International Journal of
Image Processing and
Pattern Recognition

ISSN: 2456-6985
Volume 11, Issue 2, 2025
July–December
DOI (Journal): 10.37628/IJIPPR

https://journalspub.com/journal/ijippr

Review IJIPPR

Automated Image Captioning Using Xception–CNN and LSTM for Descriptive Sentence Generation

Matheswaran P.[1,*], Abarna J.[2], Charubala A.[3], Abinaya S.[4], Atchaya N.[5]

Abstract

Generating descriptive texts for images is a complex and challenging task that requires advanced deep learning techniques. This paper presents a descriptive sentence generator for images using a combination of convolutional neural networks and recurrent neural networks. The proposed model utilizes a pretrained convolutional neural network, Xception, to extract image features, which are then processed by a long short-term memory network to generate meaningful textual descriptions. The model is trained on the Flickr8k dataset, which contains 8000 images with five different textual descriptions for each image. The integration of Xception and long short-term memory ensures effective feature extraction and sequential data processing, leading to improved accuracy in caption generation. The results demonstrate that the proposed model can generate grammatically correct and contextually relevant image captions. Further improvements can be achieved by training on larger datasets, such as Flickr16k or Flickr32k, and utilizing advanced hardware for enhanced computational efficiency. This research contributes to the field of deep learning by providing a practical approach to automatic image captioning, which can be beneficial for visually impaired individuals and various academic applications.

Keywords: Convolutional neural network, deep learning, Flickr8k, LSTM, recurrent neural network, sentence generator, Xception

INTRODUCTION

With recent advancements in deep learning technology, what once seemed unattainable has become possible: the detection of people, places, and things in images and conveying it through natural language [1]. Generating a meaningful sentence that describes everything that transpires into an image is an incredibly complex and difficult task, which is why this challenge has been tackled by dividing the problem into subproblems like classification of content, detection of objects, and text generation. This approach allowed developers to constantly improve and build upon the specific problems, and when they all come together, they operate and perform as an efficient engine that can generate sentences that uphold grammar and semantics to help users understand an image better, and in cases of visually challenged people, it helps them "see" the image through words [2]. This application has been an important one for research on deep learning technology and what it can achieve. A computer perceives images as 1s and 0s, where each pixel is given different values of the RGB color model, and these values are reported in binary to help the computer see the image. While this seemed advanced, it did not offer any groundbreaking benefits in the field of image processing. This is

***Author for Correspondence**
Matheswaran P.
E-mail: matheswaranp.cse@krct.ac.in

[1]Assistant Professor, Department of Computer Science and Engineering, K. Ramakrishnan College of Technology, Tiruchirappalli, Tamil Nadu, India
[2-5]Student, Department of Computer Science and Engineering, K. Ramakrishnan College of Technology, Tiruchirappalli, Tamil Nadu, India

Received Date: March 27, 2025
Accepted Date: April 22, 2025
Published Date: December 22, 2025

Citation: Matheswaran P., Abarna J., Charubala A., Abinaya S., Atchaya N. Automated Image Captioning Using Xception–CNN and LSTM for Descriptive Sentence Generation. International Journal of Image Processing and Pattern Recognition. 2025; 11(2): 20–28p.

where artificial intelligence (AI) comes in with its subdomain called deep learning. Deep learning made it possible for computers to understand the context behind the images and recognize what the entities in the images represent [3]. A descriptive sentence generator for images should accept an image and understand all nuanced details in the image to output a sentence that best describes what the image is to a person who cannot see the image or understand it. This is possible because the generator/system is trained on large datasets with efficient algorithms that can do natural language processing (NLP).

Due to the availability of images and huge image datasets online, it was possible to train the deep learning model on one of the datasets, which is the Flickr8k dataset that contained 8000 images with five text descriptions for each image [4]. The machine was able to learn by being trained and tested on this dataset via efficient algorithms and neural networks of both types, convolutional and recurrent. The result (Figure 1) is a model that can give out grammatically correct sentences in the English language once an image is given as input.

