HBRP PUBLICATION

Advancement in Image Processing and Pattern Recognition
Volume 5 Issue 2
DOI: https://doi.org/10.5281/zenodo.6759892

Image Captioning in Real Time

Ankit Patil[1*], Karishma Saudagar[2], Atul Maharnawar[3], Tejas Rangatwan[4], I. Priyadarshini[5]
[1,2,3,4]BE Final Year, [5]Professor
Computer Department,
KK Wagh Institute of Education Engineering and Research, Nashik, India.
[*]Corresponding Author
***E-mail Id:-** ankitpatil3003@gmail.com*

ABSTRACT

The current development in Deep Learning based Machine Translation and Computer Vision have led to incredible Image Captioning models using advanced techniques like Deep Learning. Even if these models are very accurate, they often rely on the use of exorbitant computation hardware making it problematic to apply these models in real-time scenarios, where their actual uses can be noticed. This model uses a hybrid CN-NRNN model, where the CNN part of the model system uses the Xception model for transfer learning, and RNNs are widely used in language modeling. The Flickr8k dataset is used for real-time training and testing. RNN's LSTM model is used to avoid problems with extinction or gradient explosion during the training phase.

Keywords:-*CNN, LSTM (Long Short Term Memory), RNN, Xception*

INTRODUCTION

Project Idea

For a machine to be able to spontaneously describe things in an image along with their connection or the actions being performed using a suitable language model is a difficult task. For ex. it could help visually impaired people to better understand visual inputs, thereby acting as an assistant/a guide.

Its purpose is to mimic the human ability to understand and process large amounts of visual information in a descriptive language, which is a fascinating challenge in the field of AI.

Motivation of Project

Aid to the blind **-** We can create a product for the blind that will guide them on the streets without the support of anyone else. We can do this by first translating an article into text and then the text into a word. Both are now popular Deep Learning applications.

Self-driving cars **-** Automatic driving is one of the biggest challenges and if we do not say the exact location of the incident near the car, it can improve the self-driving system.

Automatic captions are useful and can make Google Image Search similar to Google Search. That way, all images can be converted to captions first and then searched based on the captions.

In Web development, it is a good practice to give a description of any image from the page so that the image is readable or audible rather than just plain visible. This makes web content accessible.

CCTV cameras (closed TV cameras) are everywhere today, but World Viewing allows you to sound an alarm if you can recreate the relevant captions every time something bad happens. This may help reduce some crimes and accidents.

DOI: https://doi.org/10.5281/zenodo.6759892

LITERATURE SURVEY

In Literature review, various references of the existing projects are taken into consideration.

1. The article "Understanding Convolutional Neural Networks" describes one of the deepest emotional networks known as Convolutional Neural Networks (CNNs). There are many changes in CNN. Convolutional layer, non-linear layer, integrated layer, fully integrated layer, etc. CNNs perform well on ma-chine learning problems and one of the most common algorithms.

2. In the paper "The Generation Awareness Business Image", Modern photo captions produce descriptive definitions, such as businesses with words involved in photography. Here Di Lu, Spencer Whitehead had proposed a very new project that produces descriptive caption captions, rendered images as embedded. A simple solution to the problem we propose is to train a CNN-LSTM model to produce image-based captions.

3. In the article on automatic neural image caption generators for the visually impaired, automatically defining image content in well-organized English sen-tences can be a daunting task, but it can be very helpful in helping the visually impaired. Modern mobile phones can take pictures and help people with poor eyesight. Here, the image as input can generate captions high enough for the visually impaired to hear to see the surrounding objects better. Here, Christopher Elamri uses the CNN model to extract the image elements. These elements are then integrated into the RNN or LSTM model to generate an im-age definition with grammatically correct English sentences that describe the environment.

4. In the paper "Neural Image Caption Generator", the basic problem of artificial intelligence that combines computer vision and natural language processing automatically defines image content. In this article, A.L systematically analyzes the process of generating labels for neural networks. Here, the photo is presented as an insert and as a way to get out in the form of English sentences that explain the content of the photo. They analyzed three components of this method: convolutional neural networks (CNNs), recurrent neural net-works (RNNs), and sentence generation. This model analyzes the image and generates keywords and important words in the image.

