

PROJECT REPORT

(Project Term August-November 2021)

IMAGE CAPTION GENERATOR

Submitted by

Name of Student: Saksham Soni

Registration Number: 11913397

Course Code: INT 246

Under the Guidance of

(Dr. Sagar Pande)

School of Computer Science and Engineering




Declaration

I hereby declare that the project work entitled (“Image Caption Generator”) is an authentic record of our own work carried out as requirements of Project for the award of B.Tech degree in Computer Science and Engineering from Lovely Professional University, Phagwara, under the guidance of (Dr. Sagar Pande), during August to November 2021. All the information furnished in this project report is based on our own intensive work and is genuine.

Name of the Student: Saksham Soni

Registration Number: 11913397

A handwritten signature in blue ink that reads "Saksham Soni" followed by a period.

Date: 05-11-21

CERTIFICATE

This is to certify that the declaration statement made by this student is correct to the best of my knowledge and belief. They have completed this Project under my guidance and supervision. The present work is the result of their original investigation, effort, and study. No part of the work has ever been submitted for any other degree at any University. The Project is fit for the submission and partial fulfillment of the conditions for the award of B. Tech degree in Computer Science and Engineering from Lovely Professional University, Phagwara.

Name of the Mentor: Dr. Sagar Pande

Designation

School of Computer Science and Engineering,
Lovely Professional University,
Phagwara, Punjab.

ACKNOWLEDGEMENT

I would like to express my gratitude towards my University , my Mentor(Dr. Sagar Pande) and Udeemy for providing me guidance for this Project and Machine Learning, which also helped me in doing a lot of homework and learning. As a result, I came to know about so many new things. So, I am really thank full to them.

Moreover I would like to thank my friends who helped me a lot whenever I got stuck in some problem related to my course. I am really thankful to have such a good support of them as they always have my back whenever I need.

Also,I would like to mention the support system and consideration of my parents who have always been there in my life to make me choose right thing and oppose the wrong. Without them I could never had learned and became a person who I am now.

I have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. I would like to extend my sincere thanks to all of them

TABLE OF CONTENTS

Title Page.....	(i)
Declaration.....	(ii)
Certificate.....	(iii)
Acknowledgement.....	(iv)
Table of Contents.....	(v)

1. ABSTRACT	6
2. INTRODUCTION OF THE PROJECT UNDERTAKEN	7-8
• Objective	
• Scope	
• Use Case	
3. APPLICATIONS	11
4. ADVANTAGES AND DISADVANTAGES	12
5. MOTIVATION	13
6. PROJECT WORKING	14-32
7. RESULTS	33
8. VARIOUS CONCEPTS AND FRAMEWORKS USED	34-40
• Numpy	
• Pandas	
• Supervised Learning	
• Neural Networks	
• Deep Learning	
• CNN	
• RNN	
• Transfer Learning	
• Word Embeddings	
9. ABOUT DATASET AND SOURCE CODE	41
10. CONCLUSION	42
11. FUTURE WORK	43
12. REFERENCES	44

ABSTRACT

In the past few years, the problem of generating descriptive sentences automatically for images has garnered a rising interest in natural language processing and computer vision research. Image captioning is a fundamental task which requires semantic understanding of images and the ability of generating description sentences with proper and correct structure. In this study, the authors propose a hybrid system employing the use of multilayer Convolutional Neural Network (CNN) to generate vocabulary describing the images and a Long Short-Term Memory (LSTM) to accurately structure meaningful sentences using the generated keywords. The convolutional neural network compares the target image to a large dataset of training images, then generates an accurate description using the trained captions. We showcase the efficiency of our proposed model using the Flickr8K and Flickr30K datasets and show that their model gives superior results compared with the state-of-the-art models utilizing the Bleu metric. The Bleu metric is an algorithm for evaluating the performance of a machine translation system by grading the quality of text translated from one natural language to another. The performance of the proposed model is evaluated using standard evaluation matrices, which outperform previous benchmark models.

INTRODUCTION OF THE PROJECT UNDERTAKEN

Objectives of the project

Machine Learning is a field of technology developing with immense abilities and applications in automating tasks, where neither human intervention is needed nor explicit programming.

The power of ML is such great that we can see its applications trending almost everywhere in our day-to-day lives. ML has solved many problems that existed earlier and have made businesses in the world progress to a great extent. Automatically generating captions to an image shows the understanding of the image by computers, which is a fundamental task of intelligence. For a caption model it not only need to find which objects are contained in the image and also need to be able to be expressing their relationships in a natural language such as English. Recently work also achieve the presence of attention, which can store and report the information and relationship between some most salient features and clusters in the image.

Image captioning can be regarded as an end-to-end Sequence to Sequence problem, as it converts images, which is regarded as a sequence of pixels to a sequence of words. For this purpose, we need to process both the language or statements and the images. For the Language part, we use recurrent Neural Networks and for the Image part, we use Convolutional Neural Networks to obtain the feature vectors respectively.



"man in black shirt is playing guitar."



"construction worker in orange safety vest is working on road."



"two young girls are playing with lego toy."

Citation: <https://bit.ly/3pvIUvU>

Scope of the project

Image Captioning is the process of generating a textual description for given images. It has been a very important and fundamental task in the Deep Learning domain. Image captioning has a huge amount of application. NVIDIA is using image captioning technologies to create an application to help people who have low or no eyesight.

In our project, we do image-to-sentence generation. This application bridges vision and natural language. If we can do well in this task, we can then utilize natural language processing technologies understand the world in images. In addition, we introduced attention mechanism, which is able to recognize what a word refers to in the image, and thus summarize the relationship between objects in the image. This will be a powerful tool to utilize the massive unformatted image data, which dominate the whole data in the world.

Use Cases

- Some detailed use cases would be like a visually impaired person taking a picture from his phone and then the caption generator will turn the caption to speech for him to understand.
- Advertising industry trying the generate captions automatically without the need to make them separately during production and sales.
- Doctors can use this technology to find tumors or some defects in the images or used by people for understanding geospatial images where they can find out more details about the terrain

HOW IMAGE CAPTIONING WORKS



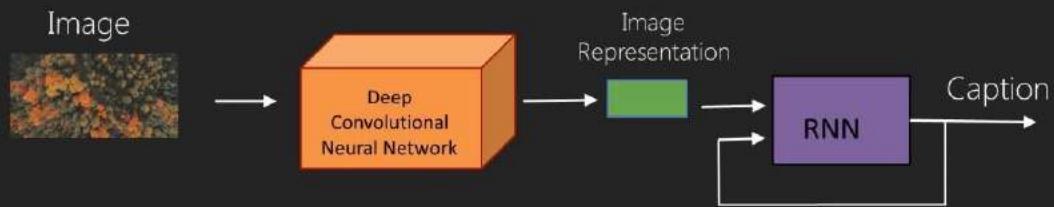
Citation: https://miro.medium.com/max/12000/1*fo5IYK6wEkCTh10TdtZNSA.jpeg

If we are told to describe it, maybe we will describe it as: “A puppy on a blue towel” or “ A brown dog playing with a green ball”. So, how are we doing this? While forming the description, we are seeing the image but at the same time, we are looking to create a meaningful sequence of words. The first part is handled by CNNs and the second is handled by RNNs.

The task of image captioning can be divided into two modules logically – one is an **image based model** – which extracts the features and nuances out of our image, and the other is a **language based model** – which translates the features and objects given by our image based model to a natural sentence.

For our image-based model (viz encoder) – we usually rely on a Convolutional Neural Network model. And for our language-based model (viz decoder) – we rely on a Recurrent Neural Network. The image below summarizes the approach given above

Automatic Image Caption



Citation: <https://bit.ly/31ri7se>

Usually, a pretrained CNN extracts the features from our input image. The feature vector is linearly transformed to have the same dimension as the input dimension of the RNN/LSTM network. This network is trained as a language model on our feature vector. Convolutional Neural Network (CNN) to generate vocabulary describing the images and a Long Short-Term Memory (LSTM) to accurately structure meaningful sentences using the generated keywords.

