MINI - PROJECT REPORT

IMAGE CAPTIONING

IMT2019032 Ganesh
IMT2019017 Amish
IMT2019524 Srijan

Team ID - 30

Abstract

We have made a CNN-LSTM system for generating captions for images.This report contains the approach we used and applied BLEU score evaluation. We have designed a training experiment with Flickr 8K data to assess the language bias of your CNN-LSTM system.

Introduction

The image captioning system is implemented the using the Keras library. We have used Convolutional Neural Network(CNN) as an encoder and an LSTM-based Recurrent Neural Network as a decoder.

Training machine learning models to automatically generate captions for images is a very challenging task in the fields of Computer Vision and Machine learning. This task is a combination of Image understanding, feature extraction and conversion of visual representation into natural languages.

With many advancements in Neural networks, CNNs and RNNs can be effectively used to perform this task to achieve accurate results. This project can greatly help visually impaired people to better understand the context of any image on the internet.

Dataset

Flickr 8k dataset is used .

It contains 8000 images with each image being labeled with 5 different descriptions. The dataset contains a lemmatized version of text data along with train,validation and test subsets.

Since we are using the pre-trained ResNet model, first our input images are converted into 3 channel RGB images of shape (3 x 22 4 x 224). After resizing the images, we construct a dictionary of words i.e a corpus of most frequently occurring words in captions. Then all these words are vectorized.

Implementation

We have used the Flickr 8k dataset for training our model. We have used the ResNet pre-trained Convolutional Neural Network model as an encoder to extract and encode the image features into a higher dimensional feature space.

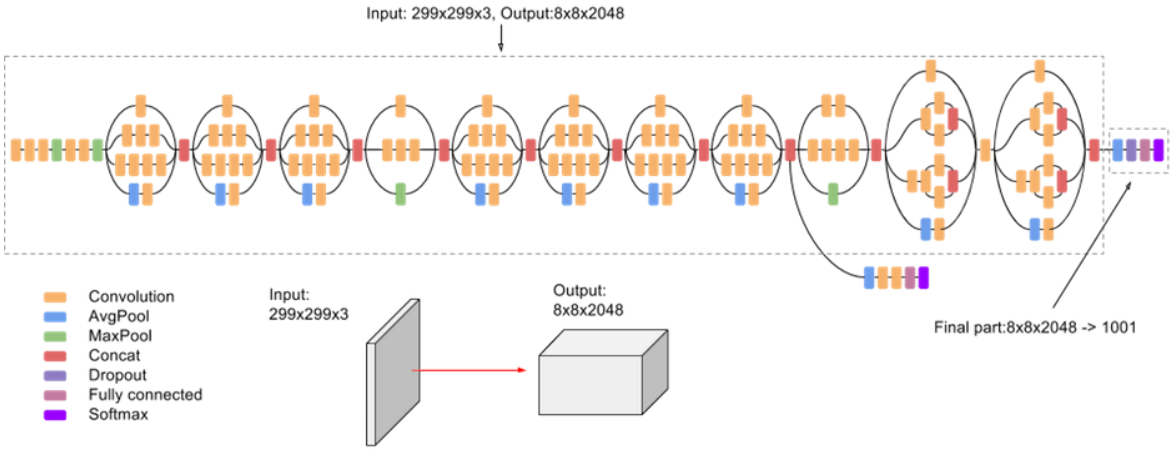
An LSTM-based Recurrent Neural Network is used as a decoder to convert the encoded features.

CNN encoder

A Convolutional Neural Network is a Deep Learning system that can take an image as input, generate weights and biases for multiple objects in the image, and distinguish between them.

The Inception-v3 model is extensively utilized, and it has achieved over 78 percent accuracy on the Image net dataset.

Convolutional Layer : This layer requires input data, a filter or feature detector, and a feature map, among other things. A 3x3 matrix is the most common filter size. The filter is then applied to a portion of the image, and the dot product of the input pixels and the filter is determined. After that, the dot product is loaded into an output array.



The filter then moves and repeats the operation until the kernel has covered the entire image. A feature map, activation map, or convolved feature is the ultimate output of a series of dot products from the input and the filter.

Pooling Layer : Pooling layer conducts dimensional reduction. Similar to the convolutional layer, the pooling operation sweeps a filter across the entire input, but the difference is that this filter does not have any weights.

In Max pooling, as the filter moves across the input, it selects the pixel with the maximum value to send to the output array. In Average Pooling, as the filter moves across the input,it calculates the average value within the receptive field to send to the output array.

Fully connected Layer : This layer performs classification tasks based on the features retrieved by the previous layers and their various filters. While convolutional and pooling layers typically utilize ReLu functions to categorize inputs, FC layers typically use a softmax activation function to produce a probability from 0 to 1. As the image data passes through the CNN's layers, it begins to recognise larger elements or shapes of the object, eventually identifying the desired object.