5. In his work "LONG TERM MEMORY", Sepp Hochreiter describes the Short Term Neural Memory Network algorithm of the Short Term Team (LSTM). LSTMs are both spatially and temporally local. Computational complexity is each step in the process and represents a weighting pattern. Compared to other algorithms, LSTMs run better and learn much faster. It also solves complex, long-term tasks that previous iterative networks could not.

PROBLEM STATEMENT

To capture image and to recognize the context of an image and describe them in a language like English.

Goals and Objectives

1. To recognize objects in the images and then generates caption for that image.
2. To learn the concepts of a CNN and LSTM model and build a working model of Image caption generator by implementing CNN with LSTM.

Assumption and Scope

System accepts images as input extensions: .png, .jpg, .jpeg

METHODOLOGY

1. Import all the required packages data cleaning- The main text file containing all subtitles is Flickr8k.token in the Flickr8k text folder. Each image has 5 captions (0-4) with a number associated with each

DOI: https://doi.org/10.5281/zenodo.6759892

caption.
2. Extract features vector-The system uses a pre-trained model that has already been trained on a large dataset and removes features from the model. Apply them to our business. The system uses an Xception model trained in an im-age database. This model has 1000 different classes to divide. Implement this model directly in your camera settings. Note that the Xception model uses an image size of 299 * 299 * 3 as input. The system removes the last partition layer and captures 2048 vector features.
3. Load the Flicker8k dataset - The Flickr 8k test folder contains a Flickr8k.trainImages.txt file that contains a list of 8000 image names used for training.
4. Tokenizing the vocabulary - The computer does not understand English words. In computers, the system needs to represent them numerically. Therefore, place a map of each vocabulary with a unique index number. The Keras library provides tokens that are used to create tokens from your vocabulary.
5. Defining the CNN-RNN model - To define model design, the system will be using the Keras Model from the Functional API. It will have three main parts:
• Feature Extractor – The feature extracted from the image has a size of 2048, with a thick layer, it will reduce the size to 256 nodes.
• Sequence Processor – The embedding layer will handle the text input, followed by the LSTM layer.
• Decoder – By combining the output from the two upper layers, the system will be processed in a compact layer to make a final prediction. The final layer will contain a number of nodes equal to our vocabulary size.
6. To Train the model - To train the model, the system uses 8000 training images by performing input and output sequences and grouping them into the model using the model.fit generator () method. The model is also saved in the model folder. It may take some time depending on the system capacity.
7. To Test the model - Generator.py creates another caption test file that downloads the model and makes predictions. The prediction includes the maximum length of the index value, so the system uses the same tokenizer file to get the name from the index value.

OUTCOME
Generates a caption for the specified image.

TYPE OF PROJECT
• It is a project-based project in which the neural image caption model integrates the image caption model into the CNN-RNN framework. It will have a positive effect on the real world, for example by helping visually impaired people better understand the content of the images.
• The domain involved is CNN using LSTM, Machine Learning.

FUNCTIONAL REQUIREMENTS
• To get real time image as input.
• To identify the objects.
• To provide caption for that image.

