Image Caption Generator

Md. Shalha Mucha
ID:1808016
Department of Electronics and Telecommunication Engineering
Chittagong University of Engineering and Technology (CUET)
Chittagong 4349, Bangladesh
u1808016@student.cuet.ac.bd

***Abstract*—Deep neural network models are being used to do all of these tasks in the image. Image captioning is the method used to generate a description for just an image. Recognizing the significant objects, their features, and the relationships between both the objects in an image is necessary. It generates phrases that are semantically and syntactically correct. In this article, we present a deep-learning approach that employs computer vision and machine translation to explain images and generate captions. This investigation aims to recognize the various objects that an image contains, analyze their relationships, and generate captions. Flickr8k is the dataset used. The programming language used was Python3, and an ML technique called Transfer Learning will be implemented with the help of the DenseNet model and Gated recurrent units (GRUs) to demonstrate the proposed experiment.**

***Index Terms*—Image, Caption, CNN, DenseNet, GRU, RNN, LSTM, Neural Networks**

I. Introduction

A long-standing challenge in artificial intelligence is teaching a computer machine to identify items and describe them using natural language processing (NLP). Until recently, researchers studying computer vision considered this an impossibility. Models are frequently constructed that will generate captions for an image due to the rapid breakthroughs in deep learning techniques, the availability of enormous datasets, and processing capability. To understand the context of an image and explain it in a natural language like English or the other language, image caption generation includes image processing and natural language processing concepts. While it is simple for people to do something, it requires a powerful algorithm and a lot of computational power for a computer system to do. There have been various attempts to break this problem into smaller, more manageable issues like object identification, image classification, and text synthesis. Using two-dimensional arrays of input images, a computer system mappings the visuals to captions or descriptive sentences. The process of automatically creating captions for photos has received a lot of attention recently. However, while benchmark datasets need quick, precise, and competitive evaluation methods to promote quick advancement, new datasets frequently spark significant innovation. The ability to automatically describe an image's content using properly constructed English phrases may be quite difficult, but it might have a great impact, for instance by assisting those who are visually handicapped in understanding the content of internet images. The well-studied picture classification or visual perception tasks, which are a primary focus within the computer vision community, are considerably easier than this problem. Deep learning techniques have produced cutting-edge outcomes for caption-generating issues. The most amazing aspect of these methods is that they frequently just demand a single end-to-end model, as opposed to a pipeline of specially created models, to predict a caption given a picture. Because it excels at a type of learning that could be extremely beneficial for real-world applications, deep learning has garnered a lot of attention. Unlabeled or unstructured data might be a goldmine for those who are interested in practical applications.

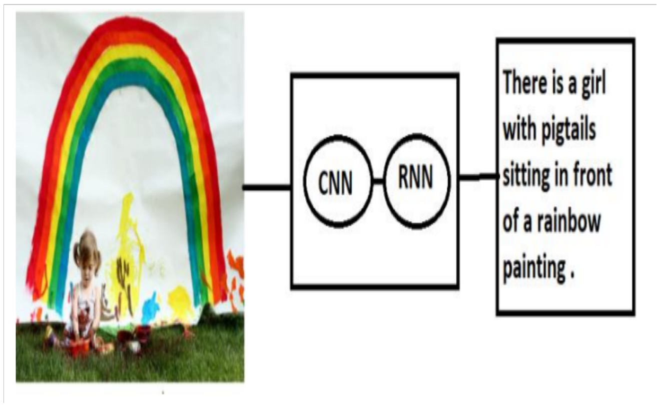

CNN
RNN
There is a girl with pigtails sitting in front of a rainbow painting .

Fig. 1. Our model relies on a deep learning neural network that has two layers: a CNN for vision and an RNN for language generation. As a result, it produces entire sentences.

II. Problem Statement

The main problem in the development of image description started with object detection using static object class libraries in the image and modeled using statistical language models.

Identify applicable funding agency here. If none, delete this.

A. CNN being used: It is a deep learning method that will take in a 2D matrix input image, give different aspects and objects in the image significance (learnable biases and weights), and be intelligent enough to tell one from the other.

