Automated Image Captioning with Convolutional Neural Networks and Recurrent Nets

Siddharth Mittal
150718
sidm@iitk.ac.in

Group No. : MLG19

Jaismin Kaur
150298
jaismin@iitk.ac.in

Rohit Gupta
150593
rgupta@iitk.ac.in

Rohit Kumar Bose
150596
rohitkb@iitk.ac.in

Ankit Bhardwaj
150101
bhankit@iitk.ac.in

Abstract

Given an image, we want to generate captions that convey key information about it in English. We will approach the problem in supervised setting where we will work with data-sets that will have images and corresponding descriptions with them. Firstly, we shall be discussing existing works; then gradually move on to the methods that we have proposed. We study the effect of different data-sets, models and parameters of the accuracy of our intended purpose. Detailed statistics have been provided for effective comparison.

1 Introduction

1.1 Problem Statement

The project is aimed at detecting features of an image and producing a caption that effectively describes it using a combination of different deep neural nets. An example is as follows:

black and white dog jumps over bar

1.2 Problem Motivation

We searched we for numerous ML applications on internet and found this problem very convincing. A picture is worth 1000 words. This decade has been the decade of images. Image collection have become extremely large with current camera technology and smart phones advancing to new heights. Image tagging and captioning is one of the famous and fairly challenging problems with many

applications. Many products like Google, Facebook, Flicker etc., take use of image tagging, to a great degree of success. The use of captions with images can provide numerous useful functions.

Some examples are mentioned below:

- Point out a specific piece of content.
- Explain some icon or graphic for sight-impaired people
- Summarize the meaning of some region.

Most searches are done via textual queries, thus there must be a mechanism to link applicable keywords or phrases to images. For blind persons, being able to convey information about the image in another medium would be good for accessibility.

1.3 Existing Work/Literature survey

Image captioning is a famous problem and quite a lot of work has been done in last two decades. Researchers have implemented various sorts of techniques to solve this old age problem.

One technique is to get words of small phrases for different image segments and join then using grammars.

- Matching Words and Pictures: A approach for modeling multi-modal data sets, focusing on the specific case of segmented images with associated text. Learning the joint distribution of image regions and words has many applications.[1]
- Connecting Modalities: Semi-supervised Segmentation and Annotation of Images Using Unaligned Text Corpora : a semi-supervised model which segments and annotates images using very few labeled images and a large unaligned text to relate image regions to text labels.[2]

Many have worked on understanding the scene, identifying and relating all the objects.

- A Sentence is Worth a Thousand Pixels: A holistic scene understanding where images are accompanied with text in the form of complex sentential descriptions. A model for which reasons jointly about which objects are present in the scene, their spatial extent as well as semantic segmentation, and employs text as well as image information as input.[3]
- Decomposing a Scene into Geometric and Semantically Consistent Regi ons: Scene understanding involves reasoning about objects, regions, and the 3D relationships between them. This requires a representation above the level of pixels that can be endowed with high-level attributes such as class of object/region, its orientation, and (rough 3D) location within the scene.[4]

Another common approach is to formulate captioning as a information retrieval problem where most reasonable caption is given to the test image.

1.4 Novel Work

We implemented our model in Keras, which is a very easy-to-understand library for Python. As far as our knowledge, this has not been done yet. Moreover we also surveyed the effectiveness of using different CNNs, different text embeddings as well as different RNNs. We fortified some well known hypotheses.

1.5 Work flow

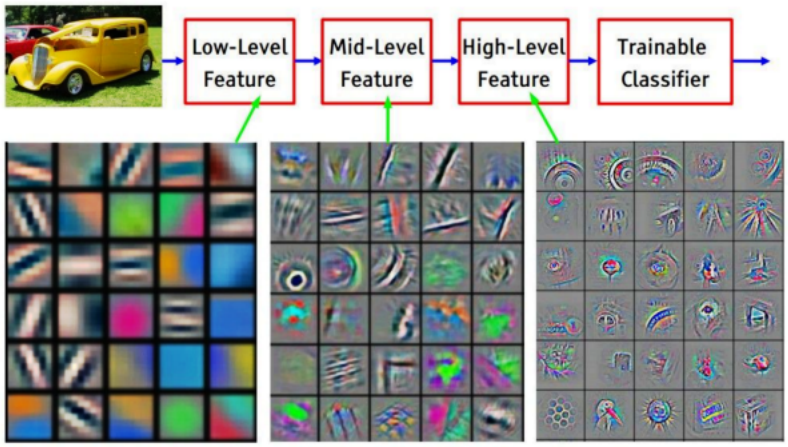
An overly simplified algorithm depicting the work-flow of our model is as follows:

Input $image$, $part\_caption$
$img\_enc \leftarrow \text{CNN}(image)$
$txt\_emb \leftarrow \text{TEXT\_EMBEDDING}(part\_caption)$
$full\_caption \leftarrow \text{RNN}([img\_enc, txt\_emb])$
Return $full\_caption$

2 Methodology

2.1 Input

As mentioned in the work flow earlier, the input to our model is $[image$, $part\_caption]$. We route the image through the CNN to extract features that are characteristic of that image. We shall be using pretrained CNNs, namely VGG16 and Inception.



