Image Caption Generator

Introduction:

Using deep learning techniques, today we can solve many tasks which were previously known to be unsolvable or very complex to be framed for computer systems. With today’s computing power and deep learning frameworks available, tasks such as object detection, sentiment analysis, natural language translation and many more are possible.

In in this project, I have tried to solve one very interesting problem i.e., generating captions for the input image. We humans tend to identify and provide suitable description of any image just by looking at the image, but this trivial looking task is very complex to handle for any computing device but using deep learning frameworks such CNN (Convolutional Neural Network) and LSTM-RNN (Long Short-Term Memory – Recurrent Neural Network) we can frame this task to be solvable.

For this project I have used **CNN and LSTM-RNN** with most widely used **Flicker8k dataset** containing **8091 images** in total with **captions in separate .txt file.**

For this task I have used a high-performance computing device containing 8GB of GDDR6 RTX 3070 GPU memory and a 8 core 16 thread Ryzen 7 5800x CPU paired with 16gb of RAM 3200mhz.

Literature Review:

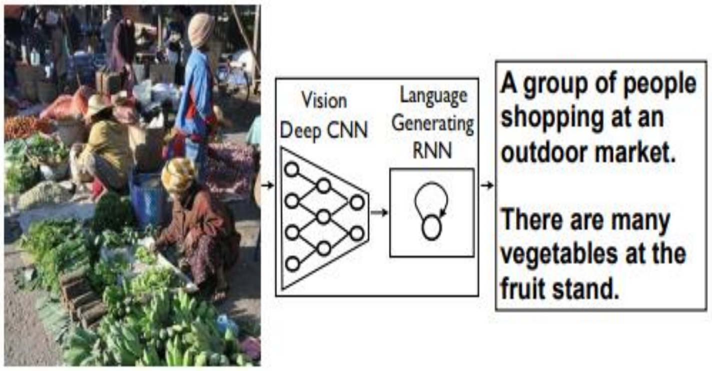
Paper -1:

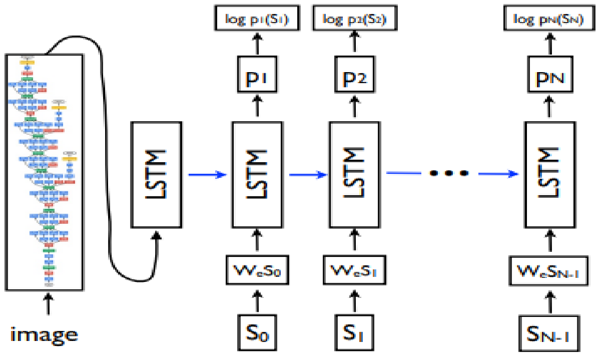
https://www.cv-foundation.org/openaccess/content_cvpr_2015/html/Vinyals_Show_and_Tell_2015_CVPR_paper.html

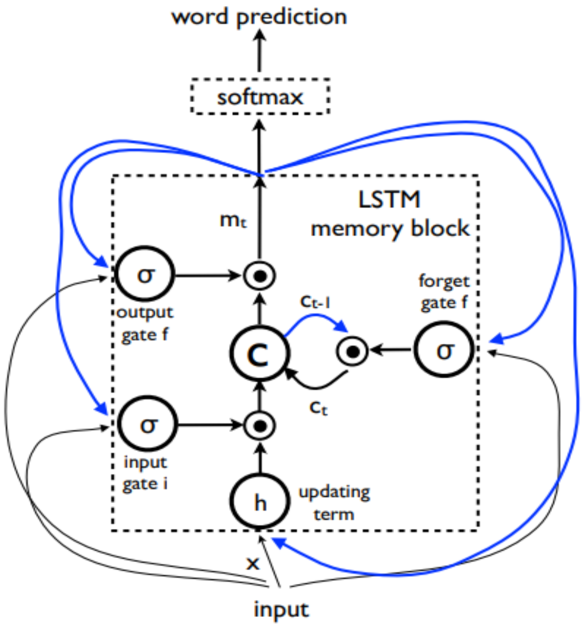
Title: *"A Neural Image Caption Generator"*