APPLICATIONS

- **VISUAL AID FOR BLIND PEOPLE:** We can build a technique in which we can fix a camera over the head of blind people which captures images of the surroundings and builds captions for the respective captured images and then we can convert that caption into some form of voice, which will ultimately help that blind people to walk around.
- **GOOGLE IMAGE SEARCHING:** As we have google lens these days.
- **AUTOMATIC SURVEILLIANCE (CCTV'S CAMERA):** We can make some alert system by generating captions from the images when suspicious things will go around.
- **SELF DRIVING**

ADAVANTAGES & DISADVANTAGES

Advantages:

1) Recommendations in Editing Applications

The image captioning model automates and accelerates the close captioning process for digital content production, editing, delivery, and archival. Well-trained models replace manual efforts for generating quality captions for images as well as videos.

2) Assistance for Visually Impaired

The advent of machine learning solutions like image captioning is a boon for visually impaired people who are unable to comprehend visuals. With AI-powered image caption generator, image descriptions can be read out to visually impaired, enabling them to get a better sense of their surroundings.

3) Media and Publishing Houses

The media and public relations industry circulate tens of thousands of visual data across borders in the form of newsletters, emails, etc. The image captioning model accelerates subtitle creation and enables executives to focus on more important tasks.

4) Social Media Posts

For social media, artificial intelligence is moving from discussion rooms to underlying mechanisms for identifying and describing terabytes of media files. It enables community administrators to monitor interactions and analysts to formulate business strategies.

Disadvantages:

My model has lots of improvisations and limitations. First of all, training and testing of model has been performed on related images. So, if we provide a very different image for testing purpose then the results are not very much satisfactory, this can be improved by training the model on bigger dataset like flickr30k instead of flickr8k.

MOTIVATION

My motivation behind building this project is based on a project idea that I got few months back.

The goal is to improve the road safety by measuring the traffic on road, speed limits, news articles, driver's posture or motion to detect the alertness which helps us to identify the road safety through machine learning and deep learning.

- Analyzing road accidents data from public databases, newspapers, web pages, etc.
- Build an AI-Based method based on Machine Learning algorithms & Deep Learning to detect traffic accidents in real-time with the use of traffic cameras.
- Build a Deep Learning-based solution for the driver monitoring system by studying a person's posture and body movements, intelligent interior vehicle algorithms can draw conclusions about a person's alertness, attention and focus.
- A web application to predict the probability of an accident from video and actions of the driver.

PROJECT WORKING

STEPS TO BUILD OUR MODEL:

- Data collection
- Understanding the data
- Data Cleaning
- Loading the training set
- Data Preprocessing — Images
- Data Preprocessing — Captions
- Data Preparation using Generator Function
- Word Embeddings
- Model Architecture

STEP-1 : DATA COLLECTION AND UNDERSTANDING OUR DATA

One of the files is “Flickr8k.token.txt” which contains the name of each image along with its 5 captions. (`<image name>#i <caption>`, where $0 \leq i \leq 4$)

```
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 street .
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 .
1002674143_1b742ab4b8.jpg#0 A little girl covered in paint sits in front of a painted rainbow with her hands in a bowl
.
1002674143_1b742ab4b8.jpg#1 A little girl is sitting in front of a large painted rainbow .
1002674143_1b742ab4b8.jpg#2 A small girl in the grass plays with fingerpaints in front of a white canvas with a rainbow
on it .
1002674143_1b742ab4b8.jpg#3 There is a girl with pigtails sitting in front of a rainbow painting .
1002674143_1b742ab4b8.jpg#4 Young girl with pigtails painting outside in the grass .
1003163366_44323f5815.jpg#0 A man lays on a bench while his dog sits by him .
1003163366_44323f5815.jpg#1 A man lays on the bench to which a white dog is also tied .
```

Citation: Screenshot of file in dataset

We will create a dictionary named “descriptions” which contains the name of the image (without the .jpg extension) as keys and a list of the 5 captions for the corresponding image as values.

```
In [7]: descriptions["1000268201_693b08cb0e"]

Out[7]: ['A child in a pink dress is climbing up a set of stairs in an entry way .',
'A girl going into a wooden building .',
'A little girl climbing into a wooden playhouse .',
'A little girl climbing the stairs to her playhouse .',
'A little girl in a pink dress going into a wooden cabin .']
```

Citation: Screenshot of my own code on google colab

STEP-2: DATA PREPROCESSING

- 1- Data Cleaning
- 2- Vocabulary
- 3- Prepare Train/Test Data
- 4- Transfer Learning
 - (i) Image Feature Extraction
 - (ii) Text Preprocessing for captions
- 5- Data Preparation using Generator Function
- 6- Word Embeddings

1. DATA CLEANING

all texts-> lower()

Remove numbers, punctuations and any non (a-z) symbol

Now our dictionary (description) will have values as :

```
In [12]: descriptions["1000268201_693b08cb0e"]  
  
Out[12]: ['child in pink dress is climbing up set of stairs in an entry way',  
          'girl going into wooden building',  
          'little girl climbing into wooden playhouse',  
          'little girl climbing the stairs to her playhouse',  
          'little girl in pink dress going into wooden cabin']
```

Now we will save this information(dictionary) in a file.

2. VOCABLURY

We will create a vocab of all the unique words present across all the 8000*5 (i.e. 40000) image captions (corpus) in the data set using set() data structure.

Vocab Size: 8424 (This means we have 8424 unique words across all the 40000 image captions.)

Total Words: 373837

Now, we will filter out the words according to certain threshold frequency. We consider only those words which occur at least 10 times in the entire corpus.

Now, Final Vocab Size: 1845 (So now we have only 1845 unique words in our vocabulary)

3. PREPARE TRAINING DATASET

The text file “Flickr_8k.trainImages.txt” contains the names of the images that belong to the training set. Same applies for “Flickr_8k.testImages.txt”

```
2513260012_03d33305cf.jpg
2903617548_d3e38d7f88.jpg
3338291921_fe7ae0c8f8.jpg
488416045_1c6d903fe0.jpg
2644326817_8f45080b87.jpg
218342358_1755a9cce1.jpg
2501968935_02f2cd8079.jpg
2699342860_5288e203ea.jpg
2638369467_8fc251595b.jpg
2926786902_815a99a154.jpg
2851304910_b5721199bc.jpg
3423802527_94bd2b23b0.jpg
3356369156_074750c6cc.jpg
2294598473_40637b5c04.jpg
1191338263_a4fa073154.jpg
2380765956_6313d8cae3.jpg
```

Now, we load the descriptions of these images from “descriptions.txt” (saved on the hard disk) in dictionary “train_descriptions”

However, when we load them, we will add two tokens in every caption as follows:

- ‘**startseq**’ (<s>) -> This is a start sequence token which will be added at the start of every caption.
- ‘**endseq**’ (<e>)-> This is an end sequence token which will be added at the end of every caption.

```
In [34]: train_descriptions["1000268201_693b08cb0e"]

Out[34]: ['startseq child in pink dress is climbing up set of stairs in an entry way endseq',
'startseq girl going into wooden building endseq',
'startseq little girl climbing into wooden playhouse endseq',
'startseq little girl climbing the stairs to her playhouse endseq',
'startseq little girl in pink dress going into wooden cabin endseq']
```

4. TRANSFER LEARNING

STEP 1: Image feature extraction/ Image Preprocessing:

We need to convert every image into a fixed sized vector which can then be fed as input to the neural network. For this purpose, we opt for transfer learning by using a pre-trained model that is, ResNet-50 model trained on more than a million images from the ImageNet database.

What is ResNet-50 model ?