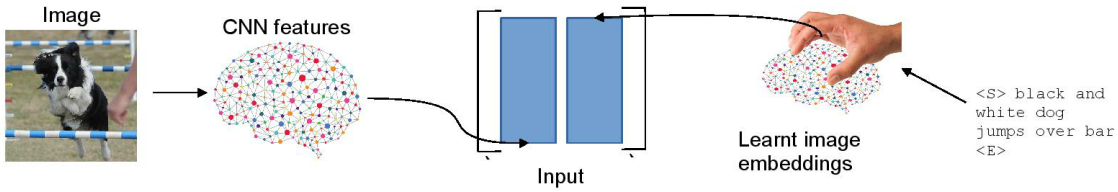
Example representation learnt by a CNN

During training time, we construct a dictionary that maps each word to an index. Similarly, we construct an inverse dictionary that maps each index to a word. This will be helpful during validation/testing. The size of the dictionary will be referred to as VOCAB_SIZE hereafter.

All captions in the data-sets have length ~20. We have constructed a matrix representation of the caption. This is a MAX_CAP_LENGTH x VOCAB_SIZE, where each row of the matrix is a one-hot vector whose i^{th} entry is marked one, where i is the index of the word in the dictionary. For example, say the words 'black', 'and', 'white', 'dog', 'jumps', 'over', 'bar' have indices 4,2,5,8,7,3,6 respectively, our matrix will look like this:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & ... & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & ... & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & ... & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & ... & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & ... & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & ... & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & ... & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & ... & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & ... & 0 & 0 \\ - & - & - & - & - & - & - & - & - & ... & - & - \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & ... & \vdots & \vdots \\ - & - & - & - & - & - & - & - & - & ... & - & - \\ - & - & - & - & - & - & - & - & - & ... & - & - \end{bmatrix}$$

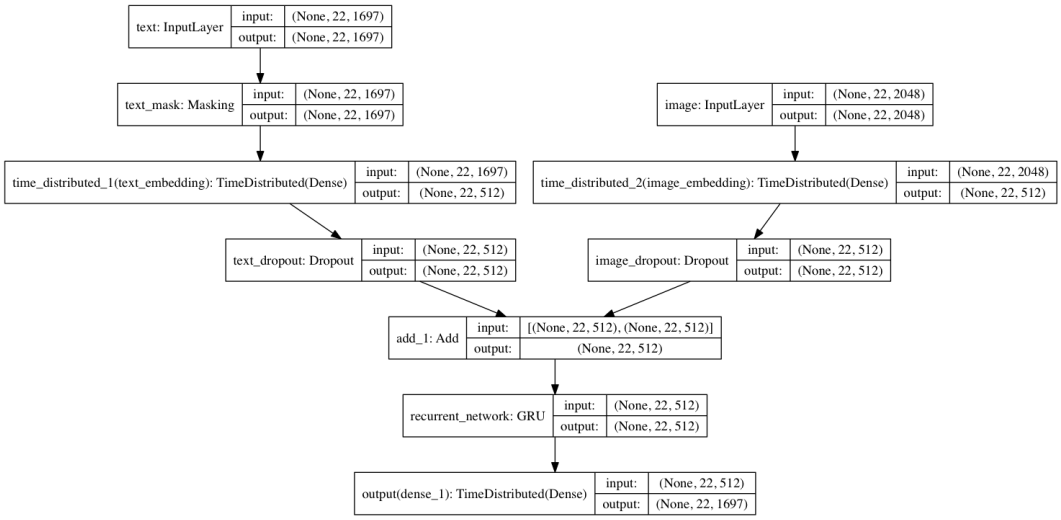
The matrix is zero-indexed. Observe that the 0^{th} and the $(n+1)^{th}$ entry also have 1s at some index. These indices correspond to the start and end tokens respectively. The reason for providing these is that we need to know when the sentence has ended. The start token is needed during testing time, whose details have been given below. The start token and end token will be referenced to as <S> and <E> from now onwards. We will be learning embeddings for the words using a language model. Also note that '-' represents 'nothing'.



The training set has been divided into 2 parts for training and validation in the ratio 80:20.