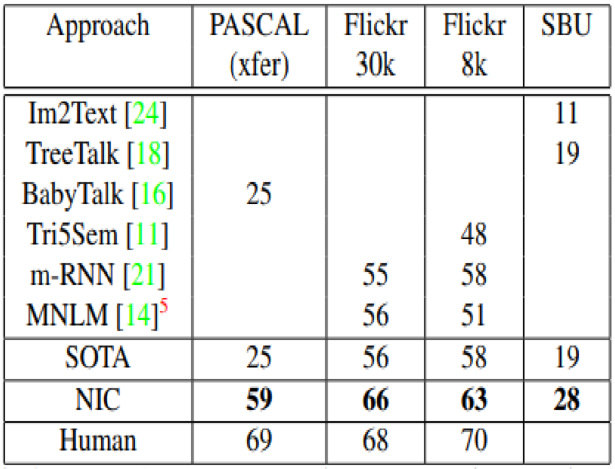
Abstract:

Using CNN architecture to fetch a feature vector from input image and then using LSTM-RNN on datasets ranging from small to large datasets.


Vision
Deep CNN
Language
Generating
RNN
A group of people shopping at an outdoor market.
There are many vegetables at the fruit stand.



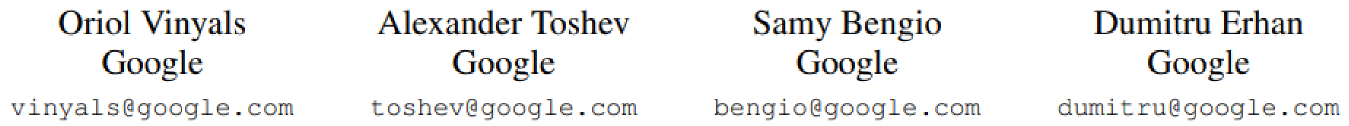




Approach	PASCAL (xfer)	Flickr 30k	Flickr 8k	SBU
Im2Text [24]				11
TreeTalk [18]				19
BabyTalk [16]	25			
Tri5Sem [11]			48	
m-RNN [21]		55	58	
MNLM [14][5]		56	51	
SOTA	25	56	58	19
NIC	**59**	**66**	**63**	**28**
Human	69	68	70	

Table 2. BLEU-1 scores. We only report previous work results when available. SOTA stands for the current state-of-the-art.

Given By :



Oriol Vinyals
Google
vinyals@google.com

Alexander Toshev
Google
toshev@google.com

Samy Bengio
Google
bengio@google.com

Dumitru Erhan
Google
dumitru@google.com

Year: 2015

Paper -2:

http://www.aclweb.org/anthology/P02-1040.pdf

Title: *“BLEU: a Method for Automatic Evaluation of Machine Translation”*

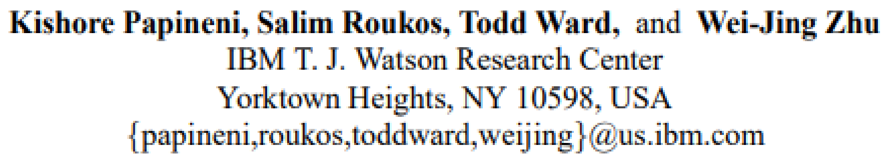
Abstract:

This paper suggested a metric for evaluating the predictions made by automatic machine translation systems known as “*Bilingual Evaluation Understudy Score*”.

According to BLEU:

“The primary programming task for a BLEU implementor is to compare n-grams of the candidate with the n-grams of the reference translation and count the number of matches. These matches are position-independent. The more the matches, the better the candidate translation is.”

Given By:



Kishore Papineni, Salim Roukos, Todd Ward, and **Wei-Jing Zhu**
IBM T. J. Watson Research Center
Yorktown Heights, NY 10598, USA
{papineni,roukos,toddward,weijing}@us.ibm.com

Year : 2002

Paper -3:

https://arxiv.org/abs/1308.0850v5

Title: “*Generating Sequences With Recurrent Neural Networks*”

Abstract:

This demonstrates how a deep recurrent neural network works in predicting a sequence and also proposes LSTM framework.