- ResNet, short for Residual Networks is a classic neural network used as a backbone for many computer vision tasks. This model was the winner of ImageNet challenge in 2015
- It is a Convolutional Neural Network that is 50 layers deep
- It has skip connections to avoid vanishing gradient problem

Now Let's see our ResNet-50 model summary after giving input image with shape as (224 X 224 X3)

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/resnet/resnet50_weights_tf_dim_ordering_tf_kernels.h5
102973440/102967424 [=====] - 2s 0us/step
102981632/102967424 [=====] - 2s 0us/step
Model: "resnet50"
```

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	(None, 224, 224, 3)	0	input_1[0][0]
conv1_pad (ZeroPadding2D)	(None, 230, 230, 3)	0	input_1[0][0]
conv1_conv (Conv2D)	(None, 112, 112, 64)	9472	conv1_pad[0][0]
conv1_bn (BatchNormalization)	(None, 112, 112, 64)	256	conv1_conv[0][0]
conv1_relu (Activation)	(None, 112, 112, 64)	0	conv1_bn[0][0]
pool1_pad (ZeroPadding2D)	(None, 114, 114, 64)	0	conv1_relu[0][0]
pool1_pool (MaxPooling2D)	(None, 56, 56, 64)	0	pool1_pad[0][0]
conv2_block1_1_conv (Conv2D)	(None, 56, 56, 64)	4160	pool1_pool[0][0]
conv2_block1_1_bn (BatchNormalization)	(None, 56, 56, 64)	256	conv2_block1_1_conv[0][0]
conv2_block1_1_relu (Activation)	(None, 56, 56, 64)	0	conv2_block1_1_bn[0][0]
conv2_block1_2_conv (Conv2D)	(None, 56, 56, 64)	36928	conv2_block1_1_relu[0][0]
conv2_block1_2_bn (BatchNormalization)	(None, 56, 56, 64)	256	conv2_block1_2_conv[0][0]

```

conv5_block3_add (Add)      (None, 7, 7, 2048) 0      ['conv5_block2_out[0][0]',
                                                    'conv5_block3_bn[0][0]']

conv5_block3_out (Activation) (None, 7, 7, 2048) 0      ['conv5_block3_add[0][0]']

avg_pool (GlobalAveragePooling (None, 2048) 0      ['conv5_block3_out[0][0]']
2D)

```

- For every image we will extract a 2048 length vector in which we will have 2048 list of numbers which will tell us about the features present in a particular image.
- So, this 2048 values is a feature representation for that given image.
- And we will save the features in a file.

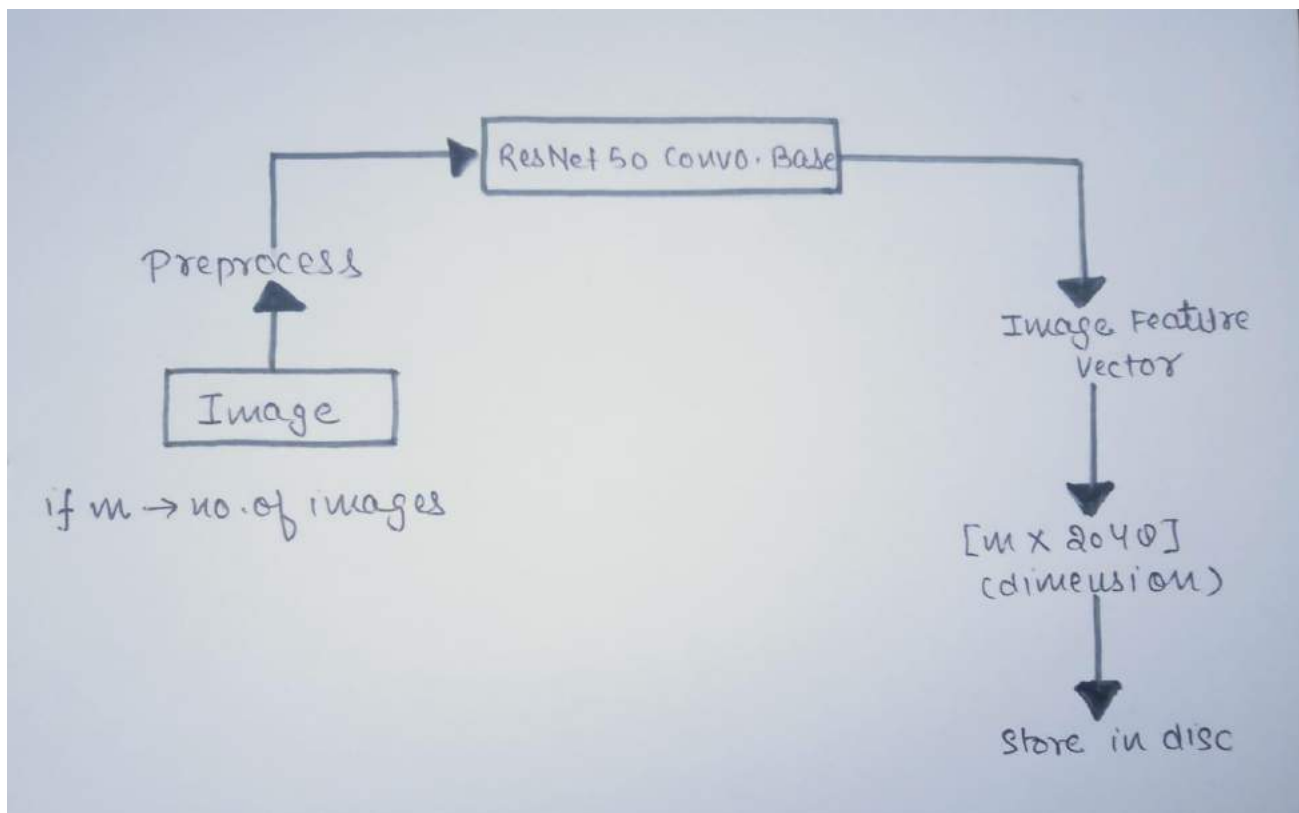


Image Feature Vector  Image Captioning Model

Image Preprocess:



Before Preprocessing



After Preprocessing

Image Encoding:

```
encode_image(IMG_PATH+"1000268201_693b08cb0e.jpg")  
  
(2048,)   
array([0.06535938, 0.16782549, 0.3251763 , ..., 0.05107132, 0.32821193,  
       1.084337 ], dtype=float32)
```

We will follow this step for all the images by adding some time stamps :