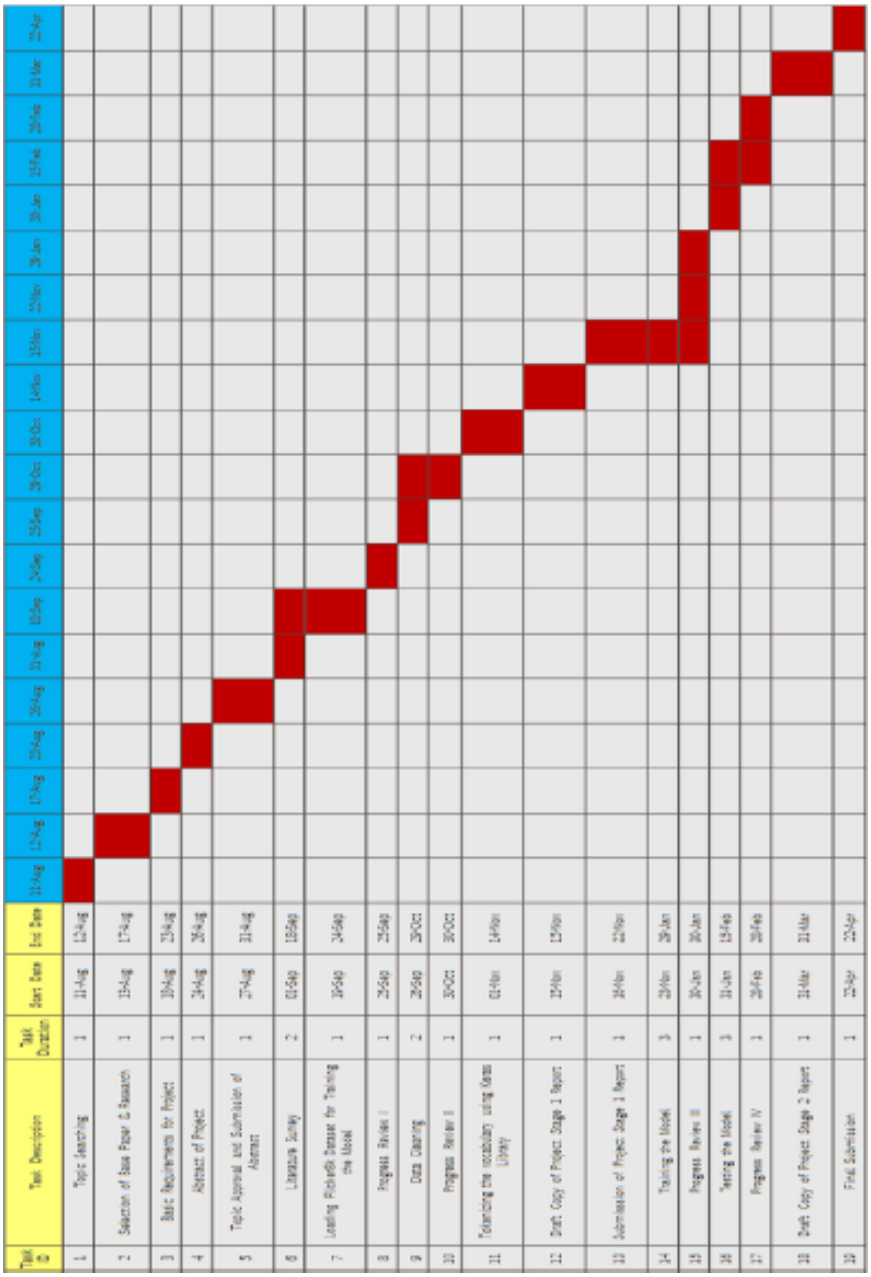
NON-FUNCTIONAL REQUIREMENTS
• Availability:- There are no system requirements, but it works fast with a high GPU.
• Maintainability:- Updates to the system database (currently holding 8000 im-ages) can be added to get accurate captions.
• Accessibility:- The system will be accessible to people who want to get photo captions.
• Speed:- Around 15 mins per model on Nvidia 1660 Ti GPU/ around 1-2 hrs on CPU.
• Usability:- People with eye problems will be able to use the program.

Advancement in Image Processing and Pattern Recognition
Volume 5 Issue 2
DOI: https://doi.org/10.5281/zenodo.6759892

CONSTRAINTS

• Operational Constraints:- Input image format must be .jpg, .jpeg, .png
• Hardware Constraints:- The system meets the minimum requirements.
• Software Constraints:- The target image for the system must have a resolution higher than 500x375.
• Assumptions;- CNN is correctly able to detect object and using LSTM caption id formed. Images subjected are in preferably higher resolution than 500x375.

