Kathmandu University

Department of Computer Science and Engineering

Dhulikhel, Kavre

A Project Report on

"Image Caption Generator "

[COMP 473]

Submitted by
Shirshak Bajgain (4)
Samyam Shrestha (15)
Krishna Gaire (10)

Submitted to

Dr. Bal Krishna Bal

Table of Content

TABLE OF CONTENTS

List of Figure

Chapter 1: Introduction

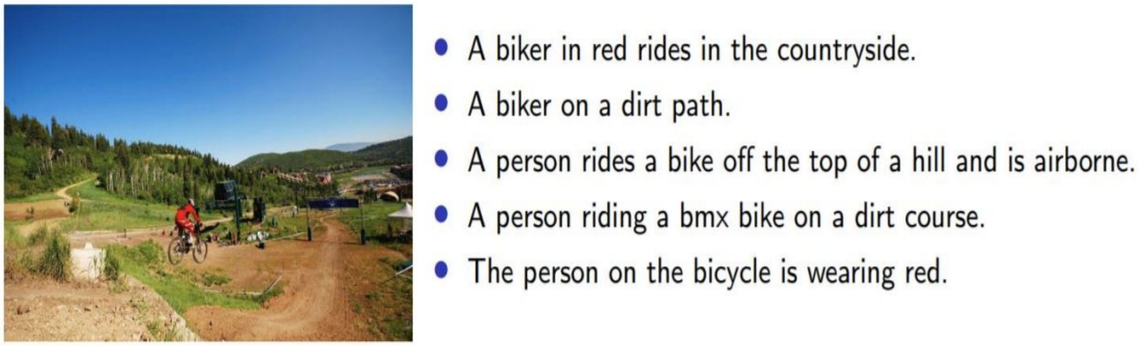
“Image Caption Generator” is a python application that scans an image and generates the caption according to image. Image captioning is an interesting problem, where we can make computer learn both computer vision techniques and natural language processing techniques.

Computer vision has become ubiquitous in our society, with applications in several fields. In this project, we focus on one of the visual recognition facets of computer vision, i.e image captioning. The problem of generating language descriptions for visual data has been studied for a long time but in the field of videos. In the recent few years emphasis has been lead on still image description with natural text. Due to the recent advancements in the field of object detection, the task of scene description in an image has become easier.

Chapter 2: Design and Implementation

2.1 Training and Testing Set:

For the task of image captioning, we use Flickr8k dataset. The dataset contains 8000 images with 5 captions per image. The dataset by default is split into image and text folders. Each image has a unique id and the caption for each of these images is stored corresponding to the respective id. The dataset contains 6000 training images, 1000 development images and 1000 test images. A sample from the data is given below:



- A biker in red rides in the countryside.
- A biker on a dirt path.
- A person rides a bike off the top of a hill and is airborne.
- A person riding a bmx bike on a dirt course.
- The person on the bicycle is wearing red.

Fig 2.1.1: Sample image and corresponding captions from the Flickr8k dataset

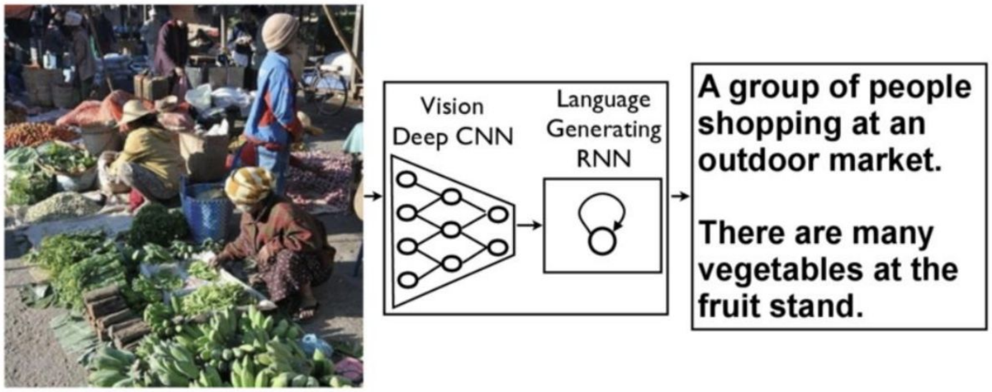
Other datasets like Flickr30k and MSCOCO for image captioning exist but both these datasets have more than 30,000 images thus processing them becomes computationally very expensive. Captions generated using these datasets may prove to be better than the ones generated after training on Flickr8k because the dictionary of words used by RNN decoder would be larger in case of Flickr30k and MSCOCO.

