Virtual Recognition

Mini Project 2

Image Captioning with CNN - LSTM system

- By Team 7

Dhamodhar Reddy (IMT2019026)

Vignesh Bondugula (IMT2019092)

Tarun Kumar (IMT2019011)

Contents

Introduction

Aim: Design a CNN-LSTM system that can perform image captioning based on Flickr 8K dataset.

As mentioned in the problem statement, we used a CNN-LSTM model to train the Flickr 8K dataset and analyze the model's weakness. Let's first see how we built the model.

Preprocessing

Text Preprocessing: Even though we were provided with the lemmatized text descriptions along with the dataset, we preferred original descriptions over the lemmatized description. Training with lemmatized descriptions gives lemmatized output captions and hence would result in poor bleu scores. So we chose the original descriptions. Now to these original texts, we performed the following text preprocessing techniques to clean the texts:

- Removed punctuation.
- Removed tokens which contained both alphabets and numerals.

Word Embeddings: Now from this preprocessed text, we need to find good word embeddings instead of just one hot encoding, to boost generalization and performance by representing words as semantically-meaningful i.e learn the required spatial relations between words by modifying these vectors along with reducing the dimensionality of the vocabulary.

GloVe is an unsupervised learning algorithm for obtaining vector representations for words. Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space. What we use here is just a pretrained version of this model for around 6 billion tokens called the GloVe6b. We used a particular variant which has 200 feature embeddings for every word.

Network Architectures

CNN Model: To start we need to extract some meaningful features from the 2 dimensional images. For this we can use any CNN model as a feature extractor. We used two different CNN models - ResNet50 and InceptionV3.

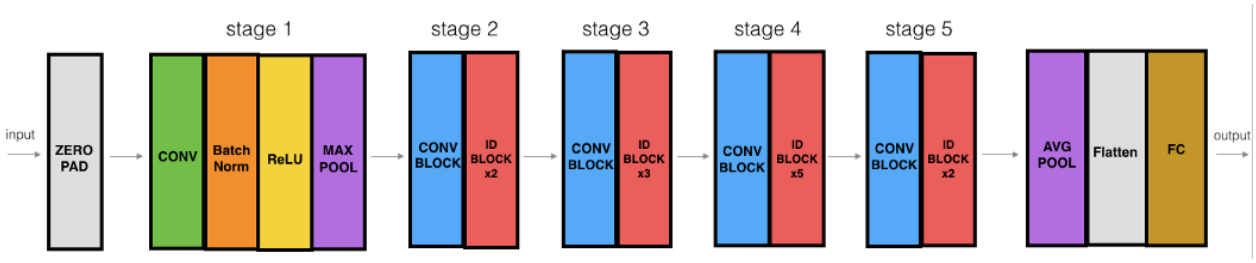
ResNet50 Architecture: ResNet, short for Residual Networks is a classic neural network used as a backbone for many computer vision tasks. The ResNet model helps us to train extremely deep neural networks with 150+layers successfully.

ResNet uses skip connection to add the output from an earlier layer to a later layer. This helps it mitigate the vanishing gradient problem. This problem of training very deep networks has been alleviated with the introduction of ResNet or residual networks and these Resnets are made up from Residual Blocks.

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

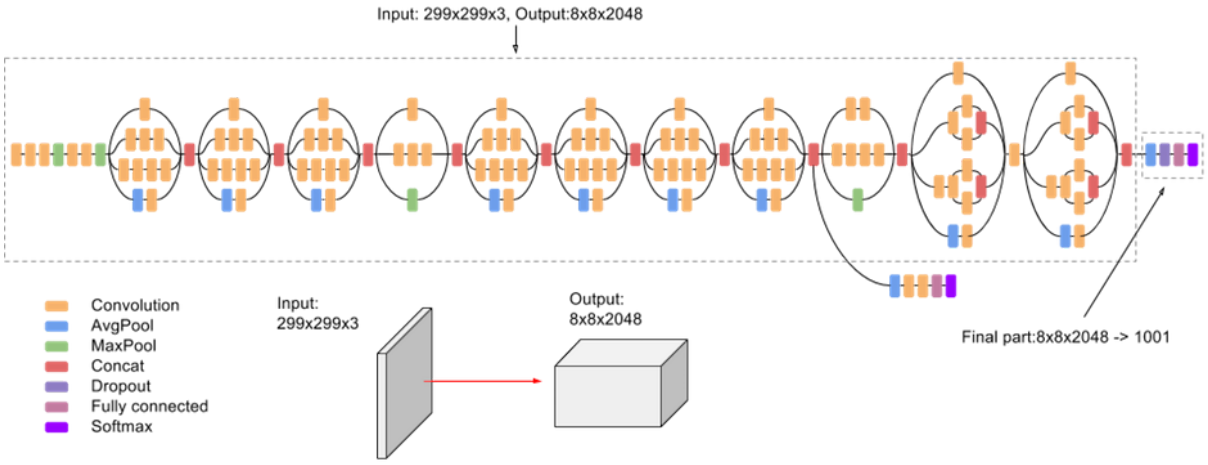
Building blocks are shown in brackets (see also Fig. 5), with the numbers of blocks stacked. Downsampling is performed by conv3 1, conv4 1, and conv5 1 with a stride of 2