```
Encoding in Progress Time step 3800  
Encoding in Progress Time step 3900  
Encoding in Progress Time step 4000  
Encoding in Progress Time step 4100  
Encoding in Progress Time step 4200  
Encoding in Progress Time step 4300  
Encoding in Progress Time step 4400  
Encoding in Progress Time step 4500  
Encoding in Progress Time step 4600  
Encoding in Progress Time step 4700  
Encoding in Progress Time step 4800  
Encoding in Progress Time step 4900  
Encoding in Progress Time step 5000  
Encoding in Progress Time step 5100  
Encoding in Progress Time step 5200  
Encoding in Progress Time step 5300  
Encoding in Progress Time step 5400  
Encoding in Progress Time step 5500  
Encoding in Progress Time step 5600  
Encoding in Progress Time step 5700  
Encoding in Progress Time step 5800  
Encoding in Progress Time step 5900  
Total Time Taken : 3839.9682354927063
```

We save all the bottleneck train features in a Python dictionary and save it on the disk using Pickle file, namely “**encoded_train_features2.pkl**” whose keys are image names and values are corresponding 2048 length feature vector. We will do all this for test set images.

STEP 2: Text Preprocessing/ Data preprocessing for captions:

**** CAPTIONS(ENG SENTENCES) -> NUMERIC VALUES ****

NOTE: The prediction of the entire caption, given the image does not happen at once. We will predict the caption **word by word**

- We will make two dictionaries->

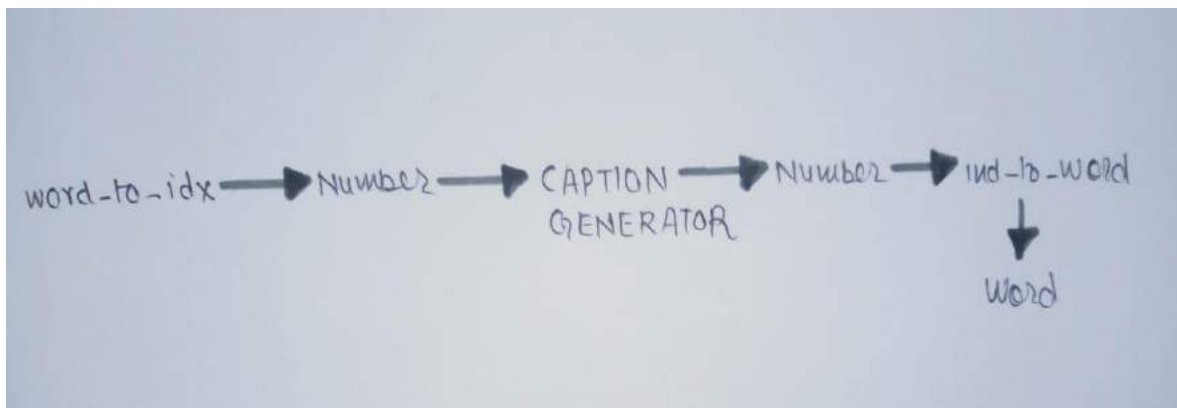
1- word_to_idx : It will have words(we had shortlisted earlier) as keys and a unique number associated to that word as value.

2- idx_to_word: it will have numbers as keys and words as values.

- We will also do computations for <s> and <e> in both of the dictionaries.

FINAL VOCAB SIZE= 1848

- We will also find length of largest caption (35) to determine batch size



5. DATA PREPARATION USING GENERATOR FUNCTION

Now we will understand how to prepare the data in a manner which will be convenient to be given as input to the deep learning model.

		X_i	PAGE NO: Y_i
i	Image feature vector	Partial Caption	Target word
1	img-1	startseq	the
2	img-1	startseq the	black
3	img-1	startseq the black	cat
4	img-1	startseq the black cat	sat
5	img-1	startseq the black cat sat	on
6	img-1	startseq the black cat sat on	grass
7	img-1	startseq the black cat sat on grass endseq	

→ Data points corresponding to one image & its caption

- It must be noted that, one image+caption is **not a single data point** but are multiple data points depending on the length of the caption.
- Similarly, if we consider both the images and their captions, our data matrix will then look as follows:

	X_i	Y_i	
1	Image Feature vector	Partial Caption	
		Target word	
1	img-1	startseq	the
2	img-1	" the	black
3	img-1	" " black	cat
4	img-1	" " " eat	sat
5	img-1	" " " " sat	on
6	img-1	" " " " " on	grass
7	img-1	" " " " " " grass	endseq
8	img-2	startseq	the
9	img-2	" the	white
10	img-2	" " white	cat
11	img-2	" " " cat	is
12	img-2	" " " " is	walking
13	img-2	" " " " " walking	on
14	img-2	" " " " " " on	road
15	img-2	" " " " " " " road	endseq

data points correspond to img-1 & its caption

data points correspond to img-2 & its caption

data points correspond to img-1 & its caption

data points correspond to img-2 & its caption

- Since we are processing **sequences**, we will employ a **Recurrent Neural Network** to read these partial captions
- We are not going to pass the actual English text of the caption, rather we are going to pass the sequence of indices where each index represents a unique word.

i	Image Feature vector	X_i	Y_i
		Partial Caption	Target Word
1	img-1	[9]	10
2	img-1	[9, 10]	1
3	img-1	[9, 10, 1]	2
4	img-1	[9, 10, 1, 2]	0
5	img-1	[9, 10, 1, 2, 0]	6