2.2 Model

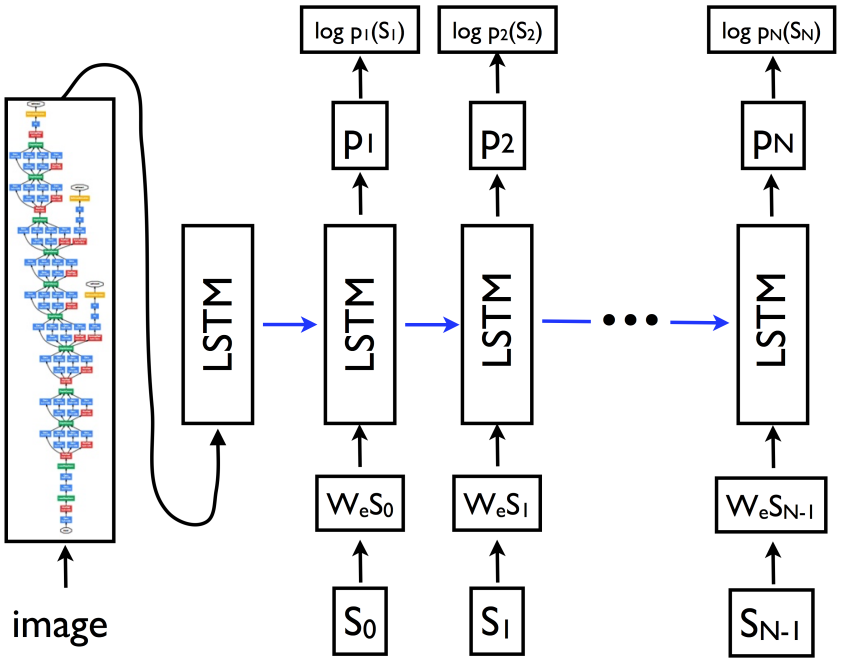
A visual representation of the neural net we are using along with their dimensions are given below:



2.3 Training

- Words in the captions are represented with an embedding model. Each word in the vocabulary is associated with a fixed-length vector representation, that is, the embedding which is learned during training.
- For language modelling we are using Long Short-Term Memory networks and their variants. RNNs are commonly used for sequence modeling tasks such as language modeling and machine translation.
- In the Show and Tell model, the RNN network is trained as a language model conditioned on the image encoding.

- For each word in a caption, we add the image encoding obtained as an output of CNN, with the embedding of word. We treat occurrence of each word in a caption as a time step, and send the add vector as input to the RNN.
- RNN uses its hidden state along with the input vector to compute the output at this step. The sequence of outputs obtained are fed through the a dense network, which get back the one hot representations. This output is compared with the true output.



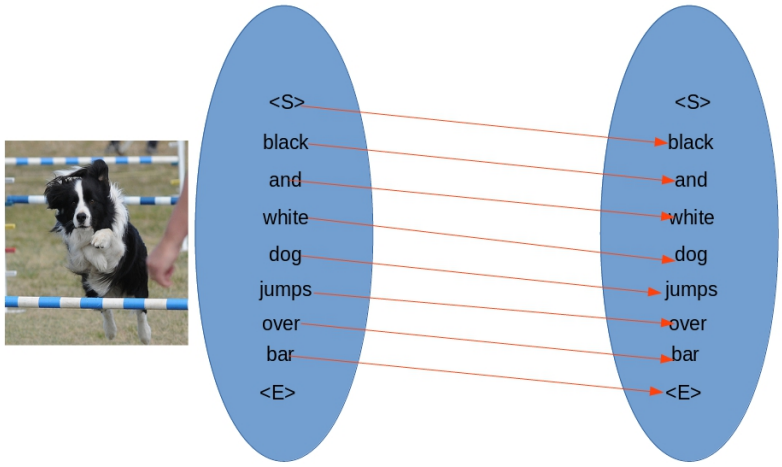
The above image is not entirely right because we are sending the image vector as input added with the word embedding at each time step, not just at first time step.

2.4 Output

The output is a matrix as well. Essentially it is the same matrix as above, except for the fact that it is shifted up by one row so that each token now corresponds to its next token. The corresponding matrix for the above matrix is:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \dots & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ - & - & - & - & - & - & - & - & - & \dots & - & - \\ - & - & - & - & - & - & - & - & - & \dots & - & - \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \dots & \vdots & \vdots \\ - & - & - & - & - & - & - & - & - & \dots & - & - \\ - & - & - & - & - & - & - & - & - & \dots & - & - \end{bmatrix}$$

For this particular example, we get the following mapping:



Mapping of words to words given image

This mapping represents the ground truth. While testing, we apply a soft-max layer over the outputs of the last layer and take the index which has the greatest value (arg max) as the next predicted word, given a partial caption.