2.2 Training the Model:

We used Recurrent Neural Network and Convolutional Neural Network which helps in detecting the object present in the image. Recurrent Neural Network help is generating meaningful sentences.

2.3 Model Architecture:

The model proposed takes an image I as input and is trained to maximize the probability of p(S|I) [1] where S is the sequence of words generated from the model and each word St Is generated from a dictionary built from the training dataset. The input image I is fed into a deep vision Convolutional Neural Network (CNN) which helps in detecting the objects present in the image. The image encodings are passed on to the Language Generating Recurrent Neural Network (RNN) which helps in generating a meaningful sentence for the image. An analogy to the model can be given with a language translation RNN model where we try to maximize the p(T|S) where T is the translation to the sentence S. However, in our model the encoder RNN which helps in transforming an input sentence to a fixed length vector is replaced by a CNN encoder. Recent research has shown that CNN can easily transform an input image into a vector.



A group of people shopping at an outdoor market.

There are many vegetables at the fruit stand.

Fig 2.2.1: An overview of the image captioning model

The details of the models are discussed in the following section. A Long Short-Term Memory(LSTM) network follows the pre trained VGG16 [2]. The LSTM network is used for language generation. LSTM differs from traditional Neural Networks as a current token is dependent on the previous tokens for a sentence to be meaningful and LSTM networks take this factor into account. In the following sections, we discuss the components of the model i.e. the CNN encoder and the Language generating RNN in details.

Convolutional Neural Networks (ConvNets or CNNs) are a category of Artificial Neural Networks which have proven to be very effective in the field of image recognition and classification. They have been used extensively for the task of object detection, self-driving cars, image captioning etc. The first convnet was discovered in the year 1990 by Yann Lecun and the architecture of the model was called as the LeNet architecture. A basic convnet is shown in the fig. below

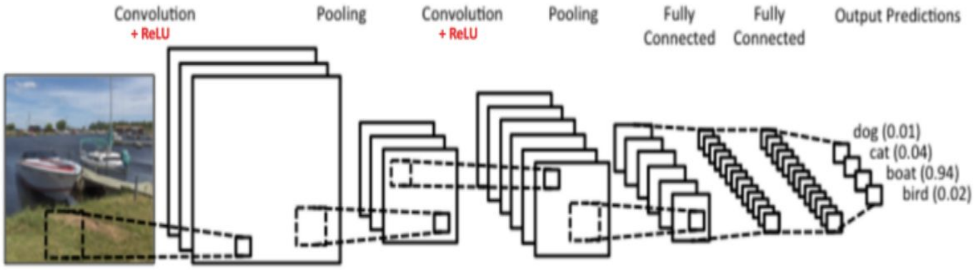
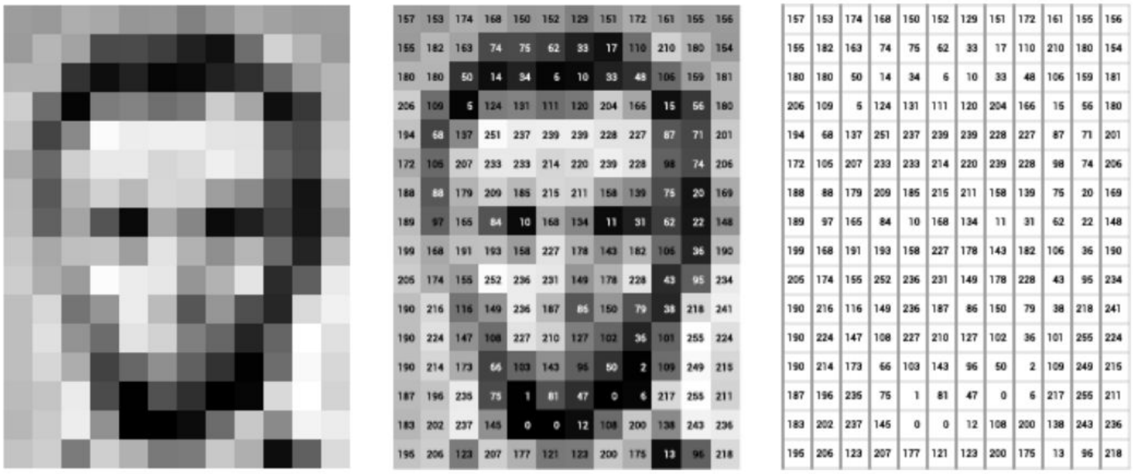