B. This model proven effective in identifying the objects in an image, but wasn't able to reveal their relations to all of us (that is just image classification).

C. We introduce a predictive model in this article that combines recent developments in computer vision and machine translation and can efficiently produce meaningful texts.

D. Using an RNN: These networks have loops in them, that enable the persistence of information. A specific type of RNN called GRU consumes less memory and performs faster than LSTM.

III. PROPOSED METHODOLOGY

A. Task:

The goal is to create a system that can accept an image as an input in the shape of a dimensional and produce an output that is a syntactically and semantically accurate sentence that characterizes the image.

B. Corpus:

The corpus that we used was the Flickr 8K dataset. The dataset has 8000 photos, with 5 captions per picture. Understanding all the different circumstances is made easier by the single image's five subtitles. The dataset includes three prepared datasets: Flickr 8k.trainImages.txt (6,000 photos), Flickr 8k.devImages.txt (1,000 images), Flickr 8k.testImages.txt (1,000 images) (1,000 images). The images are chosen from six unique Flickr groups and don't feature any famous people or locations. They are nevertheless manually chosen to display a diversity of scenes.

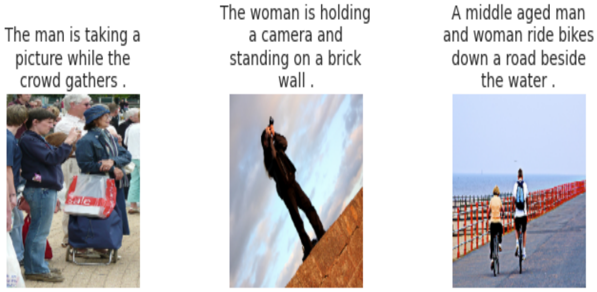

The man is taking a picture while the crowd gathers .
The woman is holding a camera and standing on a brick wall .
A middle aged man and woman ride bikes down a road beside the water .

Fig. 2. Flickr8k Picture Dataset Overview

C. Preprocessing:

The photos and their related captions are processed and prepped independently as part of the data pretreatment process. The input data is entered into the DenseNet201 application of the Keras API, which is built on top of TensorFlow, to do image preprocessing. ImageNet has been used to train DenseNet201. This allowed us to use transfer learning to train the images more quickly. The tokenizer class in Keras is used to clean up the descriptions; this vectorizes the text corpus and stores the cleaned data in a different dictionary. Then, a distinct index value is assigned to each word in the lexicon.

D. Model:

A hierarchy of levels in an artificial neural network made for deep learning is used to carry out the machine learning process. The model is based on deep networks, where the flow of information begins at the original stages, where the model learns something basic, and then the output is passed to layer two of the network, where it is combined with the input to create something slightly more complex, which then passes it on to the third level. Because each level in the network builds on the information it acquired from the prior level, the process continues to yield increasingly complicated results.

Convolutional Neural Networks (CNN):
Deep neural networks with the ability to handle data with an input shape similar to a 2D matrix are known as convolutional neural networks. A simple 2D matrix can be used to represent images.CNN is essential while using photos. It accepts an image as input, gives various elements and objects in the image weights and biases, then separates them out according to relevance. Similar to how the human brain recognizes objects in time and space, the CNN employs filters (also known as kernels) to aid in feature learning and detect abstract notions such as blurring, edge detection, sharpening, etc. The architecture achieves a make the adjustment to the picture dataset thanks to the reusability of weights and the decrease in the amount of parameters involved (2048 to 256).

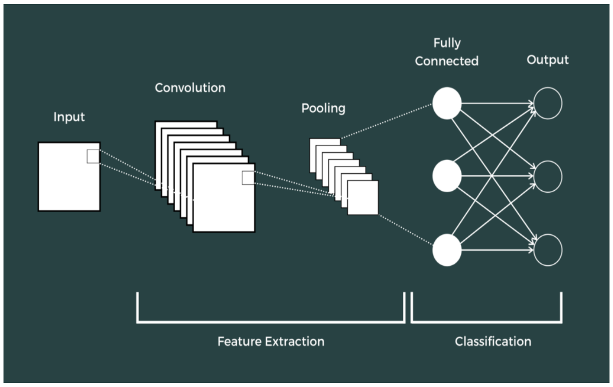


Fig. 3. Architecture of Convolutional Neural Networks for object classification.

