CSE3018 – Content Based Image and Video Retrieval

J Component Report

A project report titled

ACIT: Assistive Image Caption and Tweet

By

19BCE1635 Parth Birthare

BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE AND ENGENEERING

Submitted to

Dr S. Geetha

School of Computer Science and Engineering

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

December 2021

TABLE OF CONTENTS

ABSTRACT

With the advent of more user applications, a larger number of data is generated every second. The data includes of all sorts – structured, unstructured, visual, textual, semi-structured, etc. As is always the case to generate useful information and actionable results, data tracking is very important. Visual data however involves pixeled data and most of this data generated does not have a text or a descriptor associated with it. Therefore, to keep track of this data and to annotate this visual data, the use of artificial intelligence opens up new gates for this work. And to facilitate this, several computer vision techniques help to generate captions that has applications in numerous domains. Examples include editing application recommendation, virtual assistants, image indexing, for visually impaired persons in a way where the image description can be read out to them using gTTS, for social media, and other natural language processing applications. Once such a model is developed it can be used for the above mentioned applications and for new research areas. Moreover, since image captioning automation is a very recent topic and models that accurate are still not developed, this makes the work open to new and better artificial intelligence algorithms in the research domain and develop better models that improves the accuracy of the caption generator. Nevertheless, nowadays Twitter is an inseparable part of most people's lives. Twitter involves many images posted daily, and so there is chance to automate the tweet process too.

INTRODUCTION

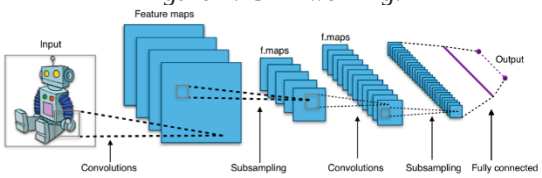
There is a lot of data generated every year [1]. Also, there are many challenges associated with Big Data too [2]. Generating meaningful information from the visual data is feasible by humans interaction and manually annotating and describing features, but it is quite tiresome and involves semantic gaps, that is different people have different views and opinion about an image. However, simple computing solutions have their limitations too. What appears to be a winning moment for a sportsperson where they smile and cry at the same time may be described by different algorithms in different manner. Since a computer is not able to understand the context of an image, complex computer vision and artificial intelligence algorithms have to be implemented. There are many methods to measure the semantic gap in image retrieval [3]. Therefore, to minimize the human involvement, computer vision, NLP and deep learning solves the issue [4].

This work involves the use of deep learning methods – convolution neural networks and recursive neural networks(LSTM – long short term memory) for generating image caption.

Both CNN and LSTM are artificial neural networks. CNN takes in an input image, assign importance or learnable weights and biases to various aspects/objects in the image and is able to differentiate one from the other (**Fig.** 1). It scans images to extract useful features and combine them to classify images. It is RTS (rotation,

translation and scaling) invariant like the HOG features.

Figure 1: CNN working.



LSTM on the other hand stand for Long Short-Term Memory networks are a complex area of deep learning and a type of recurrent neural network that are capable of learning order dependence in sequence prediction problems. It is a type of behavior or constraint that is required in complex problem domains like machine translation, speech recognition, and more. These are having some internal mechanisms (**Fig.** 2) which are known as gates that facilitate the flow of information.

A typical LSTM unit consists of a cell, input gate, output gate and forget gate. The cell is for remembering values over random time intervals whereas the gates regulate information flow ingoing and outgoing from the cell. In simple words, based on the previous word, the next word can be predicted. Unlike RNN which has short term memory, it overcomes it by having a forget gate.

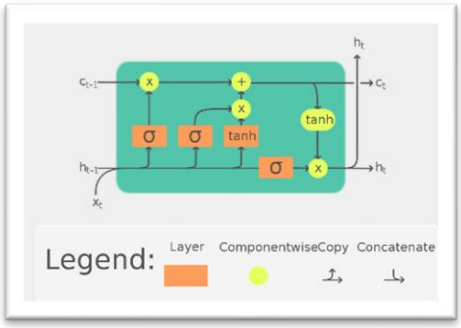


Figure 2: LSTM working.