PROJECT TIMELINE



HARDWARE REQUIREMENTS

CPU with 64-bit support (Recommended High-end GPU)

Advancement in Image Processing and Pattern Recognition
Volume 5 Issue 2
DOI: https://doi.org/10.5281/zenodo.6759892

SOFTWARE REQUIREMENTS

- Operating System: Windows
- IDE: Jupyter Notebook
- Programming Languages: Python3
- Libraries and Packages: Numpy, Keras, Tensorflow

ARCHITECTURAL DESIGN(BLOCK DIAGRAM)

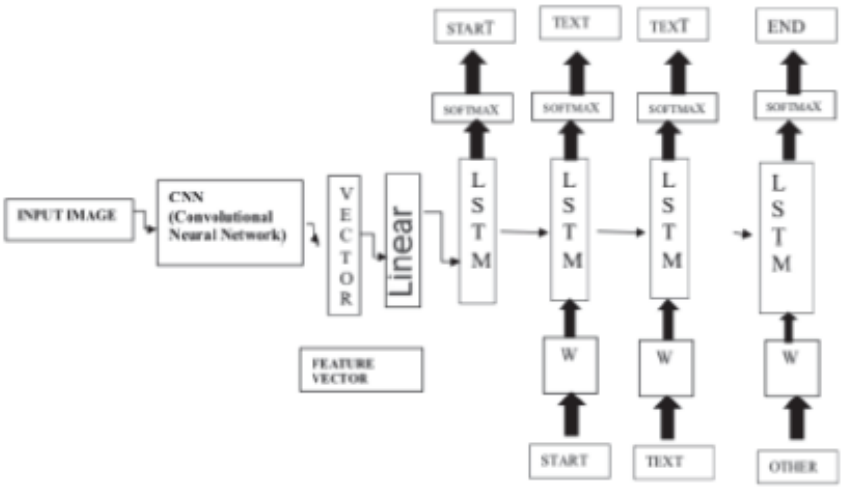


Fig.1:-*Proposed model of Image Caption Generator.*

The proposed caption generator model is shown in Figure 1. This dense vector, also known as embedding, can be used as an integration with other algorithms to produce the appropriate captions for a particular image as output.

Captions With caption products, this .

embedding is a representation of an image and the first LSTM format used to create meaningful image-based subtitles.
System Architecture of our system is shown below in Figure 2.

This is our proposed system architecture will look like:

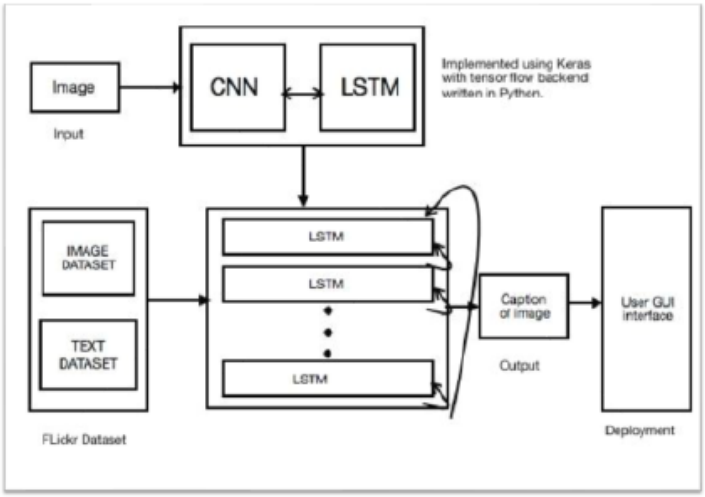


Fig.2:-*System Architecture of Image Caption Generator.*

Advancement in Image Processing and Pattern Recognition
Volume 5 Issue 2
DOI: https://doi.org/10.5281/zenodo.6759892

LSTM stands for short-term memory, a type of RNN (Continuous Neural Network) suitable for sequence prediction problems. Based on the previous text, you can predict what the next name will be. Overcoming RNN limitations with temporary storage has proven to be effective with traditional RNNs. The LSTM can generate relevant information during input processing and through the forgetting gateway and discard non-essential information.

This is how cell structure of LSTM will look like.