Fig 2.3.1 Convolutional Neural Network

The entire architecture of a convnet can be explained using four main operations namely,

1. Convolution
2. Non- Linearity (ReLU)
3. Pooling or Sub Sampling
4. Classification (Fully Connected Layer)

These operations are the basic building blocks of every Convolutional Neural Network, so understanding how these work is an important step to developing a sound understanding of ConvNets. We will discuss each of these operations in detail below. Essentially, every image can be represented as a matrix of pixel values. An image from a standard digital camera will have three channels – red, green and blue – you can imagine those as three 2d-matrices stacked over each other (one for each color), each having pixel values in the range 0 to 255.



157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

Fig. 2.3.2: A grayscale image as matrix of numbers

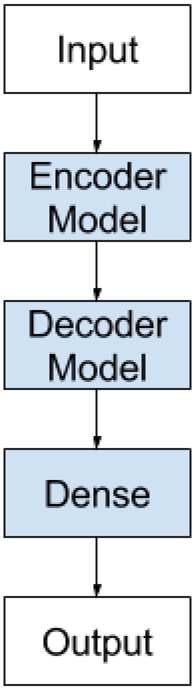
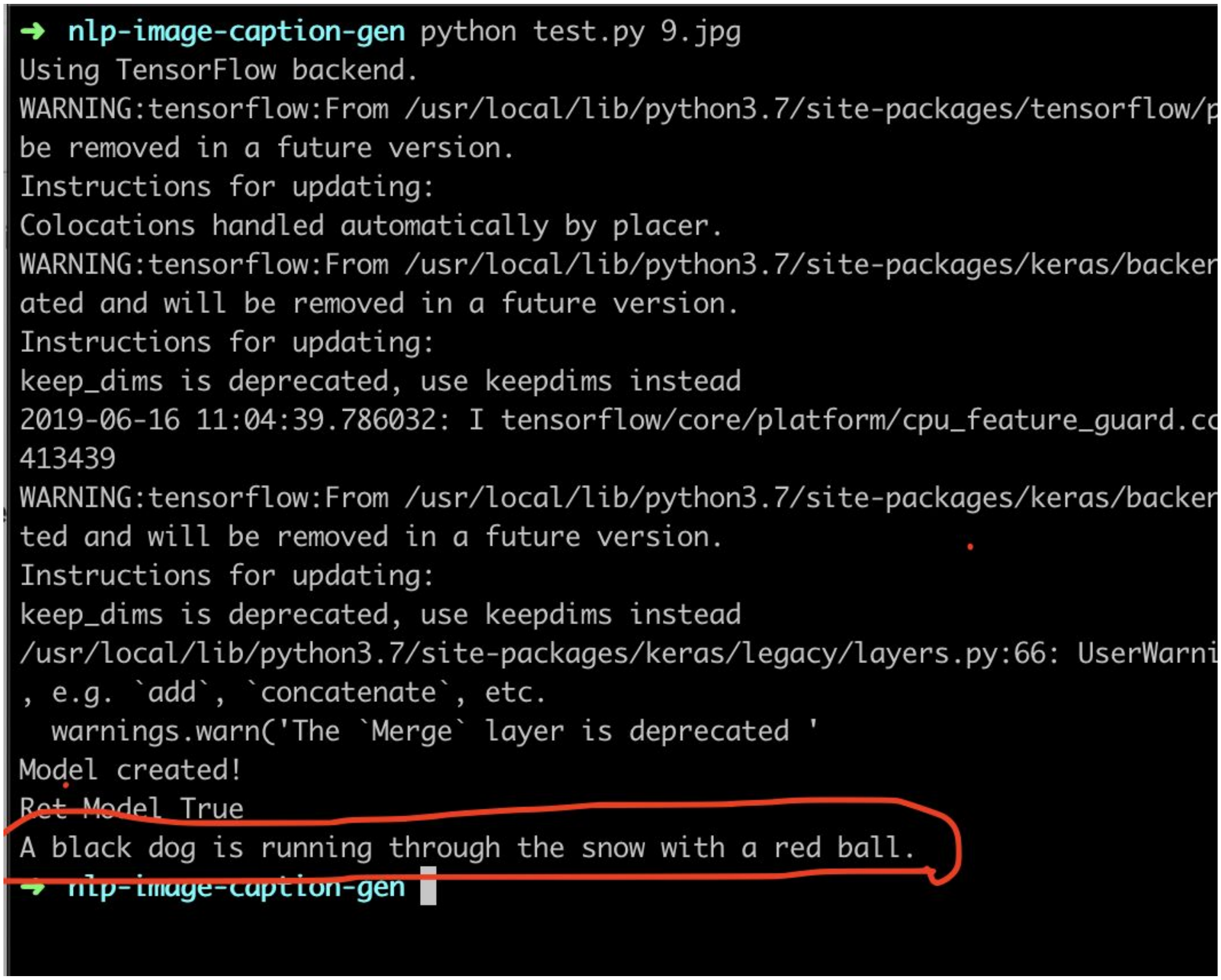