The architecture we chose for this project is ResNet50, a variant of the ResNet convolutional network which has 48 Convolution layers along with 1 MaxPool and 1 Average Pool layer. (Total 48+1+1 = 50 layers, hence the name ResNet50)



We also tried extracting features with another CNN architecture called InceptionV3, developed by a team at Google.

InceptionV3 Architecture:



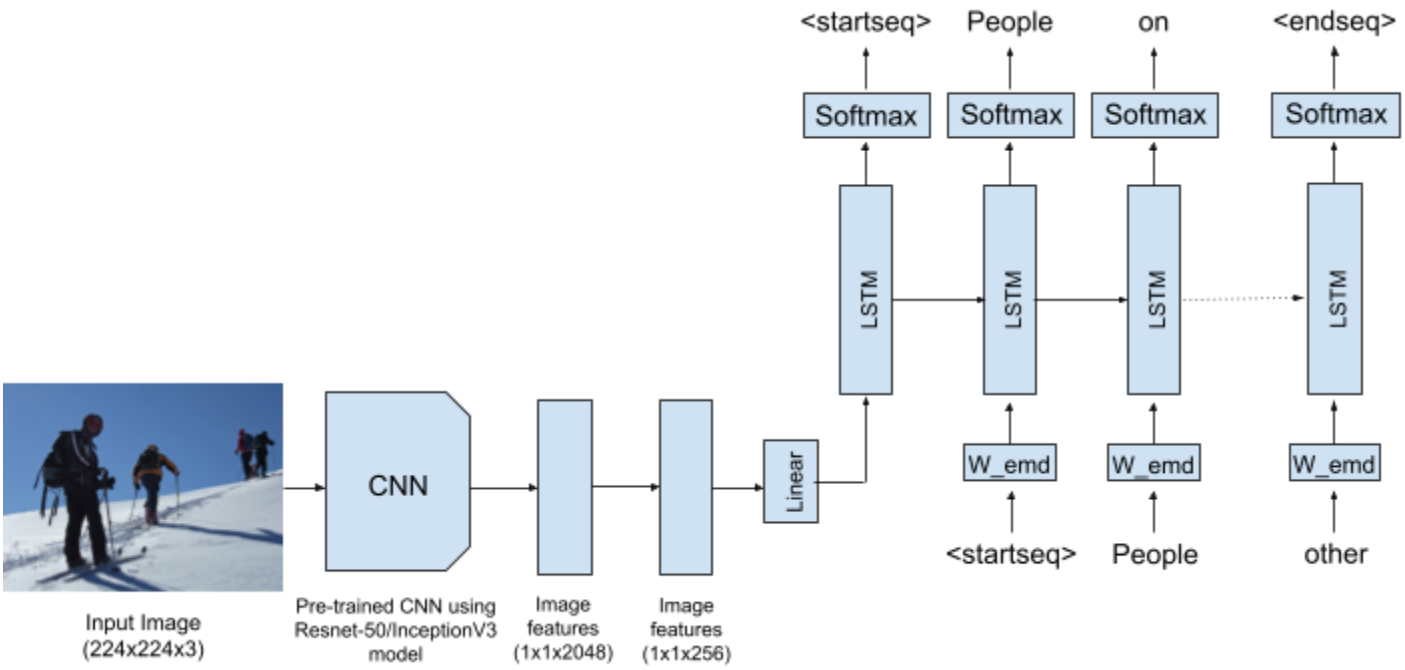
The above image describes the inception V3 architecture. We have also tried to extract image features using this pre-trained model. The inception architecture is built progressively in different steps. Factorized convolutions are performed in order to reduce the computational efficiency as it reduces the number of parameters involved in a network. Bigger convolutions are replaced by smaller convolutions which leads to faster training. Asymmetric convolutions also take place. Grid size reduction also takes place which is usually done with the help of pooling layers. This architecture the image features were obtained faster but, the image features obtained in the previous model gave comparatively better results. Hence, we progressed further with resnet-50 architecture.