6	img-1	[9, 10, 1, 2, 0, 6]	4
7	img-1	[9, 10, 1, 2, 0, 6, 4]	3
8	img-2	[9]	10
9	img-2	[9, 10]	12
10	img-2	[9, 10, 12]	2
11	img-2	[9, 10, 12, 2]	5
12	img-2	[9, 10, 12, 2, 5]	11
13	img-2	[9, 10, 12, 2, 5, 11]	6
14	img-2	[9, 10, 12, 2, 5, 11, 6]	7
15	img-2	[9, 10, 12, 2, 5, 11, 6, 7]	3

Since we would be doing **batch processing** (explained later), we need to make sure that each sequence is of **equal length**. Hence we need to **append 0's** (zero padding) at the end of each sequence. So, we will append those many number of zeros which will lead to every sequence having a length of 35, which is maximum length of the caption in the entire dataset.

i	Image Feature vector	X_i	Y_i
		Partial caption	Target word
1	img-1	[9, 0, 0, ..., 0]	10
2	img-1	[9, 10, 0, 0, ..., 0]	1
3	img-1	[9, 10, 1, 0, 0, ..., 0]	2
4	img-1	[9, 10, 1, 2, 0, 0, ..., 0]	8
5	img-1	[9, 10, 1, 2, 0, 0, 0, ..., 0]	6
6	img-1	[9, 10, 1, 2, 0, 6, 0, 0, ..., 0]	4
7	img-1	[9, 10, 1, 2, 0, 6, 4, 0, 0, ..., 0]	3
8	img-2	[9, 0, 0, ..., 0]	10
9	img-2	[9, 10, 0, 0, ..., 0]	12
10	img-2	[9, 10, 12, 0, 0, ..., 0]	2
11	img-2	[9, 10, 12, 2, 0, 0, ..., 0]	5
12	img-2	[9, 10, 12, 2, 5, 0, 0, ..., 0]	11
13	img-2	[9, 10, 12, 2, 5, 11, 0, 0, ..., 0]	6
14	img-2	[9, 10, 12, 2, 5, 11, 6, 0, 0, ..., 0]	7
15	img-2	[9, 10, 12, 2, 5, 11, 6, 7, 0, 0, ..., 0]	3

NEED OF THE DATA GENERATOR:

- In the above example, I have only considered 2 images and captions which lead to 15 data points.
- However, in our actual training dataset we have 6000 images, each having 5 captions. This makes a total of **30000** images and captions.
- Even if we assume that each caption on an average is just 7 words long, it will lead to a total of 30000×7 .e. **210000** data points.

Computing the size of the data matrix:

Size of the data matrix = $n * m$

\downarrow \rightarrow
 no. of data points length of each data point
 (assumed as 210000)

clearly, $m = \text{length of img vector (2040)} + \text{length of partial caption (x)}$

$m = 2040 + x$

$x = ?$

$\rightarrow 35? (X)$

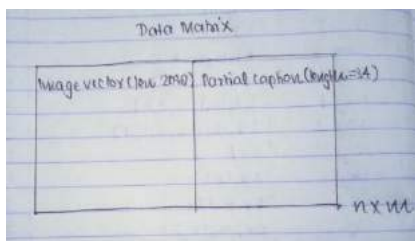
- * Every word (or index) will be mapped (embedded) to higher dimensional space through one of the word embedding techniques.
- * later, we will see that each word/index is mapped to a ~~20~~⁵⁰-long vector using a pre-trained GLOVE word embedding model.

Now, each sequence contains 35 indices, where each index is a vector of length ~~20~~⁵⁰.

$\therefore x = 35 \times 50 = 1750$

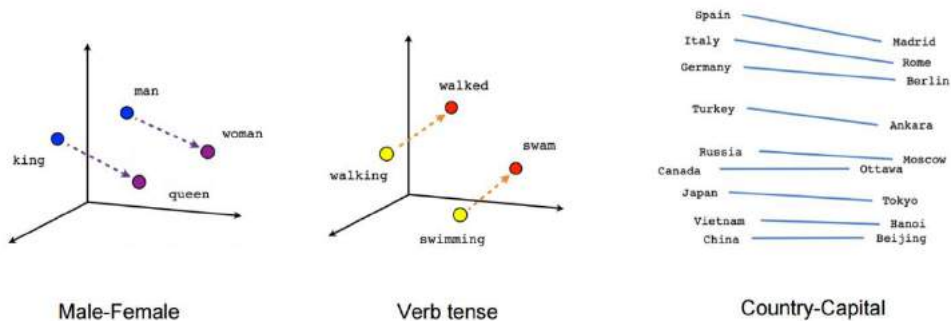
$m = 2040 + 1750 = 3790$

Finally, size of data matrix = $210000 * 3790$
 $= 797,500,000 \text{ blocks}$



6. WORD EMBEDDINGS

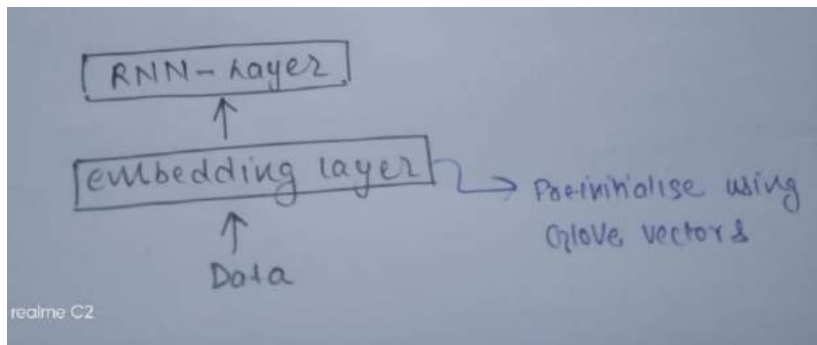
- Word representation that allows words with similar meanings to have a similar representation.
- We map words to real valued vectors in a pre-defined vector space.
- Word embeddings are created using a neural network with one input layer, one hidden layer and one output layer.



Citation: <https://bit.ly/31tDzwR>

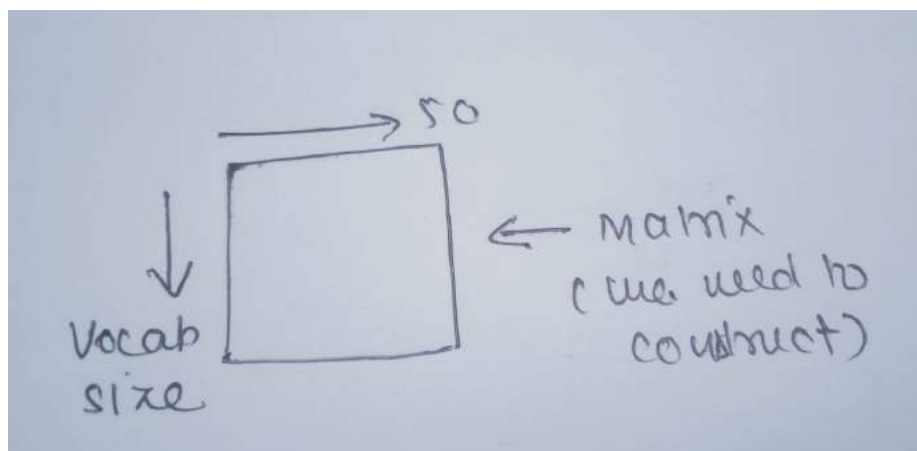
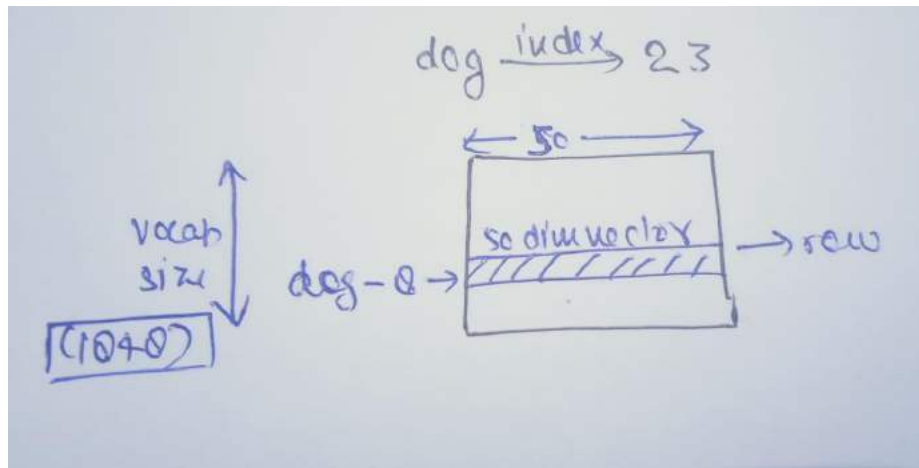
GloVe Embedding:

- GloVe stands for global vectors. GloVe is an unsupervised learning algorithm for obtaining vector representation for words.
- It was developed by Stanford for generating word embeddings.



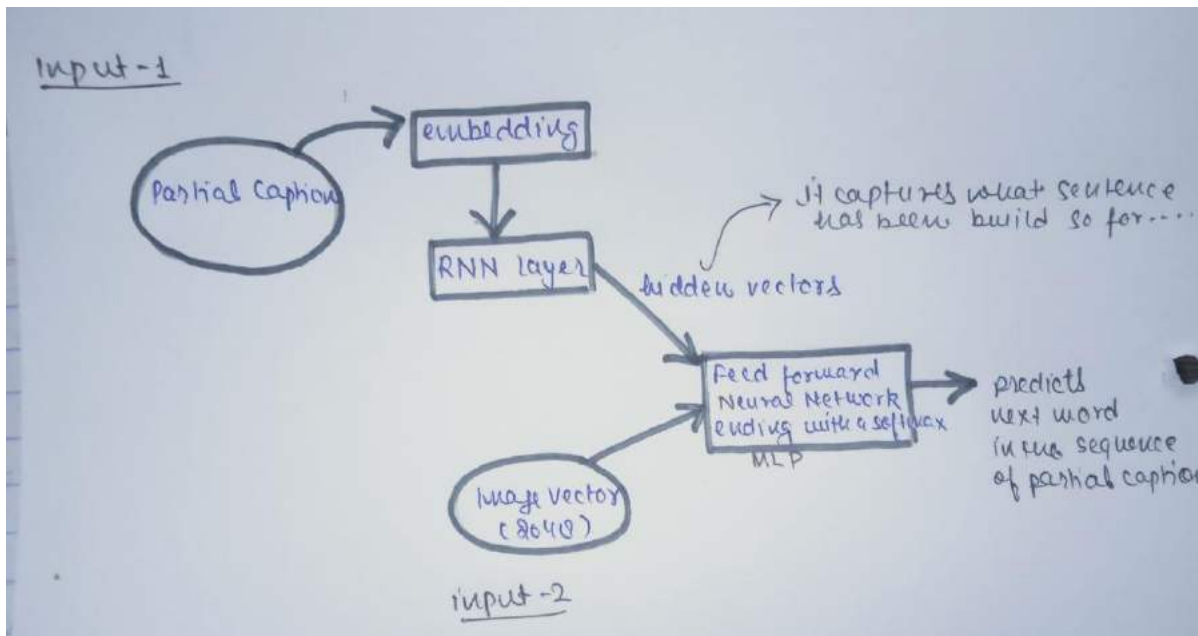
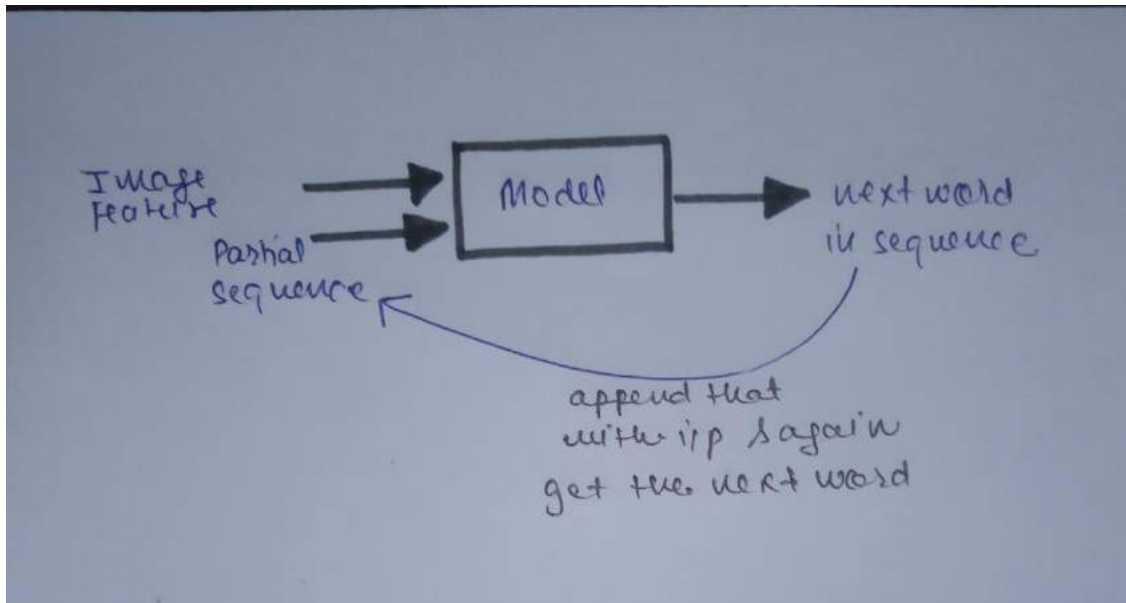
- Use GloVe6B50D.txt file which contains glove vectors of 6 Billion words and each word as 50 dimensional vector.

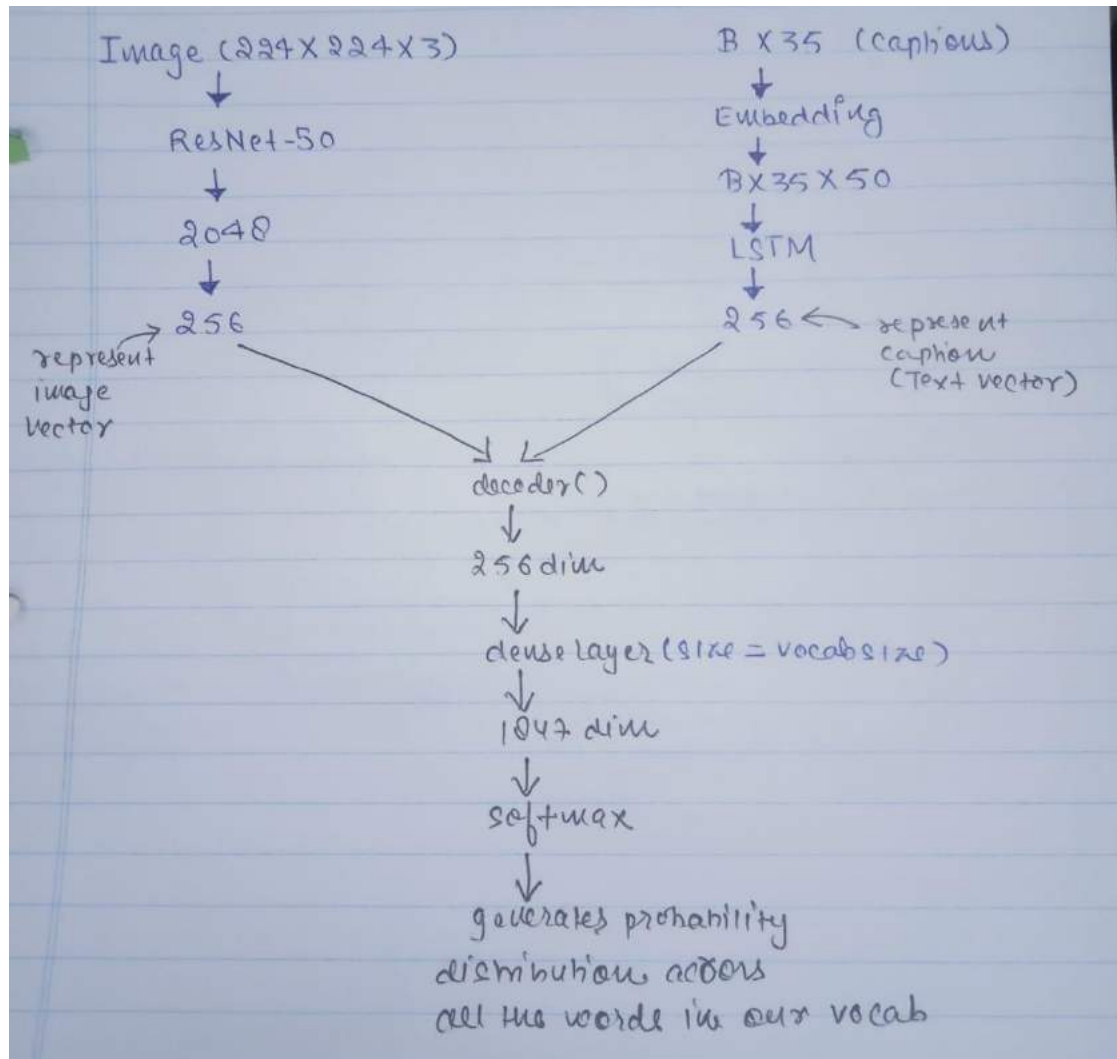
- We will create embedding_matrix.



MODEL ARCHITECTURE

On a high level->





- Finally, the weights of the model will be updated through backpropagation algorithm and the model will learn to output a word, given an image feature vector and a partial caption. So in summary, we have:
- Input_1 -> Partial Caption
- Input_2 -> Image feature vector
- Output -> An appropriate word, next in the sequence of partial caption provided in the input_1 (or in probability terms we say conditioned on image vector and the partial caption)

RESULTS

Here are some of the predictions on the testing images

mountain climber is standing on top of mountain



basketball player in white is jumping into the air



surfer rides wave



jockey is running on the grass



man in black coat and black hat is standing in front of car



VARIOUS CONCEPTS AND FRAMEWORKS USED

NUMPY

The NumPy array is a data structure that efficiently stores and accesses multidimensional arrays (also known as tensors) and enables a wide variety of scientific computation. It consists of a pointer to memory, along with metadata used to interpret the data stored there, notably 'data type', 'shape' and 'strides'

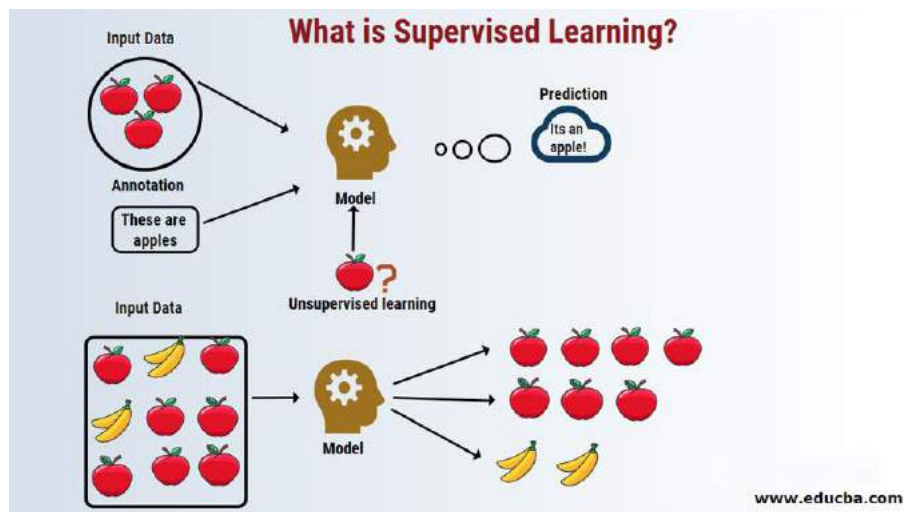
PANDAS

Pandas is a Python library used for working with data sets. It has functions for analyzing, cleaning, exploring, and manipulating data. The name "Pandas" has a reference to both "Panel Data", and "Python Data Analysis" and was created by Wes McKinney in 2008. Pandas allows us to analyze big data and make conclusions based on statistical theories.

Pandas can clean messy data sets, and make them readable and relevant. Relevant data is very important in data science.

SUPERVISED LEARNING

A training set of examples with the correct responses (targets) is provided and, based on this training set, the algorithm generalizes to respond correctly to all possible inputs. This is also called learning from exemplars. Supervised learning is the machine learning task of learning a function that maps an input to an output based on example input-output pairs. In supervised learning, each example in the training set is a pair consisting of an input object (typically a vector) and an output value. A supervised learning algorithm analyzes the training data and produces a function, which can be used for mapping new examples. In the optimal case, the function will correctly determine the class labels for unseen instances. Both classification and regression problems are supervised learning problems. A wide range of supervised learning algorithms are available, each with its strengths and weaknesses. There is no single learning algorithm that works best on all supervised learning problems



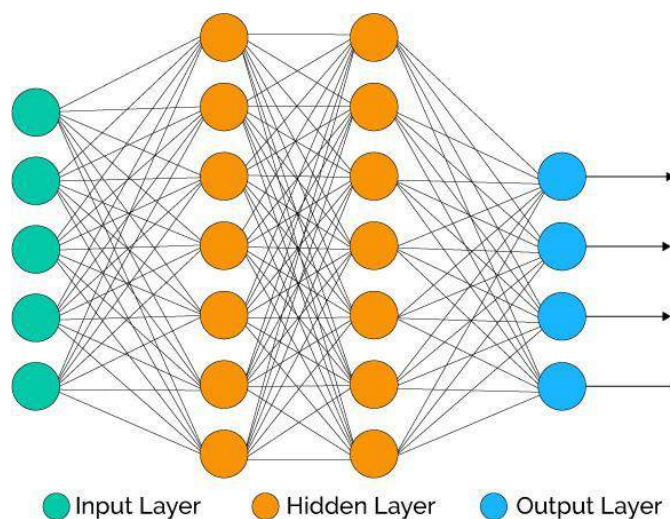
Citation: <https://bit.ly/3DpRzEV>

NEURAL NETWORKS

Neural networks, also known as artificial neural networks (ANNs) or simulated neural networks (SNNs), are a subset of machine learning and are at the heart of deep learning algorithms. Their name and structure are inspired by the human brain, mimicking the way that biological neurons signal to one another.

Artificial neural networks (ANNs) are comprised of a node layer, containing an input layer, one or more hidden layers, and an output layer. Each node, or artificial neuron, connects to another and has an associated weight and threshold. If the output of any individual node is above the specified threshold value, that node is activated, sending data to the next layer of the network. Otherwise, no data is passed along to the next layer of the network.

Neural networks rely on training data to learn and improve their accuracy over time. However, once these learning algorithms are fine-tuned for accuracy, they are powerful tools in computer science and artificial intelligence, allowing us to classify and cluster data at a high velocity. Tasks in speech recognition or image recognition can take minutes versus hours when compared to the manual identification by human experts. One of the most well-known neural networks is Google's search algorithm.