Fig. 2.3.3 Encoder-Decoder LSTM Model Architecture

2.4 Prediction

Using the script is very easy. We just put image inside test data folder and it will try to generate the caption



```
➜  nlp-image-caption-gen python test.py 9.jpg
Using TensorFlow backend.
WARNING:tensorflow:From /usr/local/lib/python3.7/site-packages/tensorflow/p
be removed in a future version.
Instructions for updating:
Colocations handled automatically by placer.
WARNING:tensorflow:From /usr/local/lib/python3.7/site-packages/keras/backen
ated and will be removed in a future version.
Instructions for updating:
keep_dims is deprecated, use keepdims instead
2019-06-16 11:04:39.786032: I tensorflow/core/platform/cpu_feature_guard.cc
413439
WARNING:tensorflow:From /usr/local/lib/python3.7/site-packages/keras/backen
ted and will be removed in a future version.
Instructions for updating:
keep_dims is deprecated, use keepdims instead
/usr/local/lib/python3.7/site-packages/keras/legacy/layers.py:66: UserWarni
, e.g. `add`, `concatenate`, etc.
  warnings.warn('The `Merge` layer is deprecated '
Model created!
Ret Model True
A black dog is running through the snow with a red ball.
➜  nlp-image-caption-gen
```

Figure 2.4.1: Testing image

2.5 Accuracy

The dataset on these convnets yield an accuracy of 85% within around 1.5 hrs of training on gpus. The plots of loss and accuracy on test and validation set are shown in the figures below.