In terms of assistive nature of the work, important modules like text to speech for the image caption and a tweet bot further decreases the need of human interaction and leads to automation. Playing the description of the image to the user and creating a twitter bot that automatically based on the image adds a tweet text along with the image according to user choice all lead to the efficient use of assistive technology.

LITERATURE SURVEY

Image captioning is a very recent topic and uses computer vision and deep learning techniques to solve the problems. Among the ones mentioned, one such was image captioning using beam search, but the one using CNN and RNN seemed to have more accurate results.

In other works in 2021 6th International Conference on Communication and Electronics Systems (ICCES), Sheshang Degadwala et al. presented a paper Image Captioning Using Inception V3 Transfer Learning Model and they used Inception V3 model which is also an image recognition model [5]. Their paper proposed Inception V3 image caption generator model uses CNN (Coevolutionary Neural Networks) and LSTM (Long Short-Term Memory) units. It involved removal from the InceptionV3model the last classification layer for the dimension (1343,) vector. The embedded matrix is then used for vocabulary connections.

In this work, Xception model is used which is better than the standard Inception model as in Xception model, the Inception modules are replaced by depth wise separable convolutions which lead to better performance.

Another work in 2019 involved A Systematic Literature Review on Image

Captioning by Raimonda Staniut and Dmitrij Šešok where they compared the performance of the models using MS-COCO dataset and concluded that it is not enough to choose between the two datasets MS-COCO and FLICKR30k [6].

Other models included using GAN networks and using Inception models with different datasets too – MSCOCO dataset, Flickr30k dataset.

The models proposed in other papers are also compared with this one and my model score is compared with them.

PROPOSED SYSTEM

The methodology involves making an image caption generator using CNN and LSTM. The features will be extracted from a CNN based model on the ImageNet dataset as mentioned earlier – Xception. Xception is a CNN which is 71 layer-deep. The features are then given to the LSTM model that generates the image caption.

CNN extracts feature from the image where a pre-trained model – Xception will be used. LSTM based on the information/output from CNN will generate a close description of the image. The architectures are thus merged for this work.

The workflow and methodology for one example used for this work is displayed if **Figure 3**.

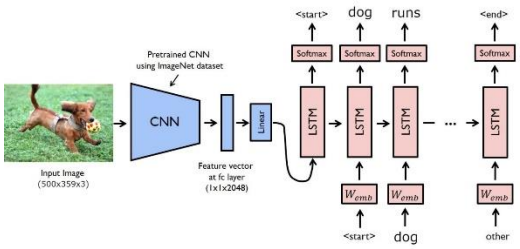


Figure 3: Methodology and principle of the model.

The feature extracted from the image are pushed to the CNN model. The output form the CNN model is taken as an input to the LSTM model. From there it is able to predict the next word from the previous word.

A start and an end tag is also added to the description from which the model can detect the start and end of the description. From the developed model, it will be able to predict for example, from dog, it will predict that the next word is run, from runs it will predict through, from through it will predict the and then grass.

After implementing this, an audio file is generated that describes the image. There is a huge boom in the speech interface in this decade in terms of human computer interaction [7].

Also, an addition feature of automatic tweet generation can be added for improved human computer interaction [8].

The main experiment or the work is sophistically divided into 9 parts:

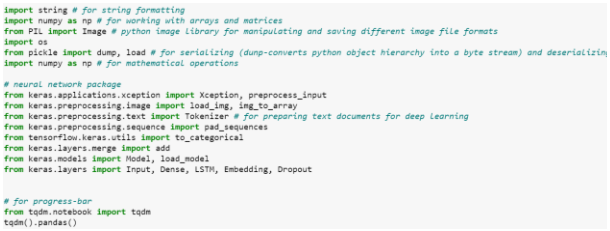
1. Import relevant packages
2. Acquire and perform data cleaning
3. Extract feature vector from images
4. Load dataset for training model
5. Tokenize vocabulary
6. Create a data generator
7. Define a CNN-RNN model
8. Train model
9. Test model

Import relevant packages

Here all the important packages that are necessary for the work are imported. The packages include string, NumPy, PIL, os pickle, neural network packages- keras and tqdm for progress bar. These help in string formatting, for working with arrays and matrices, manipulating and saving different image file formats, for system directory and files related tasks, for serializing and deserializing python object structure, for

neural networks and progress bar respectively.