LSTM Model

Now that we have the image features, word embeddings the last step is to create a sequential learning model. Since we were asked to use LSTMs we used the following LSTM architecture to train the model.

The inputs to the model are the preprocessed image (224, 244, 3) and the integer tokens of the captions. The image's features are extracted (we have already done this) and reduced to 256 dimensions using a Linear layer with ReLU activation.

The vectorized image representation is fed into the network, followed by a special start of sentence token. The hidden state produced is then used by the LSTM to predict/generate the caption for the given image.



We then used Categorical Cross Entropy loss for updating the parameters. Also, we used Adam optimizer with a constant learning rate. We trained for around 15 epochs and observed that the loss almost converged within those epochs.

Evaluation

Greedy search and beam search aim to generate the sequence outputs of tokens from a neural network model

Greedy Search:

As the model generates a 1654 long vector with a probability distribution across all the words in the vocabulary, in greedy search we greedily pick the word with the highest probability to get the next word prediction.

Beam Search:

Coming to the Beam search evaluation technique, we take top k predictions, feed them again in the model and then sort them using the probabilities returned by the model. So, the list will

always contain the top k predictions and we take the one with the highest probability and go through it till we encounter ‘endseq’ or reach the maximum caption length.

BLEU Scores: (Bilingual Evaluation Understudy Score)

- Average Sentence BLEU score with greedy : **0.554** (Tested on 500 random images from test images)
- Average Sentence BLEU score with beam (k = 3) : **0.563** (Tested on 200 images from test images)

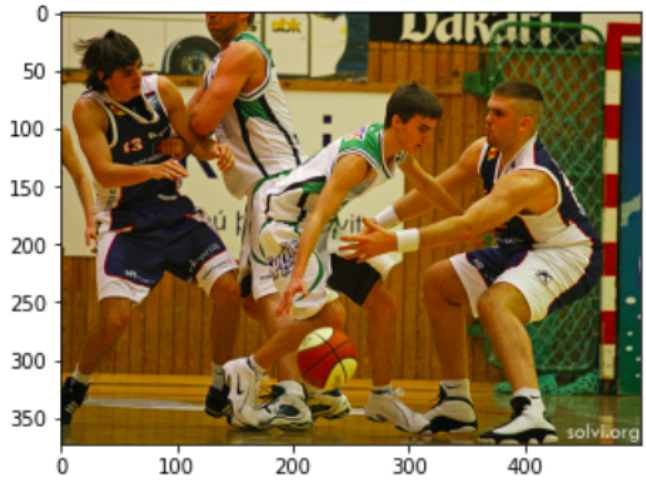
BLEU-1: 0.5242
BLEU-2: 0.3427
BLEU-3: 0.2225
BLEU-4: 0.1382

- Average Sentence BLEU score with beam (k = 5) : **0.549** (Tested on 200 images from test images)