Citation: <https://bit.ly/3okYf2H>

DEEP LEARNING

Deep learning is a subset of machine learning, which is essentially a neural network with three or more layers. These neural networks attempt to simulate the behavior of the human brain—albeit far from matching its ability—allowing it to “learn” from large amounts of data. While a neural network with a single layer can still make approximate predictions, additional hidden layers can help to optimize and refine for accuracy.

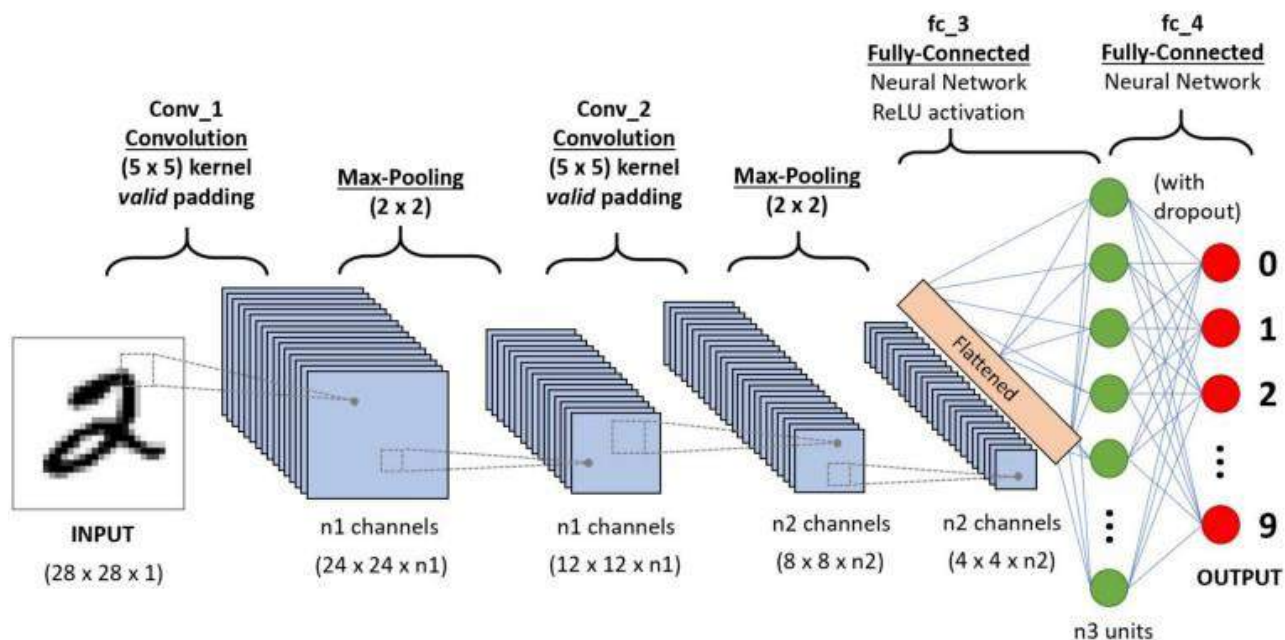
Deep learning drives many artificial intelligence (AI) applications and services that improve automation, performing analytical and physical tasks without human intervention. Deep learning technology lies behind everyday products and services (such as digital assistants, voice-enabled TV remotes, and credit card fraud detection) as well as emerging technologies (such as self-driving cars).

Deep learning neural networks, or artificial neural networks, attempts to mimic the human brain through a combination of data inputs, weights, and bias. These elements work together to accurately recognize, classify, and describe objects within the data.

CONVOLUTIONAL NEURAL NETWORK (CNN)

A **Convolutional Neural Network (CNN)** is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a CNN is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, CNN have the ability to learn these filters/characteristics.

The architecture of a CNN is analogous to that of the connectivity pattern of Neurons in the Human Brain and was inspired by the organization of the Visual Cortex. Individual neurons respond to stimuli only in a restricted region of the visual field known as the Receptive Field. A collection of such fields overlaps to cover the entire visual area.



Citation: <https://bit.ly/3xRwlyH>

RECURRENT NEURAL NETWORK (RNN)

A recurrent neural network (RNN) is a special type of an artificial neural network adapted to work for time series data or data that involves sequences. Ordinary feed forward neural networks are only meant for data points, which are independent of each other. However, if we have data in a sequence such that one data point depends upon the previous data point, we need to modify the neural network to incorporate the dependencies between these data points. RNNs have the concept of 'memory' that helps them store the states or information of previous inputs to generate the next output of the sequence.

RNN is the extension of feedforward NN with the presence of loops in hidden layers. RNN takes the input with the sequence of samples and identifies the time relationship between the samples. The Long short-term memory (LSTM) solves the classification issues by adding the network parameters with the hidden node and releases the state based on the input values. RNN achieves better performance than LSTM by activating the states based on network events. The regular RNN node consists of a single bias and weight. The RNN is evaluated using the gated recurrent unit and LSTM.

TRANSFER LEARNING

Transfer learning is a machine learning method where a model developed for a task is reused as the starting point for a model on a second task.

It is a popular approach in deep learning where pre-trained models are used as the starting point on computer vision and natural language processing tasks given the vast compute and time resources required to develop neural network models on these problems and from the huge jumps in skill that they provide on related problems.

In this post, you will discover how you can use transfer learning to speed up training and improve the performance of your deep learning model