The imported modules are displayed in **Figure 4.**

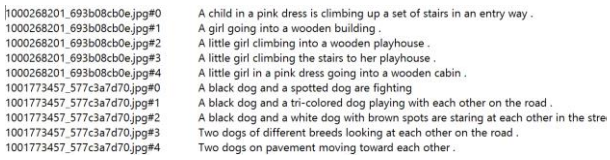


```
import string # for string formatting
import numpy as np # for working with arrays and matrices
from PIL import Image # python image library for manipulating and saving different image file formats
import os
from pickle import dump, load # for serialiizing (dump-converts python object hierarchy into a byte stream) and deserialiizing
import numpy as np # for mathematical operations

# neural network package
from keras.applications.xception import Xception, preprocess_input
from keras.preprocessing.image import load_img, img_to_array
from keras.preprocessing.text import Tokenizer # for preparing text documents for deep learning
from keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.utils import to_categorical
from keras.layers.merge import add
from keras.models import Model, load_model
from keras.layers import Input, Dense, LSTM, Embedding, Dropout


# for progress-bar
from tqdm.notebook import tqdm
tqdm().pandas()
```

Figure 4: Importing all the necessary packages.

Acquire and perform data cleaning

The main data files look like this – with images and their captions separated by a new line. Each image has 5 captions. The file content has been displayed in **Figure 5**. This step involves loading data into a string, mapping captions with their images, clean text, separate unique words to form a vocabulary and storing pre-processed data into a descriptions file.



```
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 .
1001773457_577c3a7d70.jpg#2     A black dog and a white dog with brown spots are staring at each other in the stre
1001773457_577c3a7d70.jpg#3     Two dogs of different breeds looking at each other on the road .
1001773457_577c3a7d70.jpg#4     Two dogs on pavement moving toward each other .
```

Figure 5: Captions file.

Extract feature vector from images

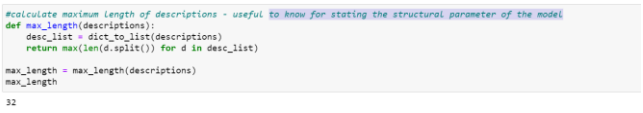
Here, transfer learning is used and a pretrained Xception model is used which is trained on large datasets and extract features from that model. The features dictionary is dumped to features file. The pickle object contains image and its feature vector extracted from the Xception pre-trained CNN model.

Load dataset for training model

Here, the model is trained using the Flickr_8k.trainImages.txt. It contains about 6000 images for training the model.

Tokenize vocabulary

Since human readable language are not understandable by computers, the data is tokenized.
Each word of vocabulary is mapped with a unique index value.
These tokens are created from the vocabulary using the keras tokenizer function and saved to tokenizer file. The maximum length of descriptions are also calculated (**Fig.** 6) to know for stating the structural parameter of the model.



```
#calculate maximum length of descriptions - useful to know for stating the structural parameter of the model
def max_length(descriptions):
    desc_list = dict_to_list(descriptions)
    return max(len(d.split()) for d in desc_list)

max_length = max_length(descriptions)
max_length
```

```
32
```

Figure 6: Calculating maximum length of descriptions.

Create a data generator

Each image from the 6000 images on which the model has to be trained has feature vector length of 2048 and even the captions are represented as numbers.

Since data for 6000 images is not possible to be held in memory, a generator method that derives batches is used.

Define a CNN-RNN model

In the model, these are the main functions:

- Extract features from pretrained model Xception.
- Word embedding layer that handles text, followed by LSTM
- Model 1 and model 2 are merged together and processed by dense layer to make final prediction since both 1 and 2 model produce fixed length vector.

A visual representation of the model is created (**Fig.** 7).

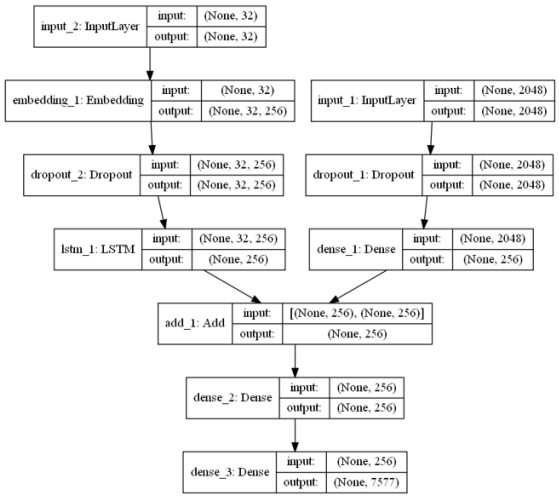


Figure 7: Visual representation of model.