BLEU-1: 0.5010
BLEU-2: 0.3207
BLEU-3: 0.2076
BLEU-4: 0.1278

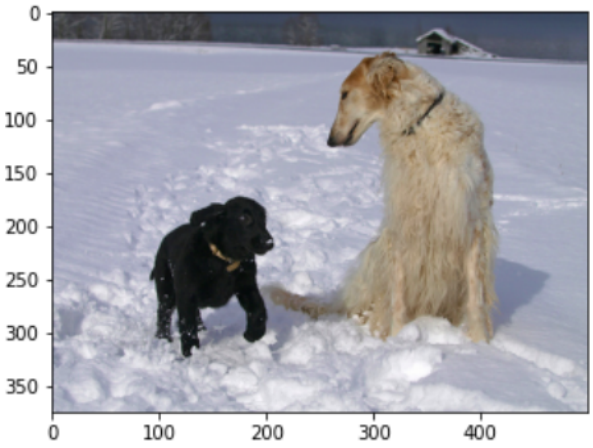
Sample Predictions from the model

Prediction 1



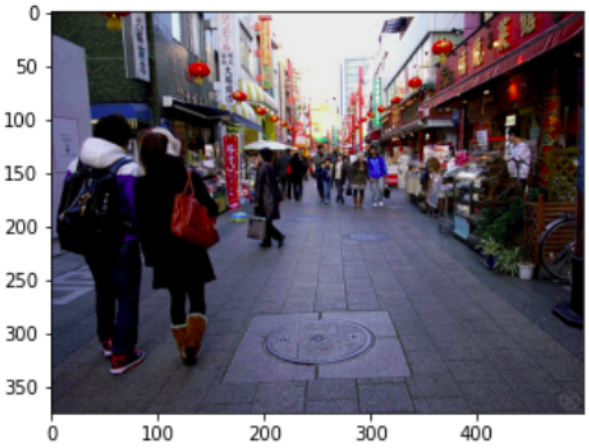
a basketball player in a white uniform is being thrown by the opposing team

Prediction 2



two dogs are running through the snow

Prediction 3



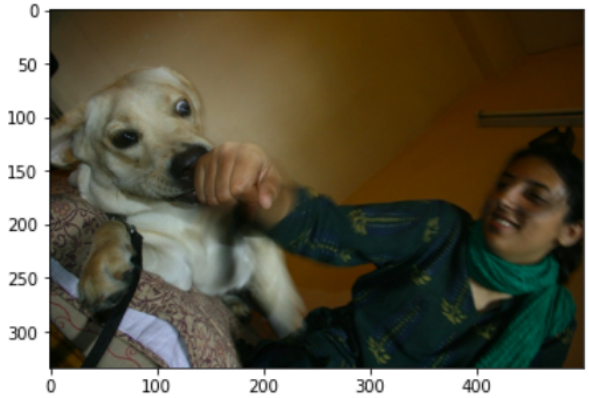
a busy street with people walking around

System 1 Modified

The LSTM based image captioning can 'blindly' learn the structure of the language and predict meaningful sentences even without learning much insight to the content of the image. This is termed as "**language bias**" of the system.

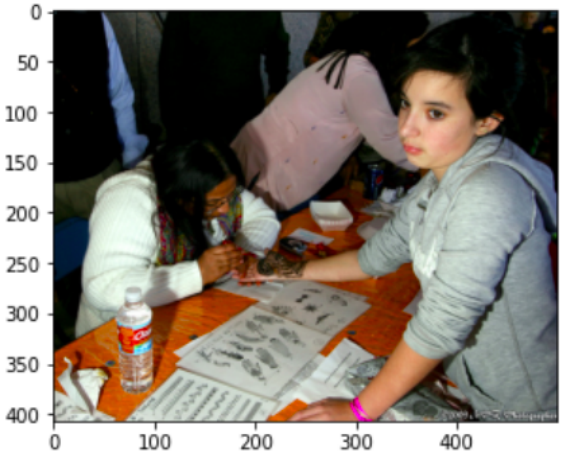
Experiment 1(Histogram)

Firstly, we have tried it by using a histogram. We have taken all the test predictions and tried to find out the number of times a 4-word phrase has repeated. We also did the same for the real predictions. If the most occurring phrase is found out to be repeated for a large number of times, which is larger than the number of occurrences in true captions, then we could tell that the model has language bias. This is because if a particular phrase is repeated a huge number of times then it explains that our model is predicting the caption without taking the image into consideration. It is predicting the next word based on the given description itself. This clearly is language bias. For example, the phrase "a man in a" has occurred 6 times in the test descriptions for the first 200 images. But when we generated a histogram for our predictions, we found out that the phrase "a man in a" has occurred 44 times. Which clearly tells that the phrase is used in images, in which it doesn't even depict it. Hence, our model has language bias. We can give a numerical value to bias by calculating the difference in the occurrences of n-word phrases. The higher the difference for higher n, the higher is the language bias. Few examples are given below,



a little girl with a pacifier in its mouth is sitting in a chair with a man in a blue shirt

