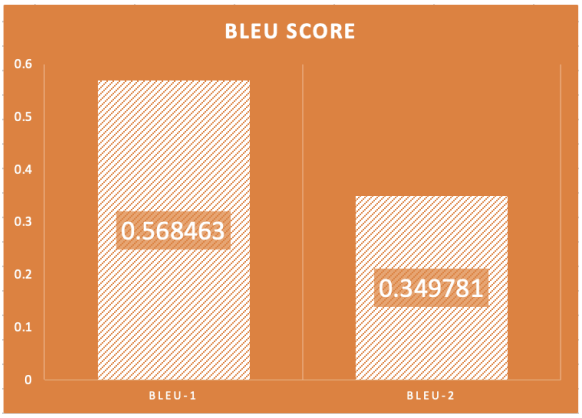

Figure 4: BLEU score comparison

BLEU-1 measures the precision of individual words in the generated captions compared to the original text. A score of 0.57 indicates that 57% of the words match those in the referenced texts. BLEU-2 measures the precision of the pairs of adjacent words. A score of 0.35 means that 35% of the neighboring words match those in the referenced texts. These two scores indicate that our model has a good prediction as it suggests some degree of similarity, but they also reveal that there is room for improvement.

The following are all the models we were able to make with different parameters. Due to time constraints we could only train and evaluate so many models since the dataset we used contains +8k images to possibly train on.

Model Parameters	BLEU Scores
Activation Function: ReLU Epochs: 3 Batch Size: 100	BLEU-1: 0.568463 BLEU-2: 0.349781
Activation Function: Sigmoid Epochs: 3 Batch Size: 100	BLEU-1: 0.365021 BLEU-2: 0.115055
Activation Function: ReLU Epochs: 2 Batch Size: 50	BLEU-1: 0.526416 BLEU-2: 0.320131

VI. CONCLUSION

In our project of building an image captioning generator, we used the concepts of RNN and LSTM, which deal with textual information and image data separately. The combined power of these two models allows us to create a reliable model for the training data. In addition to that, we use pySpark MapReduce to further speed up the process for processing large data sets, as we have used the Flickr8 image and caption data from Kaggle, which is around 1GB of storage. Running such data locally can be challenging. However, the implementation failed as we were unable to create a model using such a method. We had to switch the method to use tensorflow packages. The code still used Google Colab to download and process the data, which has helped us save time. We used helpful libraries such as Tokenizer and StopWordsRemover to process the text data and extract vocabulary for later model training. We used the VGG16 model from Keras to extract crucial image features from the picture dataset.

In our implementation, we first came up with a model that has an architecture of RNN and used the model to process the data. The model was then trained in a batch

size of 100 and epochs of 3, and we received the BLEU score result of 0.56 for BLEU-1 and 0.3 for BLUE-2, which indicates that our model is good when predicting the texts from a given picture. However, there are still some limitations to the model. For example, BLEU scores cannot provide any comprehensive assessment of the quality of the generated captions. BLEU is also not the only metric used to evaluate image captioning generators. In our later study, we found that metrics like CIDer, METEOR, and ROUGE are also popular and provide better insights into the quality of the generated captions. Overall, the model is somehow reliable, but if time allows, we will improve our study and code by implementing other evaluation metrics and developing a better model.

VII. REFERENCE

[1] Amirian, Soheyla, et al. "Automatic Image and Video Caption Generation with Deep Learning: A Concise Review and Algorithmic Overlap | IEEE Journals & Magazine | IEEE Xplore." IEEE Xplore, 2020, ieeexplore.ieee.org/document/9281287.