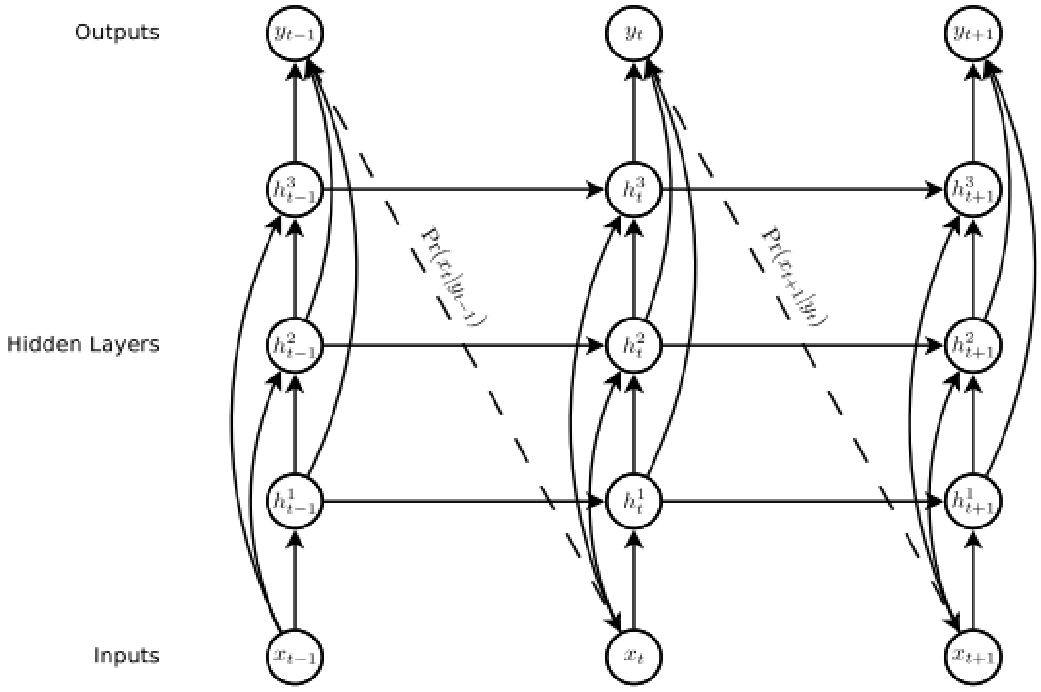


Figure 1: **Deep recurrent neural network prediction architecture.** The circles represent network layers, the solid lines represent weighted connections and the dashed lines represent predictions.

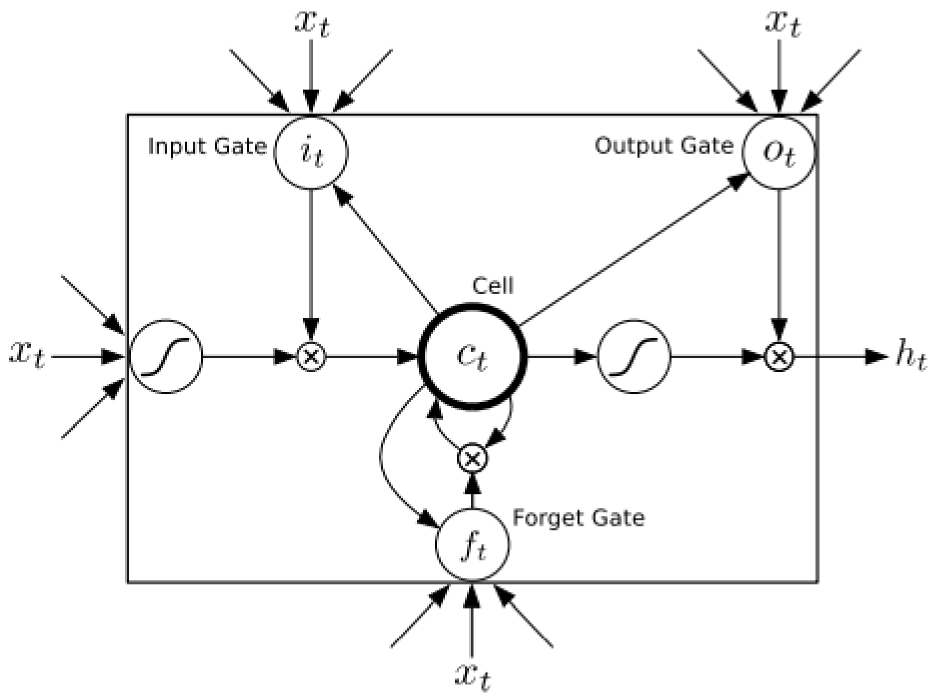
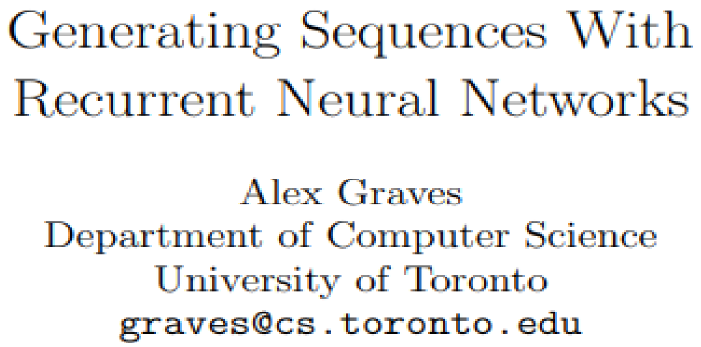