Recurrent Neural Networks (RNN):
As compared to lengthy short-term memory, the Gated Recurrent Unit (GRU) is a form of recurrent neural network (RNN) (LSTM). GRU is quicker and requires less memory than LSTM, however LSTM is more efficient when working with datasets that contain longer sequences.Moreover, GRUs handle the issue with vanishing gradients that affects regular recurrent neural networks (values used it to update network weights). Grading may become too little to have an impact on learning if it decreases over period as that back propagates, rendering the neural network untrainable.RNNs can basically

”forget” lengthier sequences if a level in a neural net is unable to learn. The update gate and reset gates are two gates that GRUs utilize to address this issue. These gates can be trained to retain information from further back and determine what information is allowed to make it to the output. As a result, it can transfer important information along an event chain to improve its forecasts.

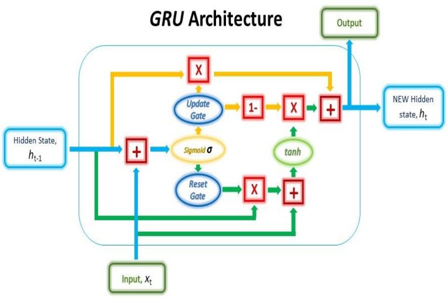


Fig. 4. GRU Architecture.

Architecture

To accept an image as an input and output a caption, we use a CNN + GRU. The final meaningful sentence is produced as a prediction by a ”decoder” RNN using the initial hidden state of a ”encoder” RNN, which first translates the source sentence (which has a different lengths) and converts it into a fixed-length input vectors.

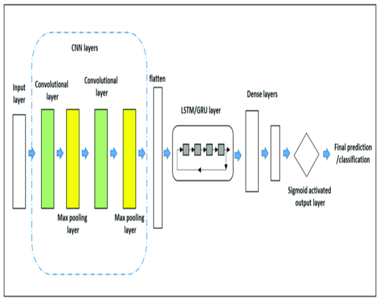


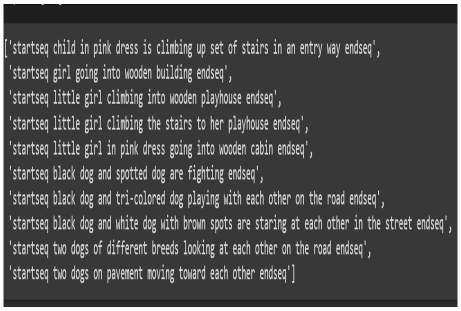
Fig. 5. CNN-GRU Structure.

IV. EVALUATION

Execution of the entire program takes place in 5 major steps. The implementation of the five major modules is as follows

A. Data Cleaning and Preprocessing:

We use Google Colaboratory, a program that offers free GPU processing power, instead of our local PCs, for a comfortable and quick working experience. The text file and the image file are first loaded into separate variables by our software. The primary goal of data cleaning is to eliminate punctuation, lowercase all text, remove stop words, and eliminate terms that contain digits. The descriptions are combined into one language, which will later be used to produce captions for test images. Tokenizing our vocabulary with a special index value is another component of preprocessing the data. We then go ahead and add the ¡start¿ and ¡end¿ identifiers to each caption because they will serve as markers for our GRU to know where a caption begins and ends.