Features of Transfer Learning

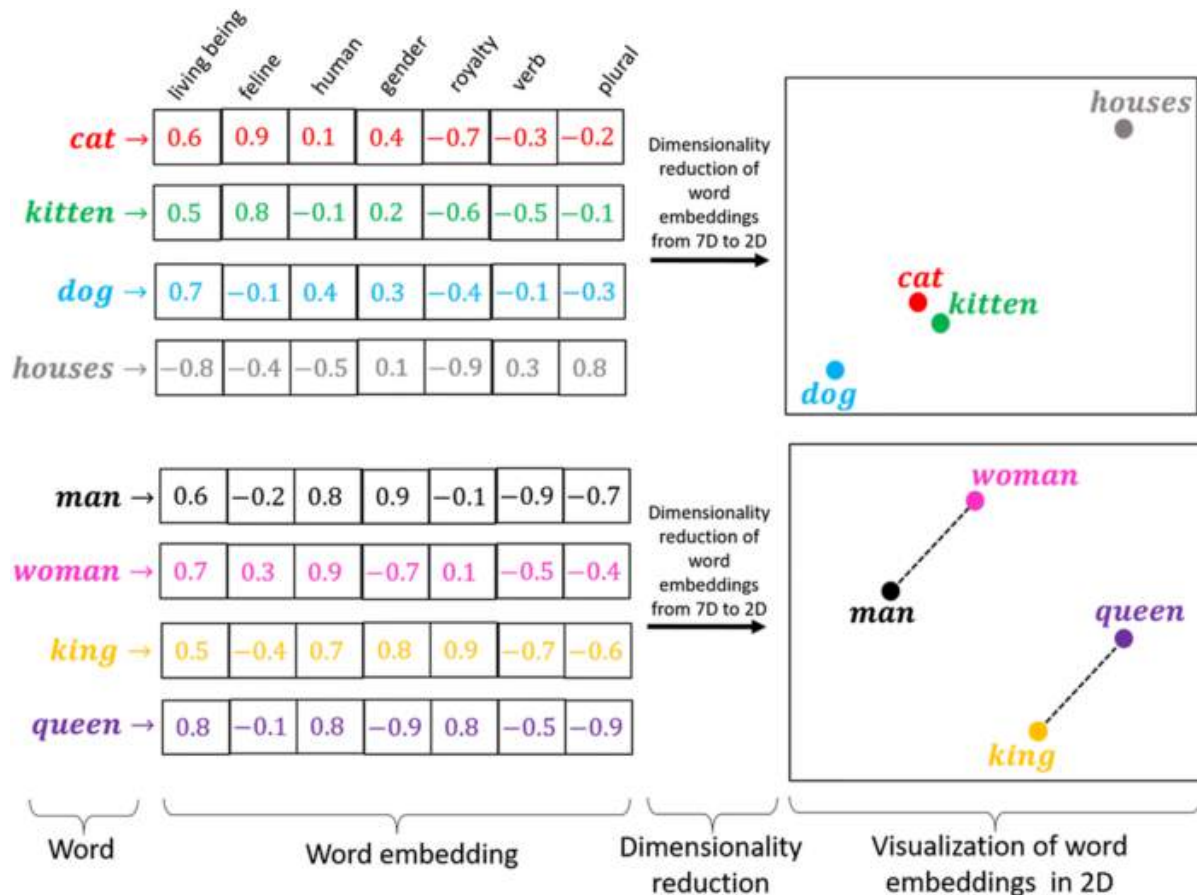
- It involves the transfer of knowledge that is grasped in one source task to learn and refine the related target task.
- It has been observed that DNN trained on the natural images shows a strange occurrence where the first layer of the network appears to learn features alike to Gabor filters.
- These first layers features are found to be general features for many dataset.

WORD EMBEDDINGS

A word embedding is a learned representation for text where words that have the same meaning have a similar representation.

It is this approach to representing words and documents that may be considered one of the key breakthroughs of deep learning on challenging natural language processing problems.

Word embeddings are in fact a class of techniques where individual words are represented as real-valued vectors in a predefined vector space. Each word is mapped to one vector and the vector values are learned in a way that resembles a neural network, and hence the technique is often lumped into the field of deep learning



Citation: <https://bit.ly/3ojbdOC>

ABOUT DATA SET AND SOURCE CODE

I have used flickr8k data set in this project which is available on kaggle website. There are various others versions for the same dataset are also available.

Link of the dataset: <https://www.kaggle.com/shadabhussain/flickr8k>

I have uploaded my project on GitHub.

Link of project on GitHub: <https://github.com/SakshamSoni-code/Image-Caption-Generator>

CONCLUSION

Automatic image captioning is far from mature and there are a lot of ongoing research projects aiming for more accurate image feature extraction and semantically better sentence generation. We successfully completed what we mentioned in the project proposal but used a smaller dataset (Flickr8k) due to limited computational power. There can be potential improvements if given more time. First of all, we directly used pre-trained CNN network as part of our pipeline without fine-tuning, so the network does not adapt to this specific training dataset. Thus, by experimenting with different CNN pre-trained netw

FUTURE WORK

- We can deploy as a web app
- Improvisations: We can take more generalized dataset which may produce more accurate results
- Deep Learning model to road accidents using this as pre model.
- Another potential improvement is by training on a combination of Flickr8k, Flickr30k, and MSCOCO. In general, the more diverse training dataset the network has seen, the more accurate the output will be. We all agree this project ignites our interest in application of Machine Learning knowledge in Computer Vision and expects to explore more in the future.
- Advanced loss function: The original softmax loss function can cause problems. It can produce force negative. For example, if we input the test picture with caption A man is riding a horse, the produced caption A horse is carrying a horse will produce high loss, but actually these two caption all correctly describe the picture. On the other hand, the model can also produce force negative.

REFERENCES

1. Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. IEEE Trans. Pattern Anal. Mach. Intell., 39(4):664–676, Apr. 2017..
2. https://www.ripublication.com/ijaer18/ijaerv13n9_102.pdf
3. Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C. Courville, Ruslan Salakhutdinov, Richard S. Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. CoRR, abs/1502.03044, 2015.
4. <https://bit.ly/3Ej5he1>
5. <https://nlp.stanford.edu/projects/glove/>
6. <https://keras.io/api/applications/resnet/>
7. Jyoti Aneja, Aditya Deshpande, Alexander Schwing, Convolutional Image Captioning, [Online] Available: <https://arxiv.org/pdf/1711.09151.pdf>