Figure 2: **Long Short-term Memory Cell**

Given By:



Generating Sequences With
Recurrent Neural Networks

Alex Graves
Department of Computer Science
University of Toronto
graves@cs.toronto.edu

Year: 2013

Paper -4:

https://aclanthology.org/D14-1179.pdf

Title : “*Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation*”

Abstract:

This paper proposed a novel neural network model called RNN Encoder - Decoder that consists of two recurrent neural networks (RNN). One RNN encodes a sequence of symbols into a fixed-length vector representation, and the other decodes the representation into another sequence of symbols. The encoder and decoder of the proposed model are jointly trained to maximize the conditional probability of a target sequence given a source sequence.

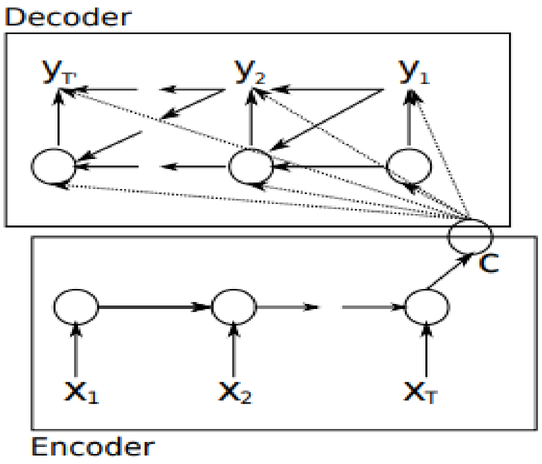


Figure 1: An illustration of the proposed RNN Encoder–Decoder.

Given By:

Kyunghyun Cho
Bart van Merriënboer **Caglar Gulcehre**
Université de Montréal
firstname.lastname@umontreal.ca

Dzmitry Bahdanau
Jacobs University, Germany
d.bahdanau@jacobs-university.de

Fethi Bougares **Holger Schwenk**
Université du Maine, France
firstname.lastname@lium.univ-lemans.fr