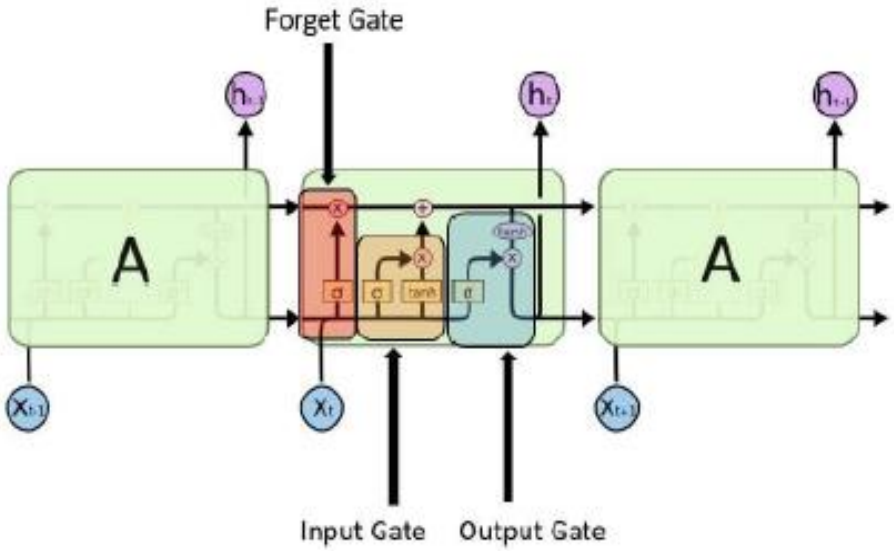


Fig.3:-*Block Diagram of LSTM.*

USER INTERFACE SCREENS

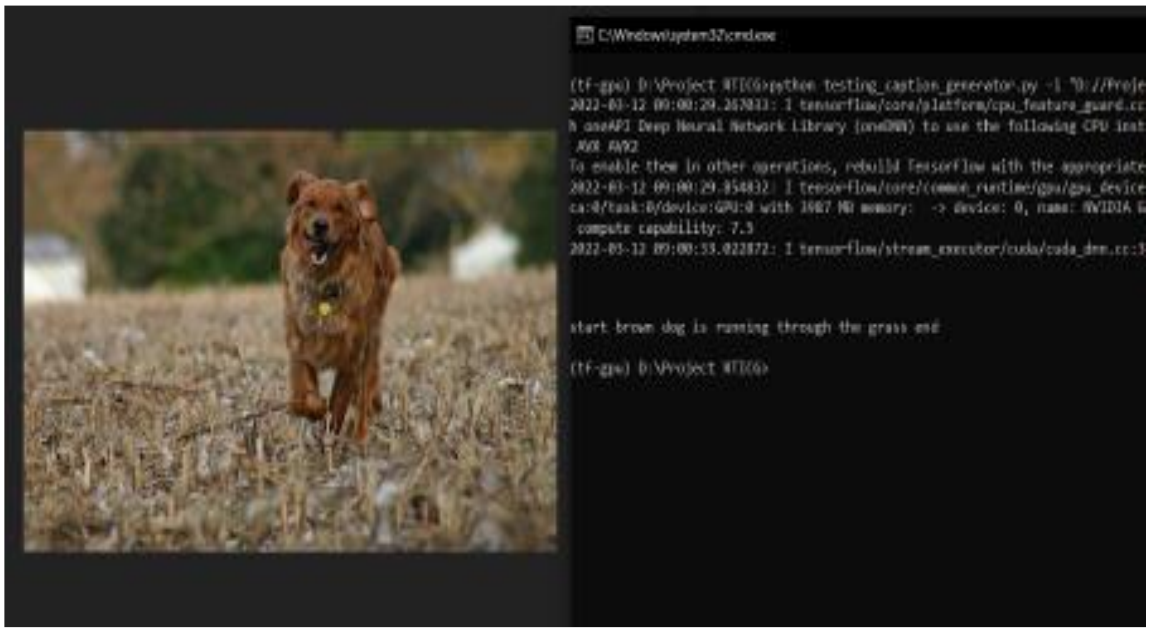


Fig.4:-*User Interface Example 1.*

Advancement in Image Processing and Pattern Recognition
Volume 5 Issue 2
DOI: https://doi.org/10.5281/zenodo.6759892

Fig.5:-*User Interface Example 2.*

Fig.6:-*User Interface Example 3.*

DATA DESIGN

Generate captions on rendered images using caption generator using CNN (Convolutional Neural Networks) and LSTM (Short-term memory).

DATA STRUCTURE

Image assets are exposed in Xception, a CNN model trained in an image database, and a function responsible for creating captions is added to the LSTM model.

Database description-

• Models - Will contain our trained models.