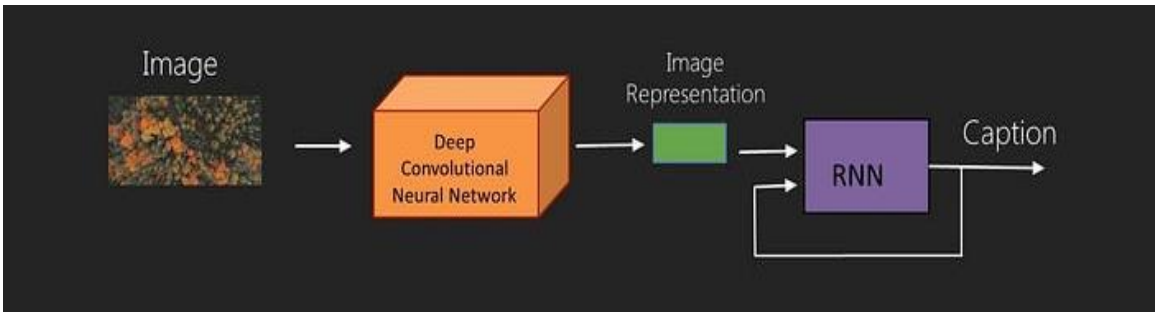

Figure 1. Deep learning model to generate image captions.

The main objective is to aid in the research towards generating captions for images through a practical working system and contributing to the analysis to help blind people perceive images by different means [5].

LITERATURE SURVEY

Several attempts have been made in both theoretical and practical aspects of using deep learning for sentence generation for image data with various machine learning algorithms and datasets. Some of them are as follows:

- In 2021, a similar system to the one developed for this paper with a convolutional neural network-recurrent neural network (CNN–RNN) pipeline where Resnet CNN is used with long short-term memory (LSTM) as RNN [6]. This Resnet–LSTM model generates captions for images passed as input. This acts as an encoder–decoder model, which is a classic way of tackling the problem of generating captions for images. Resnet processes the images by identifying features and that is passed to LSTM to be decoded, resulting in a meaningful sentence describing the input image [7].
- Another model from 2018 was trained on the significantly larger MS–COCO dataset and performed better than CNN–RNN models in most cases, as it was able to use features that CNN possessed throughout the caption generation process instead of only at the encoding stage. This model failed to understand the slight differences that can be identified only when a human examines the images. In this paper, we present a generative model built on a deep neural network architecture that combines recent advances in computer vision and machine translation, especially NLP, which can effectively generate meaningful sentences [8].
- In a paper published in 2020, the entire workflow of the process is clearly laid out, starting from the point of data gathering to generating natural language sentences using the classic method of CNN–RNN pipeline.
- In another paper from 2019, the images were compressed and fed into the encoder CNN and then the RNN–LSTM for caption generation. Here, they avoided the overfitting problem by having modules of the CNN with no initial weights for the model [9].

- A paper from 2021 introduced the concept of using two parameters, called imageability and length.
- In psycholinguistics, "imageable" is defined as something that elicits a mental image, and adjusting this parameter is said to change how detailed or abstract the captions are going to be. The second parameter "length" is said to control how long the caption should be, and using these two together resulted in a customizable caption based on user needs [10], where we obtained the dataset for the development of this system detailed here.

https://github.com/jbrownlee/Datasets/releases/download/Flickr8k/Flickr8k_text.zip (Figure 2).

```
1000268201_693b08cb0e.jpg#0     A child in a pink dress is climbing up a set of stairs in an entry way .
1000268201_693b08cb0e.jpg#1     A girl going into a wooden building .
1000268201_693b08cb0e.jpg#2     A little girl climbing into a wooden playhouse .
1000268201_693b08cb0e.jpg#3     A little girl climbing the stairs to her playhouse .
1000268201_693b08cb0e.jpg#4     A little girl in a pink dress going into a wooden cabin .
1001773457_577c3a7d70.jpg#0     A black dog and a spotted dog are fighting
1001773457_577c3a7d70.jpg#1     A black dog and a tri-colored dog playing with each other on the road .
```

Figure 2. Example captions in the Flickr8k text dataset.

https://github.com/jbrownlee/Datasets/releases/download/Flickr8k/Flickr8k_Dataset.zip (Figure 3).