Yoshua Bengio
Université de Montréal, CIFAR Senior Fellow
find.me@on.the.web

Year: 2014

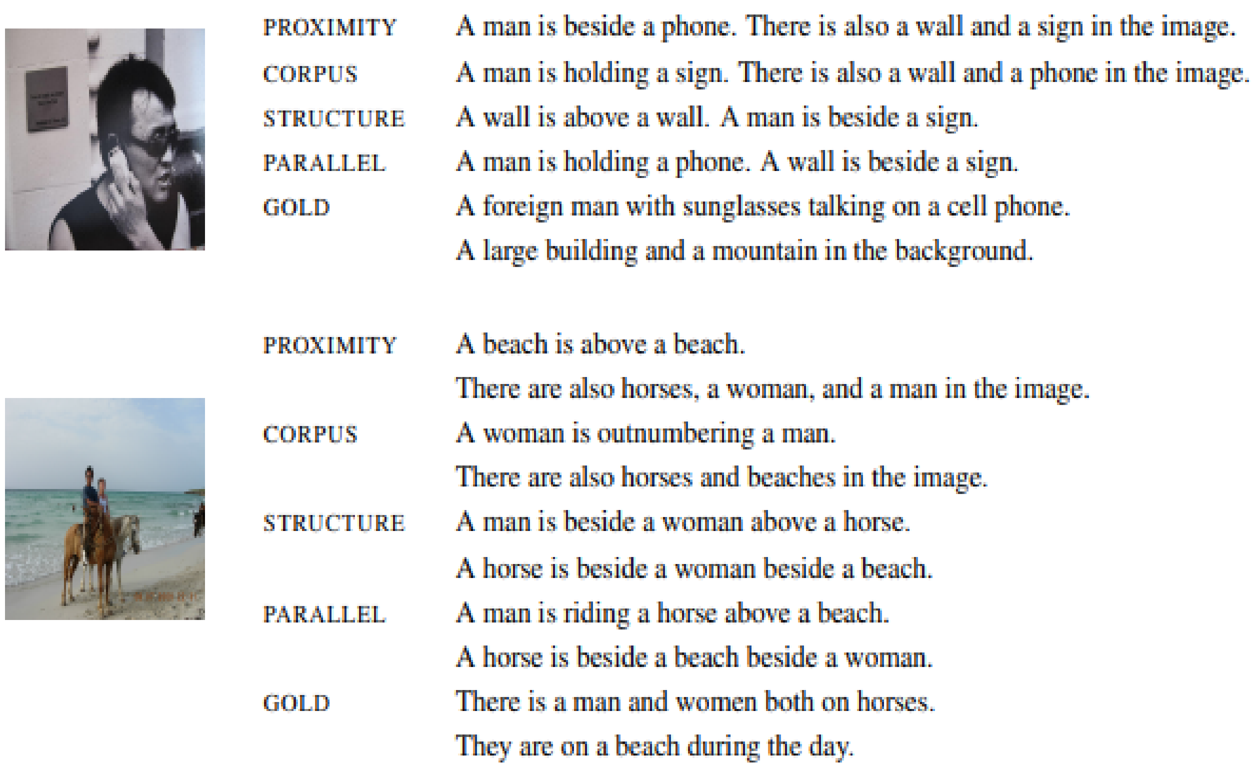
Paper -5:

https://homepages.inf.ed.ac.uk/keller/papers/emnlp13a.pdf

Title: "*Image Description using Visual Dependency Representations*"

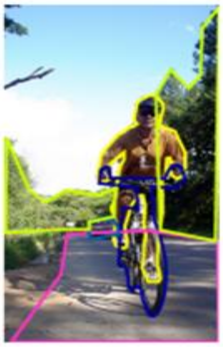
Abstract:

Describing the main event of an image involves identifying the objects depicted and predicting the relationships between them.



PROXIMITY	A man is beside a phone. There is also a wall and a sign in the image.
CORPUS	A man is holding a sign. There is also a wall and a phone in the image.
STRUCTURE	A wall is above a wall. A man is beside a sign.
PARALLEL	A man is holding a phone. A wall is beside a sign.
GOLD	A foreign man with sunglasses talking on a cell phone. A large building and a mountain in the background.
PROXIMITY	A beach is above a beach. There are also horses, a woman, and a man in the image.
CORPUS	A woman is outnumbering a man. There are also horses and beaches in the image.
STRUCTURE	A man is beside a woman above a horse. A horse is beside a woman beside a beach.
PARALLEL	A man is riding a horse above a beach. A horse is beside a beach beside a woman.
GOLD	There is a man and women both on horses. They are on a beach during the day.

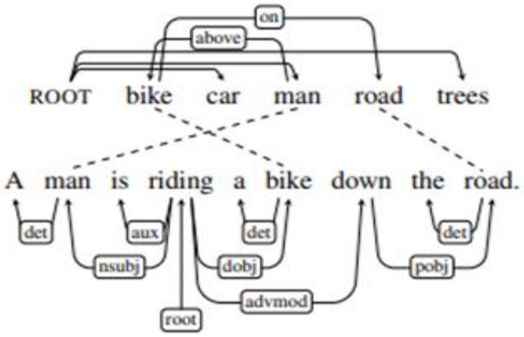
Figure 4: Some example descriptions produced by PROXIMITY, CORPUS, STRUCTURE and PARALLEL.



(a)

A man is riding a bike down the road.
A car and trees are in the background.

(b)



Given By:

Desmond Elliott
School of Informatics
University of Edinburgh
d.elliott@ed.ac.uk

Frank Keller
School of Informatics
University of Edinburgh
keller@inf.ed.ac.uk

Year: 2013

Dataset used:

Name	Flicker8k
Total Images	8091
Total Captions	8091
Training Split	7000
Testing Split	1091
Downloaded From	Kaggle

Proposed architecture:

This project uses combination of CNN and LSTM layers to produce output caption given an image.