• Descriptions.txt - This text file contains all the images' names and captions after pre-processing.

• Features.p - Embellishment containing an image and its vector feature extracted from the previously trained CNN model Xception.

• Tokenizer.p - Contains a token map with a reference value. Model.png - Visual representation of the size of our project.

• Test caption generator.py - Python file to generate captions for real time image.

• Training generator captions.ipynb - Jupyter notebook where we train and build a caption generator for our image.

Advancement in Image Processing and Pattern Recognition
Volume 5 Issue 2
DOI: https://doi.org/10.5281/zenodo.6759892

COMPONENT DESIGN/ DATA MODEL

DFD Diagram-

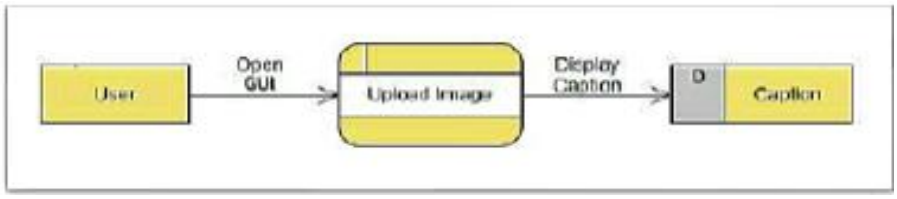


Fig.7:-*DFD Diagram Level 0.*

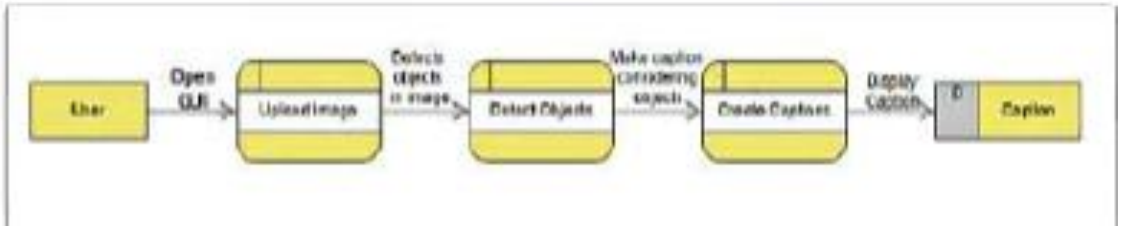


Fig.8:-*DFD Diagram Level 1.*

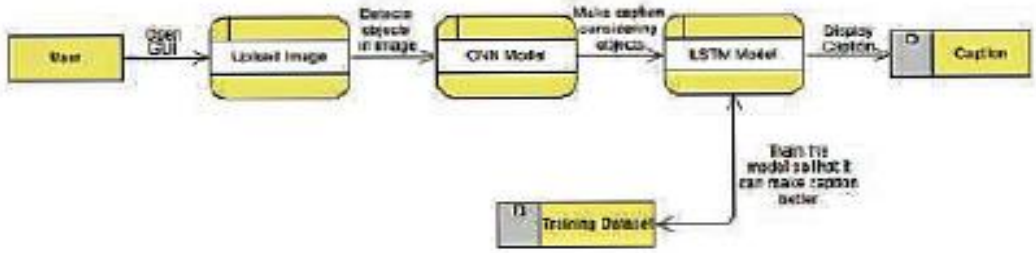


Fig.9:-*DFD Diagram Level 2.*

ACTIVITY DIAGRAM

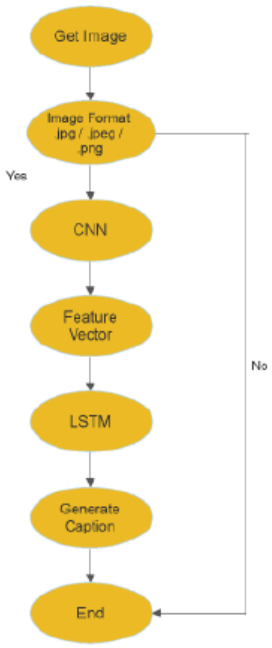


Fig.10:-*Activity Diagram.*

Advancement in Image Processing and Pattern Recognition
Volume 5 Issue 2
DOI: https://doi.org/10.5281/zenodo.6759892

OVERVIEW OF PROJECT MODULES

• Feature Extractor – The feature extracted from the image has a size of 2048, with a thick
layer, it will reduce the size to 256 nodes.
• Sequence Processor – The embedding layer will handle the text input, followed by the LSTM layer.
• Decoder – By combining the output from the two upper layers, the system will be processed in a compact layer to make a final prediction. The final layer will contain a number of nodes equal to our vocabulary size.