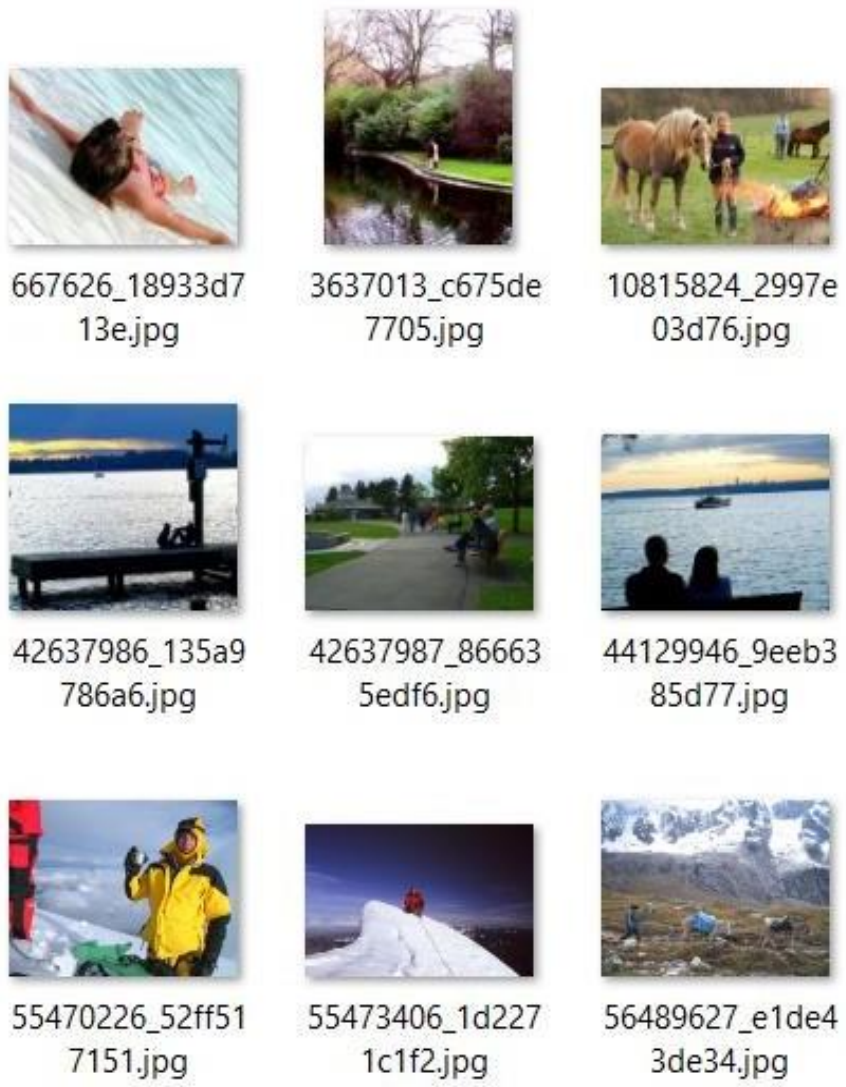

Figure 3. Sample images in the Fickr8k image dataset.

PROPOSED METHODOLOGY

Objective

The primary objective is to generate meaningful textual descriptions that are also grammatically correct for the images passed as input into the system. To achieve that, it is required to have a pipeline of deep neural networks that accept image input, processes it, and outputs a description.

Dataset

The Flickr8k dataset is used to train and test the model to generate descriptions and analyze its accuracy. This dataset contains 8000 images, and each of those images is described by five different captions that detail different elements and context behind the picture that making for an apt source for the system to learn from. These 40,000 captions, coupled with 8000 images with a predefined structure of 6000 images for training and 2000 for testing purposes, do not possess any specific locations or individuals, making this dataset very safe to use without infringing on anyone's privacy. The size of this dataset will be a little over 1 GB (1024 MB) and can be downloaded from the internet for free due to its open-source nature.

Data Preprocessing

Preprocessing is an important step to yield optimized results and to make the deep learning process easier. Here, it is carried out based on the nature of the data. The images are preprocessed separately from the text descriptions. Keras API comes with pretrained CNN like VGG16, VGG19, Resnet50, Inception, and Xception. Out of all of them, Xception comes out at top with a 94.5% accuracy. The ImageNet dataset is used to train the Xception model through which we preprocess the images fed as input.

The tokenizer class from Keras is used to clean the captions or textual descriptions in the dataset. A vocabulary of the words present is created with a unique index value for every word by vectorizing the corpus.