Before starting with Model, I cleaned the captions in dataset by removing unwanted punctuation marks and grammar in the sentences. After cleaning the caption we must save it in local storage for further use.

I have saved it in descriptions.txt file.

A **Pre-trained CNN model** is first used Known as **Xception.**

Xception model is available in Keras .application.

Features of Xception model:

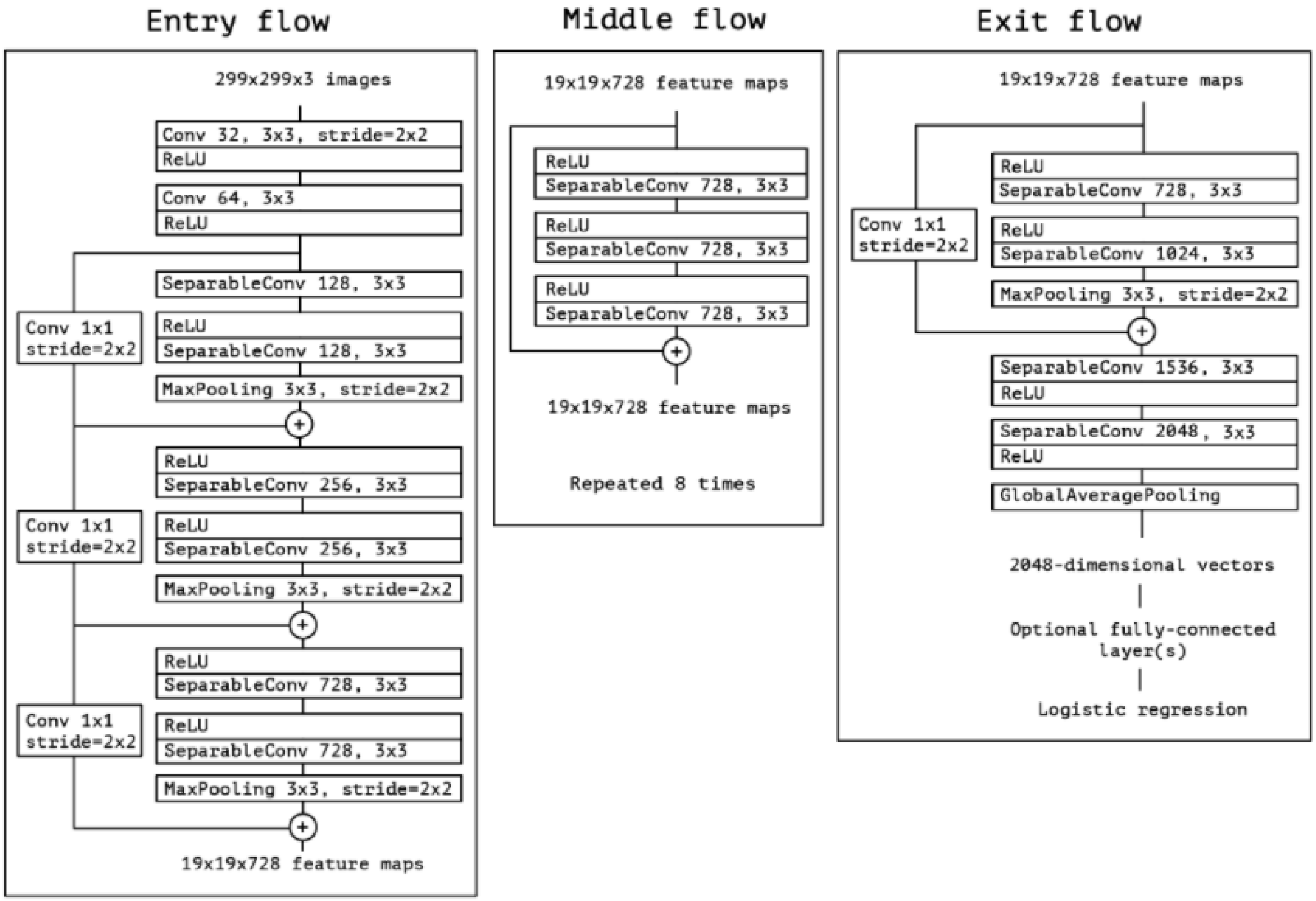
This model comes under IMAGENET project which is largest visual object detection project till date.

Input -> Image of size 299 x 299 with 3 channels namely RGB

Output -> 2048 dimensional vector.