TOOLS AND TECHNOLOGY USED

•CNN
•LSTM
•Xception

ALGORITHM DETAILS

• CNN Convolutional Neural Network (CNN) is a deep learning algorithm that can take a captured image, assign values (readable weights and biases) to various items / items in the image, and distinguish them from each other. The initial processing required for a CNN is much less than for other partitioning algorithms. Filters are handcrafted in the old fashioned way, but with sufficient training, CNN can read these filters / symbols.
• Xception model Xception is a convolutional neural network with 71 deep layers. Download a pre-trained version of the network trained with over 1 million photos from the site. Pre-trained networks can categorize images into 1000 object categories such as keyboards, mice, pens, and many animals. As a result, the network has learned to represent the rich features of various images. The image input size of the network is 299x299.
• LSTM long-term memory is a kind of continuous neural network (RNN). The RNN provides the output of the last step as input to the current step. It faces the problem of long-term RNN dependencies, where RNNs can predict names stored in long-term memory, but can provide more accurate predictions from the latest information. If the gap length is long, the RNN will not provide optimal performance. LSTMs can automatically store information longer. Used for processing, forecasting and classification based on time series data.

EXPERIMENTAL SETUP

Data Set

• Flicker8k Dataset:- Dataset folder which contains 8000 images.
• Flickr8k text:-Dataset folder which contains text files and captions of images.

Performance Parameters

• Model Accuracy
• Timing Required to train the model (Around 5 mins for Nvidia 1660 Ti GPU and around 1-2 hours for CPU)
• Timing Required to test the model.

Efficiency Issues

• Since this is a database-trained model, you can predict the names of its members.
• I used a small database of 8000 images. Production-level models require training on datasets containing over 100,000 images. This allows you to create a more accurate model.
• Depending on your system, this process may take some time. When using the Nvidia 1660Ti GPU for training purposes, it can take up to 5 minutes to complete the task. However, it is CPU-intensive and this process can take up to 1 or 2 hours.

Advancement in Image Processing and Pattern Recognition
Volume 5 Issue 2
DOI: https://doi.org/10.5281/zenodo.6759892