Deep Learning Model

The proposed deep learning model uses artificial neural networks of both kinds, recurrent and convolutional, which contain several hidden layers that process the data between the input and output layers. The working of CNN and RNN differ significantly and thus should be chosen for their specific applications. The basic concept behind neural networks is that there are three kinds of layers called input, hidden, and output. The input layers accept the input and pass it to the hidden layers to start processing the input, and there are several hidden layers that perform some processing on the data received either from the input layer or the previous hidden layer. Eventually, this process reaches the output layer with the respective output for the input given.

CNNs

CNNs are modified variations of a multilayer perceptron that have multiple convolutional layers ready to be connected. These layers process image inputs as a 2D matrix and create a feature map of a specific portion of the input. CNN can learn features of an image and detect objects like people would, making it the best option for processing image data. Here Xception is used, a CNN from the Keras API that is 71 layers deep and was built as an extension of Inception v3, improving upon it by having depth-wise separable convolutions (Figure 4).

RNN

RNNs are more complicated than CNNs, as data travels in both directions through a feedback mechanism or backpropagation. As opposed to the linear manner of data moving in CNN and RNN nodes have memory cell that stores processed information from the respective layer they reside in and make predictions about the other layers and if the predictions turn out to be accurate, the layers give more priority to the weights of the nodes that predicted correctly and if the predictions turns out to be wrong, they learn from it by backtracking and try to achieve correct predictions.

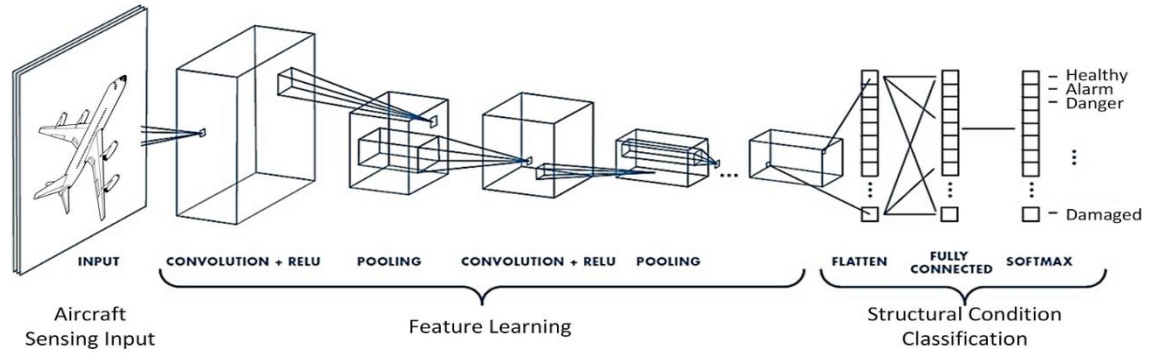

Figure 4. Architecture of CNN.

Because of the loops that are structured to create the recurrent feedback mechanism in RNN, it is the most sought-after choice for processing sequential data, especially NLP. The RNN being used here is a LSTM network commonly referred to as LSTMs. This is a type of RNN that solves the gradient vanishing problem encountered in traditional RNNs. LSTMs provide memory to persist information over numerous steps in the learning process and decide if the persisted information should be discarded or kept in memory. This is realized using gates in the network of three kinds: forget gate, input gate, and output gate. Each LSTM unit has a cell that holds the information over subjective intervals of time, and the three gates control the stream of information as it enters and exits the cell. The forget gate selects the information that is going to be discarded by comparing the current state and previous state and gives a value of either 0 or 1 to the previous state. If the value is 0, the data will be discarded, and if the value is 1, the data will be kept. The input gate and output gate use a similar concept like the forget gates. The input gate determines what information to store in the current state, while the output gate uses the values 1 and 0 to carefully pick the information to be outputted (Figure 5).