Train model

Here, the 6000 training images are used for training the model by generating the input and output sequences in batches and fitting them to the model using model.fit_generator() method. The model is saved in the models directory. **Figure 8,9** represents the output after training the model.

```
Dataset:  6000
Descriptions: train= 6000
Photos: train= 6000
Vocabulary Size: 7577
Description Length:  32
Model: "model_1"
__________________________________________________________________________________________________
 Layer (type)                Output Shape              Param #   Connected to
==================================================================================================
 input_5 (InputLayer)        [(None, 32)]              0         []

 input_4 (InputLayer)        [(None, 2048)]            0         []

 embedding_1 (Embedding)     (None, 32, 256)           1939712   ['input_5[0][0]']

 dropout_2 (Dropout)         (None, 2048)              0         ['input_4[0][0]']

 dropout_3 (Dropout)         (None, 32, 256)           0         ['embedding_1[0][0]']
```

Figure 8: Output after training the model.

```
 dense_3 (Dense)             (None, 256)               524544    ['dropout_2[0][0]']

 lstm_1 (LSTM)               (None, 256)               525312    ['dropout_3[0][0]']

 add_13 (Add)                (None, 256)               0         ['dense_3[0][0]',
                                                                  'lstm_1[0][0]']

 dense_4 (Dense)             (None, 256)               65792     ['add_13[0][0]']

 dense_5 (Dense)             (None, 7577)              1947289   ['dense_4[0][0]']

==================================================================================================
Total params: 5,002,649
Trainable params: 5,002,649
Non-trainable params: 0
```

Figure 9: Output after training the model.

Test model

The model is then tested against testing images.

Figure 10,11,12 depicts the model result for generated caption.

Figure 10: Dog running through grass picture with generated caption.

Figure 11: Man picture with generated image caption.

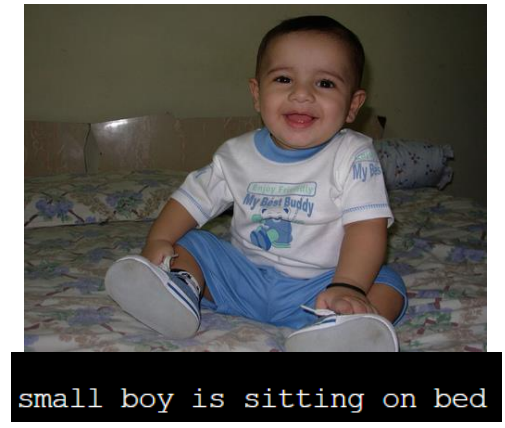


Figure 12: Boy picture on a bed image caption.

What the model also creates is an audio file for the generated image along with the caption. This audio automatically gets played after the caption is generated. This helps for visually impaired people to understand the context of the picture and get a closely matching description for it.

RESULTS AND DISCUSSION

The images and their description for the existing works and the new proposed model is listed in the **Figure** 13, 14, 15, 16.

Figure 13: Image for dog coming out of water on sand using my model.

Caption: black dog is running through the grass

Figure 14: Caption for dog coming out of water using existing model.

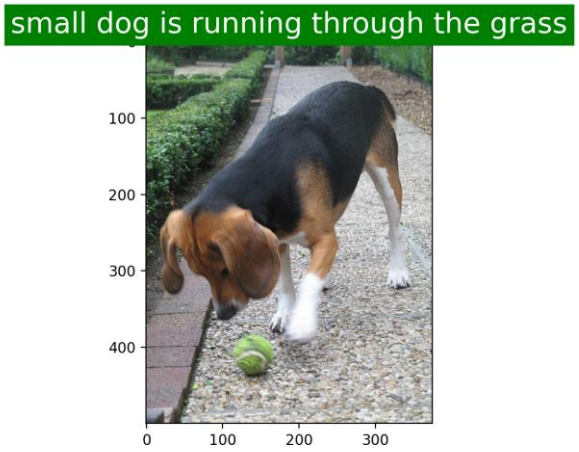
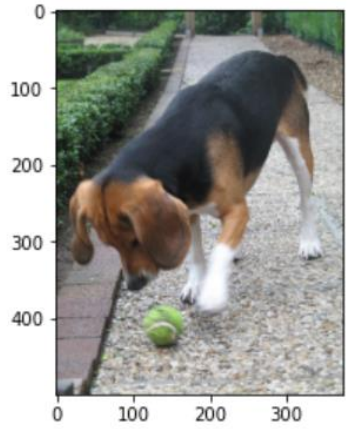