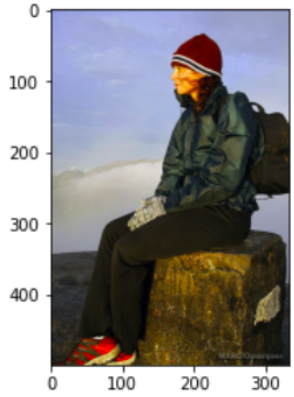
a man in a white shirt and white pants is standing in front of a crowd



a man in a white shirt and jeans is sitting on a bench with his feet on a tripod

All these images predicted the phrase “a man in a” blindly because of the structure of the sentence but not because of the image. All the images do not depict a man, but still they are captioned in that way due to language bias.

a man in a red shirt with a black bandanna and sunglasses is smiling

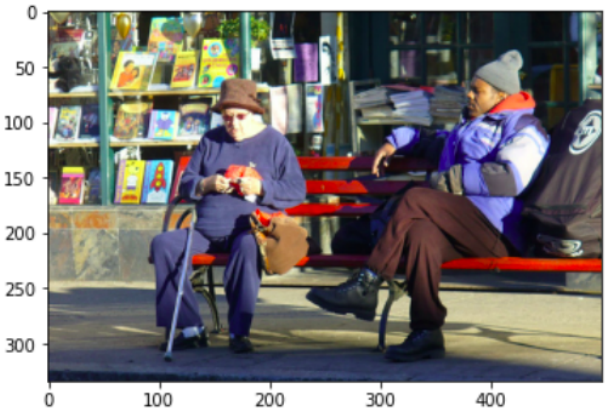


a man in a red shirt and blue hat is sitting on a bench with his dog

a little boy in a red shirt is playing with a toy

All these images predicted the phrase “in a red shirt” blindly because of the structure of the sentence but not because of the image. There are a lot of training captions in which we find the “in a red shirt” attached to a human. All the images do not contain the red color shirt but are still captioned that way.

a man in a green shirt is standing in front of a red building



a group of people are standing in front of a door

a little girl in a red shirt is standing in front of a house

All these images predicted the phrase “standing in front of” blindly because of the structure of the sentence but not because of the image. Most of the “in front of “ phrases in the training were attached to the phrase “standing” before. So even when the model looks at images where OBJECT is sitting in front of OBJECT, we can see that it is being captioned as standing.

Experiment 2 (**Using Image detection**)

If we have a high BLEU score for the image captioning model that we need to detect the for language bias, we can first try to detect the list of objects present in the image we want to caption using any well known object detection models, and then try to see how

well they are correlated or how close those are using any distance metric between the objects that were predicted in the image detection models and the words that were predicted in the captions.

If we find that the distance between the list of objects detected and the list of words in the caption is high we can flag that there might be a language bias as we have a high BLEU score but high distance as well.

Experiment 3 (**Using BLEU score**)

One way to get closer to detect language bias is with the help of modified BLEU scores. This function checks the correctness of the generated sentence at multiple levels, as specified by the user. That is, it checks uni-gram matches (single words, BLEU-1), bi-gram matches (two word pairs, BLEU-2), tri-grams, and so on.

So if we find very high BLEU-1 and BLEU-2 scores but surprisingly low BLEU-3 or BLEU-4 scores, then there is a high chance that we might have observed language bias. Because the BLEU-4 scores show how well the 4-grams are matched between predictions and references. So the model figured out some part of the true prediction and filled the remaining part of the sentence from the structure it learned from training captions. (instead of looking at the image as well)

Examples:

a little boy in a red shirt is sitting in the snow
(0.5643211499270759, 0.41677879041417215, 0.2480434035330246, 4.07941328593143e-78)

Caption along with BLEU scores

a man in a green shirt is sitting in a chair
(0.7272727272727273, 0.46709936649691375, 1.804819547402325e-102, 1.0194756444945366e-154)

References

- Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun , “Deep Residual Learning for Image Recognition” - referred to get a brief idea on the ResNet CNN model architecture.
- Jessica López - “Understanding greedy search and beam search” https://medium.com/@jessica_lopez/understanding-greedy-search-and-beam-search-98c1e3cd821d
- Advanced Guide to Inception v3 - https://cloud.google.com/tpu/docs/inception-v3-advanced
- https://www.researchgate.net/publication/329751273_Learning_to_Evaluate_Image_Captioning
- https://www.hindawi.com/journals/cin/2020/3062706/