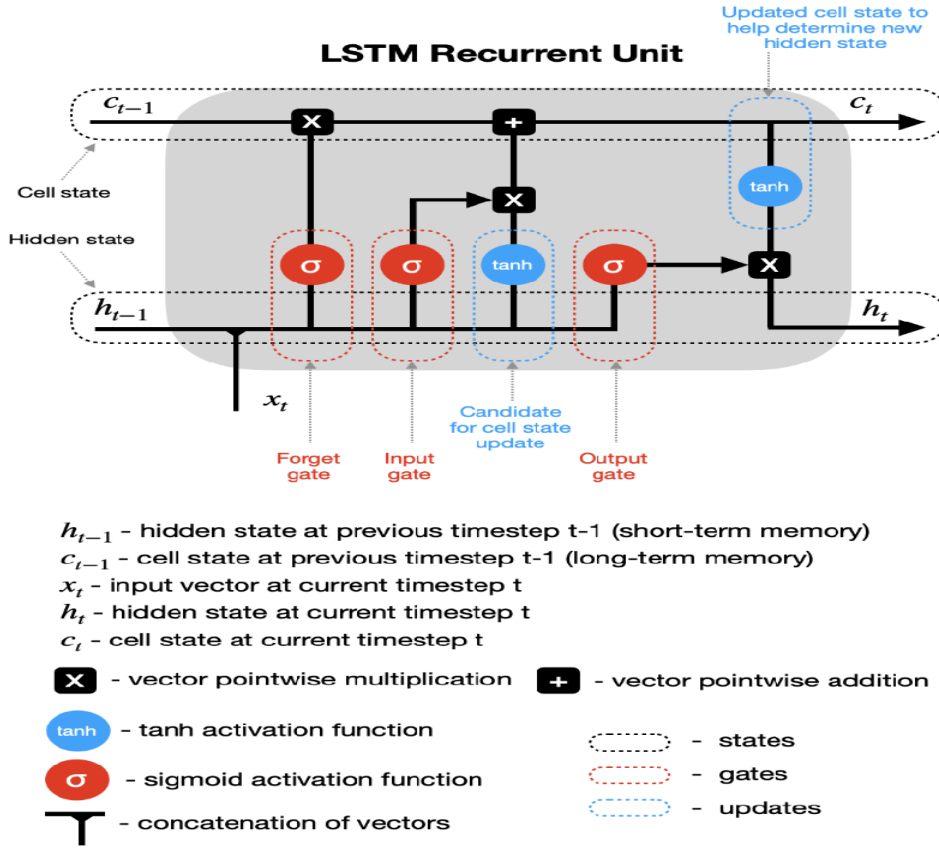

Figure 5. Architecture of the LSTM recurrent unit.

CNN + LSTM Architecture

The combination of CNN (Xception) and LSTM is the paramount aspect of this system. The CNN accepts the image input and performs feature extraction to identify as many features from the input as possible by considering it as a 2D matrix. Then these features are passed to the LSTM, where the entire vocabulary is built from the text dataset and is searched to match the corresponding feature with its word, and finally LSTM performs NLP to create a grammatically correct and meaningful textual description (Figure 6).

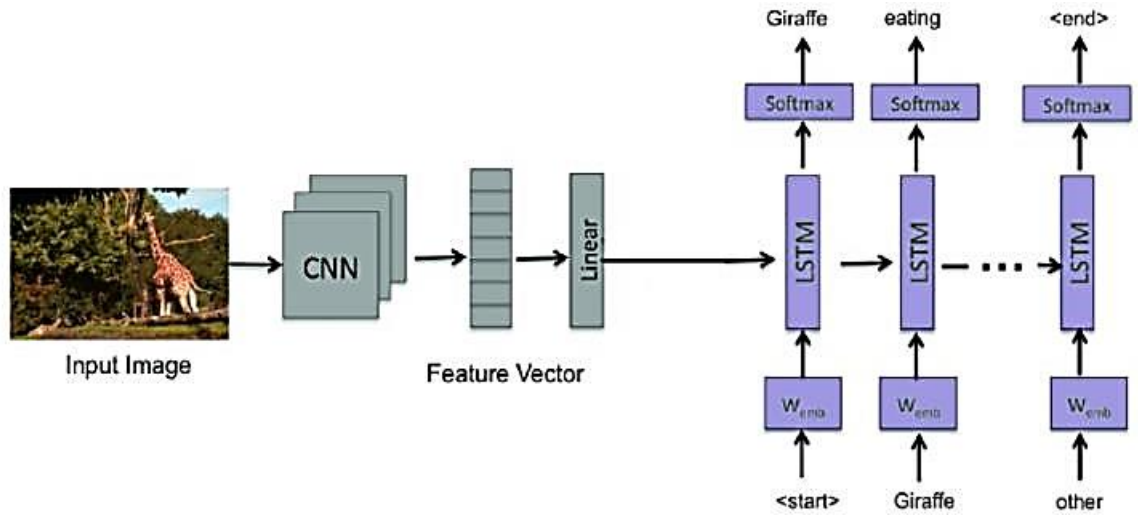

Figure 6. CNN + LSTM architecture.