Figure 15: Caption for small dog playing with ball on the grass with ball using my model.



Caption: dog is running on the grass

Figure 16: Caption for small dog playing with ball on the grass with ball using existing model.

Figure 17: Caption for a boy playing in water using my model.

Caption: boy in red shirt is jumping into pool

Figure 18: Caption for a boy playing in water using existing model.

Visual results show that my model generates captions for images better than the existing model and besides that it also generates an audio for that caption with also an option for tweeting about the same.

Comparing the BLEU scores of my model and the existing models, it is found that my model showed better accuracy and a good score concluding that it is better.

The BLEU score for various model and my model showed that it is comparatively better than most models but is still lagging though not far behind the show, attend and tell model.

Thus it is concluded that my model is considerably good than the other models and is substantial to generate captions from them.

EXPERIMENTAL SETUP

The dataset is Flickr8k dataset which consists of around 8000 images out of which 6000 are training and remaining are testing images for the model.

The hardware and the software requirements involves having the required libraries in the caption generator file including TensorFlow, gTTs, translate and pillow, where the hardware is at least a 4 cores system and graphic card is a supplement for faster execution.

CONCLUSION

Therefore AICT is implemented and solves the various problems mentioned in the aim and motivation by generating the image caption and narrates the same to the user. Further this model was found to be better than the previously generated models.

The proposed system will help to automatically generate captions and tweet them according to the user's choice. Moreover, the test to speech module will allow people to listen to the audio as well.

This system model is found to be better than the model generated using Inception V3 which was used in the existing works. Therefore a better model is developed.

If better computing capacity is obtained, the images can be trained on bigger datasets and more complex models can be tried, but due to computational constraints, this is the best model with this kind of computational power and dataset.

The future works for this project at this stage is to convert this to a paper and start looking for publishers for it, and possibly create an application for opening it for public use so that people could use it in everyday life.

REFERENCE

R.Devakunchari, “Analysis on big data over the years.” *International Journal of Scientific and Research*, 2014, 4, 1.

Sivarajah, U., Kamal, M., Irani, Z. and Weerakkody, V., “Critical analysis of Big Data challenges and analytical methods”. *Journal of Business Research*, 2014, 70, pp.263-286.

C. Liu and G. Song, "A method of measuring the semantic gap in image retrieval: Using the information theory," 2011 International Conference on Image Analysis and Signal Processing, 2011, pp. 287-291, doi: 10.1109/IASP.2011.6109048.

Valencia-García, R. and García-Sánchez, F., “NATURAL LANGUAGE PROCESSING and Human–Computer Interaction”. *Computer Standards & Interfaces*, 2013, 35(5), pp.415-416.

S. Degadwala, D. Vyas, H. Biswas, U. Chakraborty and S. Saha, "Image Captioning Using Inception V3 Transfer Learning Model," 2021 6th International Conference on Communication and Electronics Systems (ICCES), 2021, pp. 1103-1108, doi: 10.1109/ICCES51350.2021.9489111.

Staniūtė, R. and Šešok, D., “A Systematic Literature Review on Image Captioning.” *Applied Sciences, 2019,* 9(10), p.2024.

Clark, L., Doyle, P., Garaialde, D., Gilmartin, E., Schlögl, S., Edlund, J., Aylett, M., Cabral, J., Munteanu, C., Edwards, J. and R Cowan, B., “The State of Speech in HCI: Trends, Themes and Challenges”. *Interacting with Computers*, 2019, 31(4), pp.349-371.

Lloret, E. and Palomar, M., “Analysing and evaluating the task of automatic tweet generation: Knowledge to business.” Computers in Industry, 2016, 78, pp.3-15.