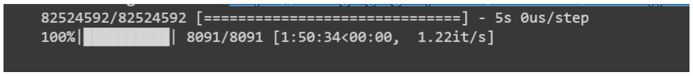


```
['startseq child in pink dress is climbing up set of stairs in an entry way endseq',
 'startseq girl going into wooden building endseq',
 'startseq little girl climbing into wooden playhouse endseq',
 'startseq little girl climbing the stairs to her playhouse endseq',
 'startseq little girl in pink dress going into wooden cabin endseq',
 'startseq black dog and spotted dog are fighting endseq',
 'startseq black dog and tri-colored dog playing with each other on the road endseq',
 'startseq black dog and white dog with brown spots are staring at each other in the street endseq',
 'startseq two dogs of different breeds looking at each other on the road endseq',
 'startseq two dogs on pavement moving toward each other endseq']
```

Fig. 6. Text file after performing data cleaning

B. Extraction of feature vectors:

We’ll be employing transfer learning in our model, which is just a fancy way of saying that we’ll be extracting features from a pre-trained model (in our case, the DenseNet201 model).A convolutional neural network with 201 layers is called DenseNet-201. The ImageNet database contains a pre-trained version of the network that has been trained on more than a million images. The pre-trained network can categorize photos into 1000 different object categories, including several animals, a keyboard, a mouse, and a pencil. The network has learned rich image features for a variety of images as a result



```
82524592/82524592 [==============================] - 5s 0us/step
100%|██████████| 8091/8091 [1:50:34<00:00,  1.22it/s]
```

Fig. 7. Glimpse of extracted features with corresponding image

C. Layering the CNN-GRU model:

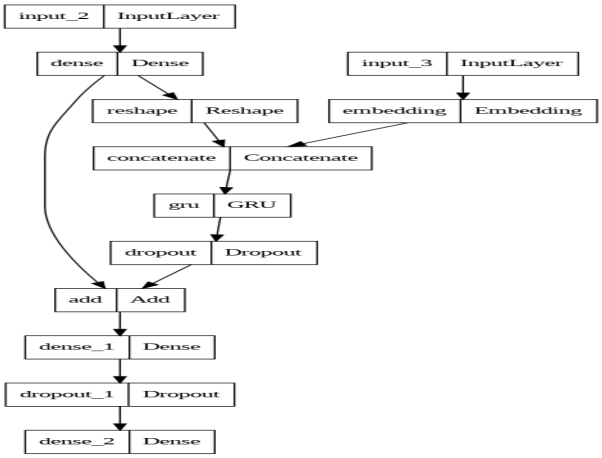
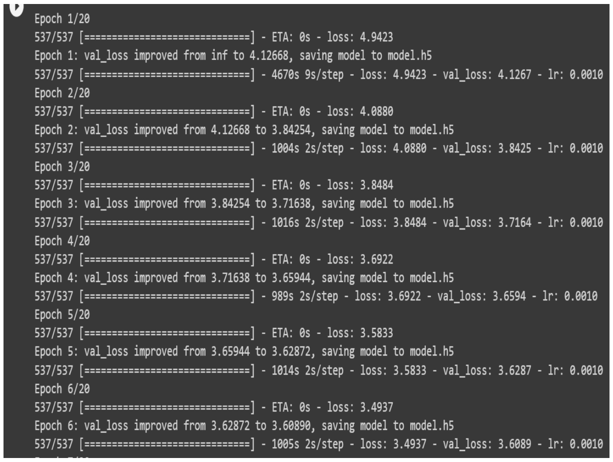