Implementation

The proposed system is implemented based on the concepts described above by performing the following steps:

Preprocessing and Data Cleaning

- First, we needed Python (Python 3.10 was used during development) to be installed in the local machine, along with pip, which is a package installer and manager for Python that can be installed by checking a box in the Python installation wizard, and an environment variable can also be set by checking a box in the same wizard.
- Once pip and Python are installed, using the command prompt as administrator and typing the following command: "pip install jupyterlab" will install JupyterLab and can then be opened using the command: "jupyter lab".
- Then, after choosing a location for the application, the text and image datasets are loaded as individual variables in Python.
- The text dataset is cleaned first by converting every character into lowercase, removing punctuations and numbers, and hanging "s and a". Every unique word from the text dataset is collected to create an entire vocabulary.
- The cleaned descriptions are saved by mapping them to their corresponding images, as detailed by the dataset, and the vocabulary is indexed, and all the tokens reside in a pickle file.
- The Keras library provides multiple useful classes, like a tokenizer, to make preprocessing easier.
- The Nvidia CUDA (compute unified device architecture) toolkit can be utilized to make use of a graphics processing unit (GPU), if available, to make the entire configuration and deep learning process incredibly faster than when using a central processing unit (CPU).

Feature Extraction

- Feature Extraction is the process of identifying recognizable elements from the picture by looking for patterns, shapes, and colors.

- A feature vector is identified from an object as its most important aspect, like the most intense pixel, in the region of the identified feature.
- The pretrained Xception model is used to carry out this process. Xception is trained on the large ImageNet dataset that contains over a million images. It is a 71-layer deep neural network with higher accuracy over other pretrained CNNs like VGG16 and Inception.
- By using a technique called transfer learning, where a pretrained model is utilized instead of training it us, we import the Xception model from the keras library with avg pooling that uses the global average pooling to get an output as a 2D tensor.
- Xception identifies features from the images and stores them as a pickle file that will be fed to the LSTM, where the words for the caption are generated based on the identified features.

Building the CNN–LSTM Model

This pipeline is the integral working part of the system, which requires connecting the CNN, Xception, with the LSTM network.

- The pickle file containing the features identified by CNN and the tokens from the vocabulary creation is sent to LSTM.
- The pipeline will have a feature extractor that operates as an encoder of the data to be processed, followed by the decoder LSTM.
- CNN feature size is reduced to 256 from 2048 to be able to fit the model, and a dropout layer is created to set random input units to 0 and reduce the chances of overfitting.
- The decoder is defined by adding the dense layer from the CNN and the LSTM sequence with SoftMax activation.
- The model is compiled and saved as an h5 file so that the model can be retrieved from this file after training it, without having to train it every time.

Model Training Process

- The Flickr 8k dataset contains 8000 images with five different captions for each image. The dataset is split into a training set and a testing set.
- In this step, training will be done on 6000 images and loading the h5 model file into a variable.
- The DataGenerator class from the Keras library helps load this large dataset seamlessly into the model in real time by feeding the data in batches at each epoch. The batch size and number of channels can be customized as per need.
- The number of times we wish to run the training iterations is referred to as epochs. After experimenting with different numbers of epochs, we have deemed 20 epochs to be the point at which there are no overfitting or underfitting problems and running the training process for more iterations than 20 will not be necessary unless we go for a bigger dataset like Flickr 30k.
- Every time an epoch runs to its completion, the model generated is saved under an incremented numerical-valued name. The final model file saved will be used to generate a textual description, as it will have learned the most from the dataset.