As CNN models are very good at image processing thus using Xception model we will extract out feature vectors for each image in our dataset.

Following is the overview of Xception model architecture:



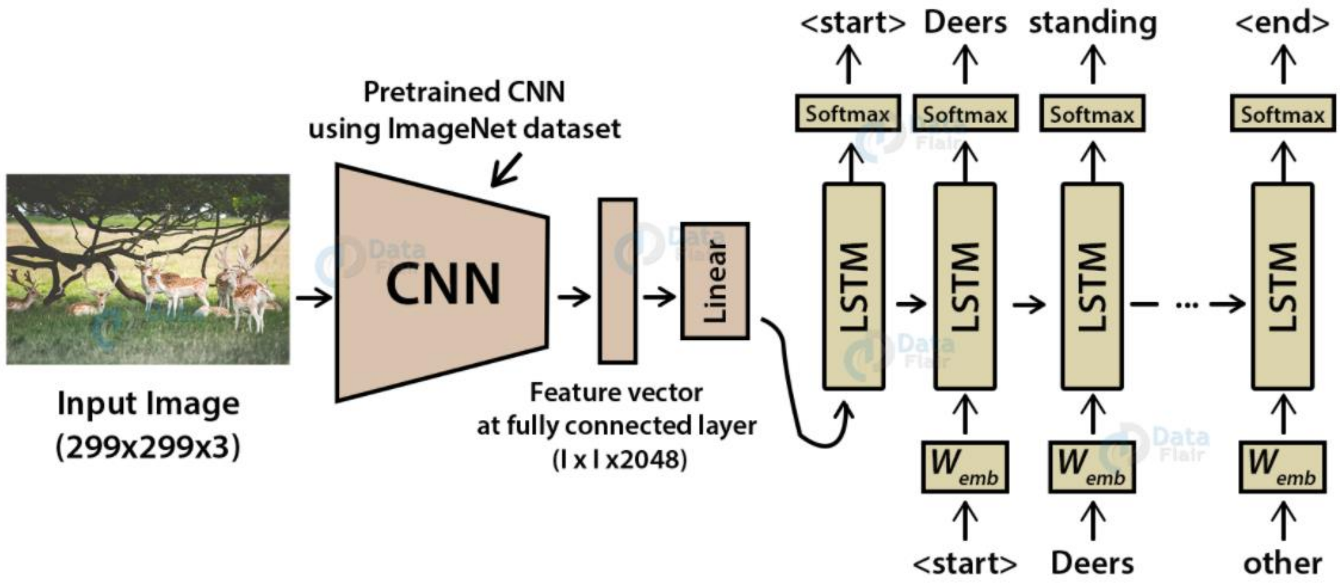
Overall Architecture of Xception (Entry Flow > Middle Flow > Exit Flow)

After generating features I have saved them in a pickle file named features.pkl for later use.

Now it’s time to train a LSTM model on training dataset, this model will be using a supervised learning methodology to train. I will be training on 7k images and number of epochs will be 10. Before training we need to load description.txt file in memory and process it to know what is maximum description length available in file so as to make dense input vector.

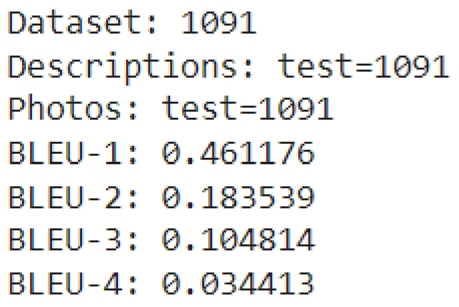
In my case max length is 33, so now I know what is max length of description, I need to pad each smaller description and this can be achieved by pad_sequences() available in `keras.preprocessing.sequence.`

Following is complete proposed architecture:



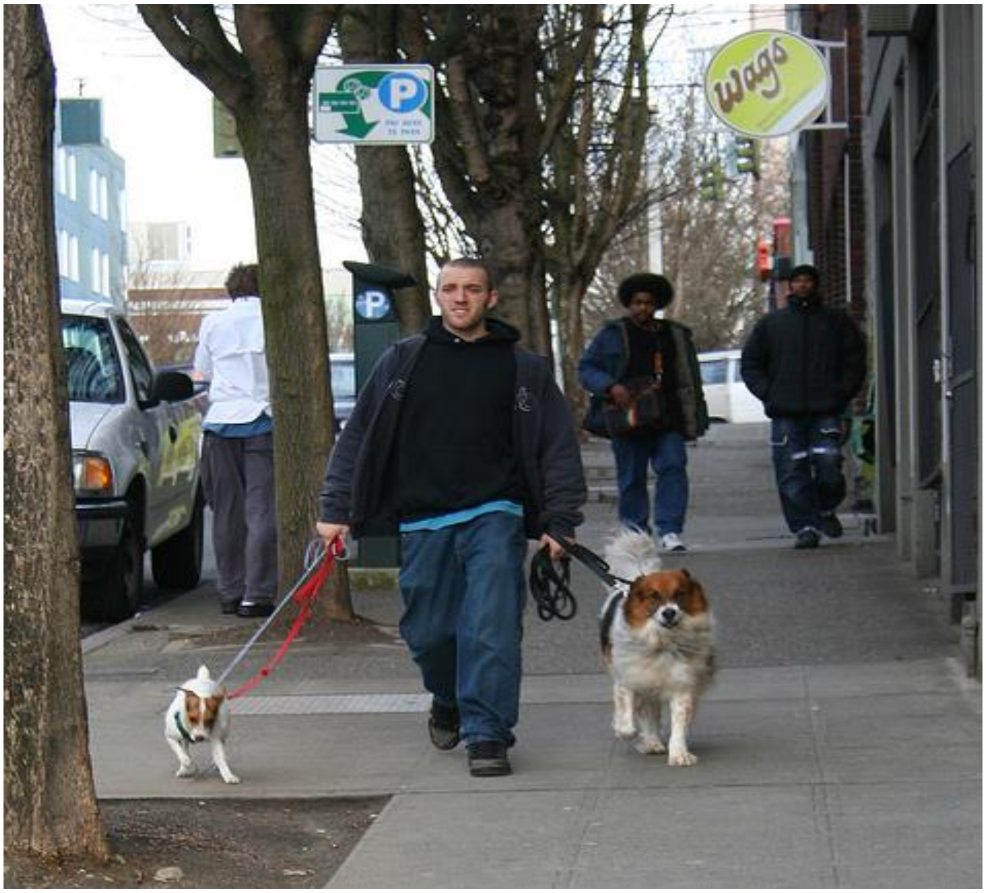
Results:

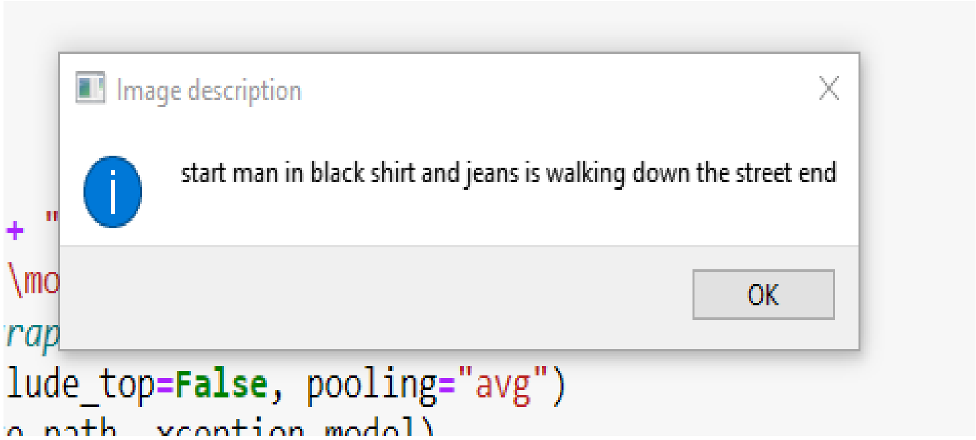
BLEU scores on test set:



```
Dataset: 1091
Descriptions: test=1091
Photos: test=1091
BLEU-1: 0.461176
BLEU-2: 0.183539
BLEU-3: 0.104814
BLEU-4: 0.034413
```

Example:





start man in black shirt and jeans is walking down the street end

```
+ "
\mo
rap
lude_top=False, pooling="avg")
o_path, xception_model)
```

Conclusion:

Overall model perform not bad on test set but performance on general examples shows that more training and data can improve performance by a lot but that comes at expense of high-computing devices available for long hours which is hard to manage.