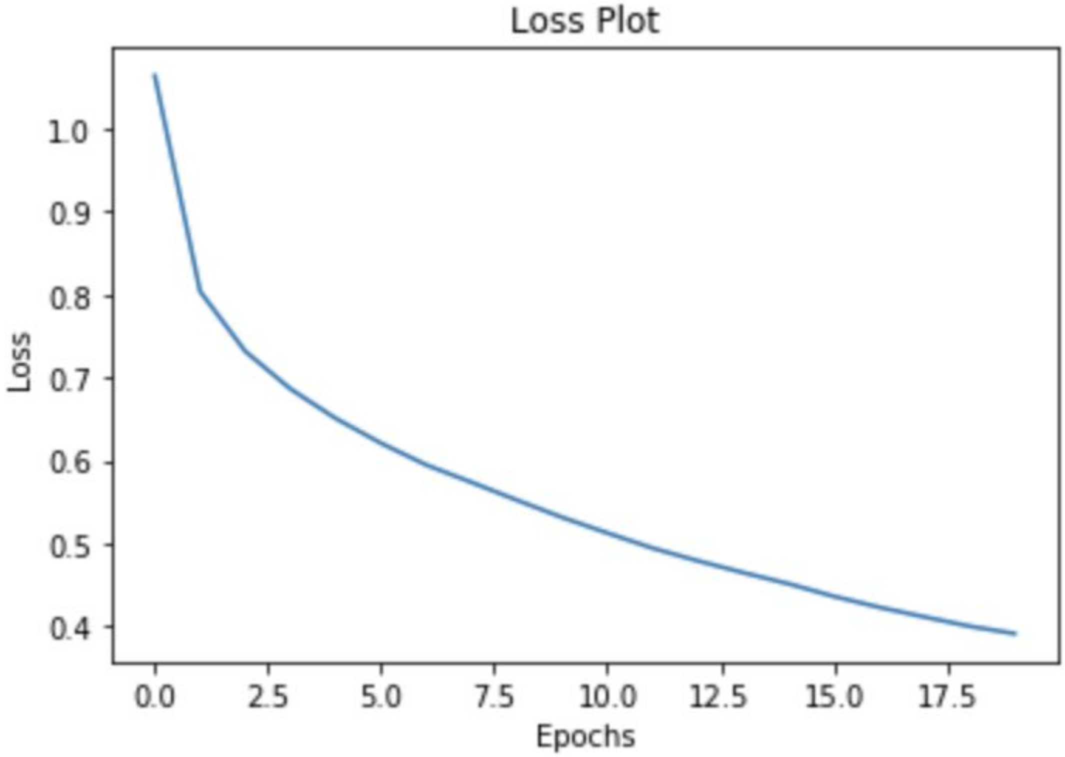


Fig. 2.5.1:Loss plot on training and validation set

The model is built using tensorflow. Tensorflow is an open source library developed by Google brain team for machine learning. Though being a python api, most of the code of tensorflow is written in C++ and CUDA which is nvidia's programming language for gpus. This helps tensorflow in faster execution of code since python is slower than C++. Also, the use of gpu enhances the performance of the code significantly

Chapter 3: System Requirement Specification

3.1. Software Specification

- **Python**

  Python is an interpreted, high-level, general-purpose programming language. Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

  It is the programming language of our application.

- **Tensorflow**

  TensorFlow is an open source software library for high-performance numerical computation. Its flexible architecture allows easy deployment of computation across a variety of platforms (CPUs, GPUs, TPUs), and from desktops to clusters of servers to mobile and edge devices.

  It has been used to implement the mobile-net architecture.

- **Numpy**

  NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

  It has been used to manipulate image vectors in our application.

3.2 Hardware Specification

The minimum configuration of the system to run the work smoothly is:

- 8 GB RAM

Chapter 4. Conclusion

Our end-to-end system neural network system is capable of viewing an image and generating a reasonable description in English depending on the words in its dictionary generated on the basis of tokens in the captions of train images. The model has a convolutional neural network encoder and an LSTM decoder that helps in generation of sentences. The purpose of the model is to maximize the likelihood of the sentence given the image. Experimenting the model with Flickr8K dataset show decent results. We evaluate the accuracy of the model on the basis of the BLEU score. The accuracy can be increased if the same model is worked upon a bigger dataset. Furthermore, it will be interesting to see how one can use unsupervised data, both from images alone and text alone, to improve image description approaches.

Chapter 5: Evaluation

Reference

[1] Vinyals, Oriol, et al. "Show and tell: A neural image caption generator." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

[2] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).

[3] Fang, Hao, et al. "From captions to visual concepts and back." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

[4] Karpathy, Andrej, and Li Fei-Fei. "Deep visual-semantic alignments for generating image descriptions." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015.

[5] Johnson, Justin, Andrej Karpathy, and Li Fei-Fei. "Densecap: Fully convolutional localization networks for dense captioning." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016.

[6] Wang, Cheng, et al. "Image captioning with deep bidirectional LSTMs." Proceedings of the 2016 ACM on Multimedia Conference. ACM, 2016.

[7] Xu, Kelvin, et al. "Show, attend and tell: Neural image caption generation with visual attention." International Conference on Machine Learning. 2015.

[8] Papineni, Kishore, et al. "BLEU: a method for automatic evaluation of machine translation." Proceedings of the 40th annual meeting on association for computational linguistics. Association for Computational Linguistics, 2002.

[9] Vedantam, Ramakrishna, C. Lawrence Zitnick, and Devi Parikh. "Cider: Consensus-based image description evaluation." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.