SOFTWARE TESTING

Test Cases and Test Results-

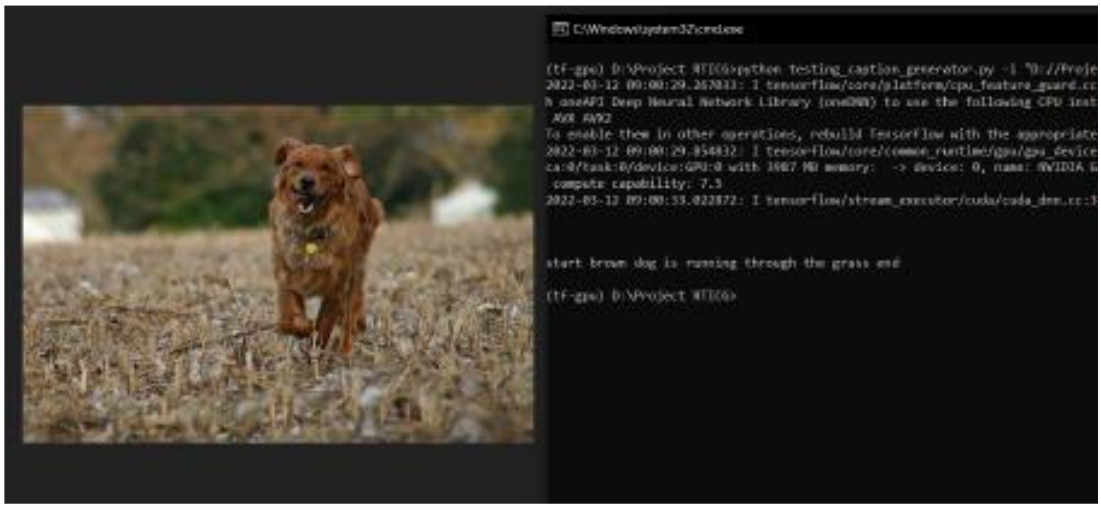


Fig.11:*-Software Testing Example 1.*

Fig.12:*-Software Testing Example 2.*

Fig.13:*-Software Testing Example 3.*

Advancement in Image Processing and Pattern Recognition
Volume 5 Issue 2
DOI: https://doi.org/10.5281/zenodo.6759892

RESULTS

Results Analysis and Discussion

Table 1:*-Results Analysis and Discussion.*

Sr. No.	Generated Caption	Test Case	Size	Testing Time
1	Brown Dog is running through the grass	Test Case 1	126KB	3sec
2	Man is standing on top of mountain	Test Case 2	88KB	3sec
3	Two people are sit-ting on dock overlooking the water	Test Case 3	84KB	3sec

CONCLUSION

Although computer vision has made great strides, it is a relatively new task to allow computers to define transmitted images, although features such as visual acuity, behavioural separation, image separation, feature separation, and scene recognition are possible. A human-like sentence format.

This principle requires real-time capture of image semantically based captions to be expressed in a natural language such as English in the desired way. It has a huge impact on the real world, for example, to help visually impaired people better under-stand the content of images on the web.

Therefore, use CNN to create your own image caption model. Image features are re-moved via CNN. I used a previously trained exception model. Information obtained from the CNN and used by the LSTM to create image captions.

In this project, I created a caption generator to create a CNNRNN model. Another important point is that the model is data dependent and cannot predict words other than itself. I used a small database of 8000 images. Production-level models require training on datasets containing over 100,000 images. This allows you to create a more faithful model.

FUTURE SCOPE

Extend our system by taking large datasets to improve accuracy.

REFERENCES

1. Zhao, W., Wu, X., & Luo, J. (2020). Cross-domain image captioning via cross-modal retrieval and model adaptation. *IEEE Transactions on Image Processing*, *30*, 1180-1192.
2. Huang, Y., Chen, J., Ouyang, W., Wan, W., & Xue, Y. (2020). Image captioning with end-to-end attribute detection and subsequent attributes prediction. *IEEE Transactions on Image processing*, *29*, 4013-4026.
3. Yu, N., Hu, X., Song, B., Yang, J., & Zhang, J. (2018). Topic-oriented image captioning based on order-embedding. *IEEE Transactions on Image Processing*, *28*(6), 2743-2754.
4. Lu, D., Whitehead, S., Huang, L., Ji, H., & Chang, S. F. (2018). Entity-aware image caption generation. *arXiv preprint arXiv:1804.07889*.
5. Hossain, M. Z., Sohel, F., Shiratuddin, M. F., & Laga, H. (2019). A comprehensive survey of deep learning for image captioning. *ACM Computing Surveys (CsUR)*, *51*(6), 1-36.
6. Albawi, S., Mohammed, T. A., & Al-Zawi, S. (2017, August). Understanding of a convolutional neural network. In *2017 international conference on engineering and technology (ICET)* (pp. 1-6). IEEE.
7. Elamri, C., & de Planque, T. (2016). Automated neural image caption generator for visually impaired people. *Department of Computer*

DOI: https://doi.org/10.5281/zenodo.6759892

Science Stanford University, *28*.

8. Vinyals, O., Toshev, A., Bengio, S., & Erhan, D. (2015). Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3156-3164).
9. Karpathy, A., & Fei-Fei, L. (2017). Deep visual-semantic alignments for generating image descriptions. Department of Computer Science.
10. Vinyals, O., Toshev, A., Bengio, S., & Erhan, D. (2015). Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3156-3164).
11. Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735-1780.

Cite this article as: Ankit Patil, Karishma Saudagar, Atul Maharnawar, Tejas Rangatwan, & I. Priyadarshini. (2022). Image Captioning in Real Time. Advancement in Image Processing and Pattern Recognition, 5(2), 1–12. https://doi.org/10.5281/zenodo.6759892