Fig. 8. Structure of the Neural Network

D. *Training the model:*

In the train section, we use the callback method which stop the model for training when it reached the desired loss valu or learning rate. In some cases, it stops the model for trainin when the loss value doesn't change after particular epochs.



```
Epoch 1/20
537/537 [==============================] - ETA: 0s - loss: 4.9423
Epoch 1: val_loss improved from inf to 4.12668, saving model to model.h5
537/537 [==============================] - 4670s 9s/step - loss: 4.9423 - val_loss: 4.1267 - lr: 0.0010
Epoch 2/20
537/537 [==============================] - ETA: 0s - loss: 4.0880
Epoch 2: val_loss improved from 4.12668 to 3.84254, saving model to model.h5
537/537 [==============================] - 1004s 2s/step - loss: 4.0880 - val_loss: 3.8425 - lr: 0.0010
Epoch 3/20
537/537 [==============================] - ETA: 0s - loss: 3.8484
Epoch 3: val_loss improved from 3.84254 to 3.71638, saving model to model.h5
537/537 [==============================] - 1016s 2s/step - loss: 3.8484 - val_loss: 3.7164 - lr: 0.0010
Epoch 4/20
537/537 [==============================] - ETA: 0s - loss: 3.6922
Epoch 4: val_loss improved from 3.71638 to 3.65944, saving model to model.h5
537/537 [==============================] - 989s 2s/step - loss: 3.6922 - val_loss: 3.6594 - lr: 0.0010
Epoch 5/20
537/537 [==============================] - ETA: 0s - loss: 3.5833
Epoch 5: val_loss improved from 3.65944 to 3.62872, saving model to model.h5
537/537 [==============================] - 1014s 2s/step - loss: 3.5833 - val_loss: 3.6287 - lr: 0.0010
Epoch 6/20
537/537 [==============================] - ETA: 0s - loss: 3.4937
Epoch 6: val_loss improved from 3.62872 to 3.60890, saving model to model.h5
537/537 [==============================] - 1005s 2s/step - loss: 3.4937 - val_loss: 3.6089 - lr: 0.0010
```

Fig. 9. Model under Training

E. *Testing the model:*

In the test section, the path of image from the 2000 tes image is passed to the predict caption function. In orde to produce meaningful caption, the model must be calle repeatedly while employing the start sequence, end sequence and real functioning behind image production.

Output:

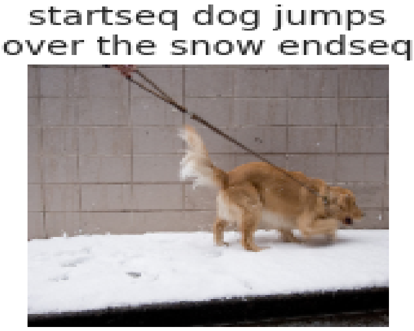


Fig. 10. Caption generated using deep neural network for input Image 1

Output:

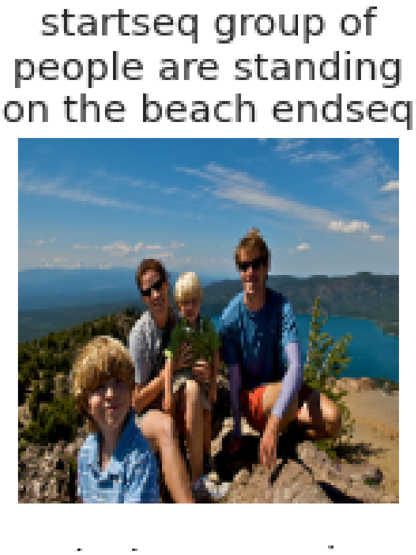


Fig. 11. Caption generated using deep neural network for input Image 2

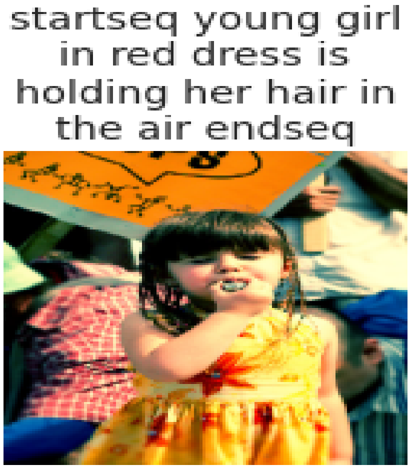


Fig. 12. Caption generated using deep neural network for input Image 3

V. RESULT / ANALYSIS

They employ the CNN+LSTM architecture within the reference notebook (Github link provided in reference). This model exhibits overfitting when the training loss vs. validation loss is visualized.

For this model, we substitute GRU for LSTM because it performs significantly better. With the same database, GRU processes it 29.29 % quicker than LSTM. It effortlessly avoids overfitting and preserves memory.