2.5 Testing

During testing we obviously do not have any partial caption to begin with, so we simply pass <S> to our model. We do have the image encoding though. Coupled with that, we send the caption representation matrix with only the start token highlighted as 1 in the first one-hot vector. We find out the caption iteratively. In the second iteration, we pass the output matrix of the first matrix coupled with the image encoding. We do this till <E> is obtained. The final caption is obtained by using the inverse dictionary. The index of the word having the greatest probability in each row is used as the inverse dictionary index. Beam search was also implemented for faster and better searching.

3

4 Experimental Details

4.1 Data

Experiments were conducted on the following data-sets:

1. FLICKR8K : Includes images obtained from the Flickr website. 8000 images with 5 captions for each image. The images do not contain any famous person or place so that the entire image can be learnt based on all the different objects in the image.
2. MICROSOFT COCO 2014 : 20000 images with 5 caption per image.
3. IAPR 2012 : 20000 images with 1 caption per image

Preprocessing:

- Owing to shortage of space on the CSE GPU server, we had to subset the COCO data-set because it is very large. A sample of 30000 images and their corresponding captions were taken.
- Each image was re-sized to a dimension of 224x224x3 pixels.
- Each image was converted to its encoded format using different conv-nets.
- Annotations/captions were given in various formats e.g. JSON. Converted them into simple CSV files. A start token <S> and an end token <E> were added to each of them.
- We removed words have very less frequency (< 2) in the entire dictionary of caption words because they, being outliers, would interfere with the results of the experiment.
- Initially used 50 dimensional GLOVE word embeddings for each word, so that they can be fed into the RNN. They were giving poor results so we tried to learn our own embedding.

4.2 Tuning

Since the model learnt almost everything except the CNN weights, many of the parameters were tuned. A few salient tuning parameters are as follows:

- **Data-set** : Upon testing with Flickr8k, COCO and IAPR datasets, we found that the best dataset was indeed COCO, and IAPR was significantly worse than the other two.
- **Choice of CNN** : From among Inception and VGG16, Inception gave significantly better results. This confirms the fact that Inception has an accuracy of 79%, whereas VGG16 has 71%.
- **Use LSTM / GRU** : Although it is a known fact that GRUs are better than LSTMs, we decided to test this because it is theoretically believed that LSTMs have more memory than GRU. Since our caption lengths were restricted to 20, LSTM did not show any significant difference. Training with GRU was fast and gave good results.
- **RNN dimensions** : We trained with dimensions 256 and 512. Using more units makes it more likely to perfectly memorize the complete training set, so naturally 512 RNN size gave somewhat better results.
- **Text embedding dimension** : Among 50 and 100, 100 text embedding dimensions proved to be better.

4.3 Experimental Results

BLEU SCORES : Metric of similarity between human translated and machine translated sentences

TRAIN	MODEL	TEST BLEU	
		Flickr8k	COCO
Flickr8k	LSTM	0.48	0.40
	GRU	0.49	0.42
COCO	LSTM	0.51	0.57
	GRU	0.52	0.60

Table 1: CNN used : Inception

TRAIN	MODEL	TEST BLEU	
		Flickr8k	COCO
Flickr8k	LSTM	0.42	0.36
	GRU	0.42	0.39
COCO	LSTM	0.49	0.55
	GRU	0.50	0.54

Table 2: CNN used : VGG16

5 Future Work

Given the fact that every one in our group was a complete novice in deep-learning, we managed to learn quite a lot about how neural networks work. We managed to learn how to train a language model and a recurrent neural network added with a convolutional neural network.
We can also caption images in Indian languages.
In future, it is possible to include an attention model that will focus on particular areas of the picture and help classify the images better. Some work has already done on this. Since image captioning is indeed a very difficult task, we have a long way to go till we give the machine sufficient intelligence so that it can understand any image correctly.

References

[1] K. Barnard, P. Duygulu, D. Forsyth, N. De Freitas, D. M. Blei, and M. I. Jordan. Matching words and pictures.JMLR,2003.

[2] R. Socher and L. Fei-Fei. Connecting modalities: Semi-supervised segmentation and annotation of images using unaligned text corpora. In CVPR, 2010.R. Socher and L. Fei-Fei. Connecting modalities: Semi-supervised segmentation and annotation of images using unaligned text corpora. InCVPR, 2010.

[3] S. Fidler, A. Sharma, and R. Urtasun. A sentence is worth a thousand pixels. In CVPR, 2013.

[4] S. Gould, R. Fulton, and D. Koller. Decomposing a scene into geometric and semantically consistent regions. In Computer Vision, 2009 IEEE 12th International Conference on, pages 1–8. IEEE, 2009.