Testing the Trained Model

- Out of the 8000 images, 6000 were used for the training process, and now the remaining 2000 images are used for testing the model to see how accurate it is.
- The model saved after the final epoch is completed is loaded to be tested with the remaining 2000 images.
- Traditional accuracy scores and measures cannot be used for text-based predictions, which is why we use a metric here called the BLEU score.
- BLEU stands for bilingual evaluation understudy; it is a value between 0 and 1 that compares how close the machine-generated text comes to a high-quality reference text.
- The high-quality reference texts needed to find the BLEU score are present on the text dataset of Flickr 8k, where the image name is mapped to five sentences.
- Each of these 2000 images is passed to the model, where the CNN extracts features and the LSTM generates descriptions for these images, and those generated descriptions are compared with the mapped sentences to estimate the BLEU score.

RESULT/ANALYSIS

Finally, all the Figures 7, 8, and 9 that were subjected to testing and their BLEU scores are analyzed. Here are three randomly selected ones from the training set and their BLEU score, as shown in Table 1.

Figure 7. A man in a yellow kayak paddle through the water.

Figure 8. A man in a red shirt is doing tricks on his bike. BLEU score.

Figure 9. Two dogs are playing together on the grass.

Table 1. Analysis of the captions.

Image Name	Original Caption	Generated Caption
Image 1	A person kayaking in the ocean	A man in a yellow kayak paddle through the water.
Image name	Original caption	Generated caption.
Image 2	A boy wearing a red shirt and jeans is doing a flip on his bike	The man in a red shirt is doing tricks on his bike.
Image 3	A beagle and a golden retriever wrestling in the grass	Two dogs are playing together on the grass.

CONCLUSIONS

After analyzing the test data with additional images from other sources, there were very few images that had a 0 BLEU score. The model was able to work together seamlessly to predict captions for images that were foreign to the dataset. There was a pattern of having a higher BLEU score with images that had common animals, like dogs, cats, horses, etc., and BLEU scores dropped to the lower end with uncommon animals like pangolin and echidna. The overall performance was calculated by calculating the mean of all the individual test images' BLEU scores, which came to be about 0.59 (rounded).

It is remarkable how the model was able to identify not only objects but the relationship between them to understand the activity being performed or the context being depicted by the images. The facts and findings mentioned in this paper are explained concisely and clearly to help support further advancement and research on the topic of generating textual descriptions for images using deep learning. We hope this will serve as a reference for researchers to set up their own deep learning model so that better models can be developed. Adopting a bigger dataset than Flickr 8k, like Flickr 30k or MS COCO dataset, or CIFAR-10 or CIFAR-100, would require a more powerful GPU than what was used here and is speculated to yield better results and a higher BLEU score.

REFERENCES

1. Wang H, Zhang Y, Yu X. An overview of image caption generation methods. Comput Intell Neurosci. 2020;2020(1):3062706.
2. Krishnakumar B, Kousalya K, Gokul S, Karthikeyan R, Kaviyarasu D. Image caption generator using deep learning. Int J Adv Sci Technol. 2020;29(3s):975–80.
3. Hossain MZ, Sohel F, Shiratuddin MF, Laga H. A comprehensive survey of deep learning for image captioning. ACM Comput Surv. 2019;51(6):1–36.
4. Alahmadi R, Park CH, Hahn J. Sequence-to-sequence image caption generator. In: Eleventh International Conference on Machine Vision (ICMV 2018). Bellingham (WA): SPIE; 2019. p. 85–91. (Proc. SPIE; vol. 11041).
5. Panicker MJ, Upadhayay V, Sethi G, Mathur V. Image caption generator. Int J Innov Technol Explor Eng (IJITEE). 2021;10(3):1–5.
6. Sharma G, Kalena P, Malde N, Nair A, Parkar S. Visual image caption generator using deep learning. In: 2nd International Conference on Advances in Science & Technology (ICAST). 2019.
7. Dandwate P, Shahane C, Jagtap V, Karande SC. Comparative study of Transformer and LSTM network with attention mechanism on image captioning. In: International Conference on Information and Communication Technology for Intelligent Systems. Singapore: Springer Nature; 2023. p. 527–39.
8. Redmon J, Divvala S, Girshick R, Farhadi A. You only look once: Unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2016. p. 779–88.
9. Hasan MZ, Ahamed MS, Rakshit A, Hasan KZ. Recognition of jute diseases by leaf image classification using convolutional neural network. In: 2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT). Piscataway (NJ): IEEE; 2019. p. 1–5.
10. Kuo CC. Understanding convolutional neural networks with a mathematical model. J Vis Commun Image Represent. 2016;41:406–13.