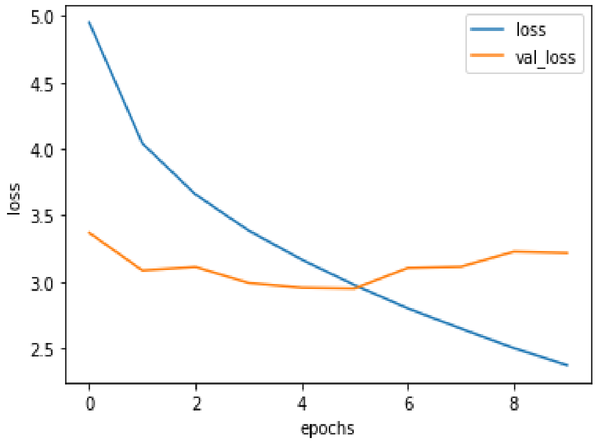


Fig. 13. Training loss vs Validation loss Graph for CNN+LSTM

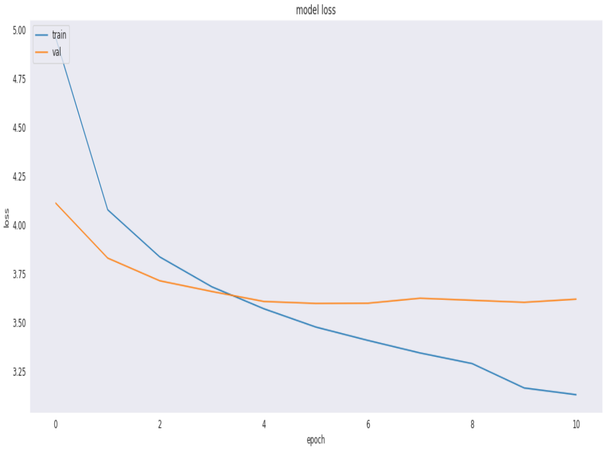


Fig. 14. Training loss vs Validation loss Graph for CNN+GRU.

TABLE I
COMPARISON BETWEEN ORIGINAL AND PREDICTED CAPTION.

Image	The actual caption	Predicted caption
1	black dog and spotted dog are fighting	two dogs play in the grass
2	man wears an orange hat and glasses	man in black shirt blue hat sitting
3	dog with its mouth opened	dog is running through the grass

VI. CONCLUSION

We can see from the data that the deep learning technology employed here produced fruitful outcomes. Together, the CNN and the GRU were able to identify the relationship between objects in photos by synchronizing their efforts. We can use BLEU to evaluate the predicted captions' accuracy against target captions in the Flickr8k test dataset. Text translation uses BLEU ratings to compare translated text to one or more reference translations. This paper showed us several recent advances in the field of AI and machine learning and how broad this field is. While this work attempts to cover the fundamental requirements needed to produce an image caption generator, some subjects within it are open to further study and development.

VII. ACKNOWLEDGMENT

The authors are appreciative of Google's support in outfitting Colab Notebooks with GPU. Thank you, honorable Farhad Hossain sir, for providing me with the right guidance for this project.

REFERENCES

[1] HaoranWang, Yue Zhang, and Xiaosheng Yu, "An Overview of Image Caption Generation Methods", (CIN-2020).

[2] B.Krishnakumar, K.Kousalya, S.Gokul, R.Karthikeyan, and D.Kaviyarasu, "IMAGE CAPTION GENERATOR USING DEEP LEARNING", (International Journal of Advanced Science and Technology- 2020).

[3] Rehab Alahmadi, Chung Hyuk Park, and James Hahn, "Sequence-tosequence image caption generator", (ICMV-2018).

[4] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, et al., "Show, attend and tell: Neural image caption generation with visual attention", Proceedings of the International Conference on Machine Learning (ICML), 2015.

[5] D. Bahdanau, K. Cho, and Y. Bengio. "Neural machine translation by jointly learning to align and translate.arXiv:1409.0473", 2014.

[6] Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi, "Understanding of a convolutional neural network", IEEE - 2017.

[7] Github:Sayandeep-123/Image-Caption-Generator