For our task, the encoder needs to extract image features of various sizes and encode them into vector space which will be sent to a RNN in later stages. Since our CNN need not classify images but encode features, we removed the fully connected layers and the max pool

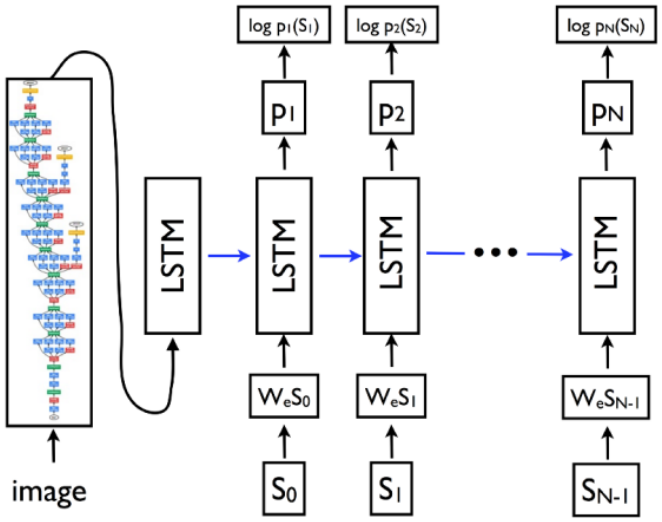
layers at the end of the network. As we can see from the above picture, the circled layers are removed from our architecture

LSTM decoder

Long short-term memory networks are an extension for recurrent neural networks, which basically extends the memory.LSTMs enable RNNs to remember inputs over a long period of time.

A gated cell can be compared to this memory. There are three gates in an LSTM: input, forget, and output. These gates control whether fresh input should be allowed (input gate), whether it should be deleted (forget gate), or if it should have an impact on the output at the current timestep (impact gate) (output gate).

For our task, by using LSTMs the decoder needs to generate image captions word by word. The input for the decoder is the encoded image feature vectors from CNN and the encoded image captions produced in the data preprocessing stage. In each iteration of the LSTM network, we concatenate the embedded captions of all previous words and the encoded images and feed them to the LSTM to get the next state of LSTM.



FC layers uses the softmax activation function can predict the probabilities of current word embedding based on the current state and append it to the word embedding prediction matrix

Loss function - The nature of our RNN output is a series of word occurrences, and in order to increase the quality of the RNN output, we use Cross Entropy Loss. This is the most effective measurement for the performance of a classification model whose output is a probability value between 0 and 1. We use Adam optimizer , an adaptive learning rate optimization algorithm that has been designed specifically for training deep neural networks.

Word embeddings - We need to convert the captions for training images into embedded captions,so we need to create word embeddings. We can create either by using the corpus of data we have as captions or either we can use pre-trained word embeddings like gloves.

We tried using glove embeddings for Image captioning.

Beam search - We should decode the most likely output sequence involving searching through all the possible output sequences based on their likelihood.

A popular approach in such a case is to use Beam Search. The algorithm is a best-first search algorithm which iteratively considers the set of the k best sentences up to time t as candidates to generate sentences of size t + 1, and keep only the resulting best k of them, because this better approximates the probability of getting the global maximum.

We tried a few Beam search sizes and the BLEU score came out to be better under a size of 5. Hence we set our Beam search size to 3 and also 5.

Model size - We have initially sent the image into inception v3 model and then the word embeddings are taken from pretrained gloves. We also added a dropout layer to avoid overfitting and then to a dense layer.

The output of the dense layer is connected to the LSTM and added two linear dense layers at the end which gives the captions for the input image. The model has approximately 1.7 million trainable parameters.

```
[26] Model: "model_1"
__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
==================================================================================================
 input_3 (InputLayer)           [(None, 38)]         0           []

 input_2 (InputLayer)           [(None, 2048)]       0           []

 embedding (Embedding)          (None, 38, 200)      270200      ['input_3[0][0]']

 dropout (Dropout)              (None, 2048)         0           ['input_2[0][0]']

 dropout_1 (Dropout)            (None, 38, 200)      0           ['embedding[0][0]']

 dense (Dense)                  (None, 256)          524544      ['dropout[0][0]']

 lstm (LSTM)                    (None, 256)          467968      ['dropout_1[0][0]']

 add (Add)                      (None, 256)          0           ['dense[0][0]',
                                                                  'lstm[0][0]']

 dense_1 (Dense)                (None, 256)          65792       ['add[0][0]']

 dense_2 (Dense)                (None, 1351)         347207      ['dense_1[0][0]']

==================================================================================================
Total params: 1,675,711
Trainable params: 1,675,711
Non-trainable params: 0
__________________________________________________________________________________________________
```

Training the Model

The Model trained for 25 epochs with categorical loss entropy as loss function and adam optimizer with a batch size of 3. We tried out these parameters and saved the model into a .h5 file which will be used to run the captioning next time without starting from training, we can load the model weights next time.

Using beam search, we got improved scores. Beam search is performed for different values of k and results are better when we have taken k = 3 or 5.

So beam search is preferred over argmax function i.e greedy search function at the end.

```
       1.4000208 ], dtype=float32), '3383491811_fd9d3a891d.jpg': array([0.30746275, 0.1504609 , 0.280
       0.494645  ], dtype=float32), '2470493181_2efbbf17bd.jpg': array([0.34737527, 0.5220052 , 0.639
       0.35976914], dtype=float32), '2469498117_b4543e1460.jpg': array([0.2855435 , 0.7756386 , 0.227
       0.18797562], dtype=float32), '1858963639_4588cd4be9.jpg': array([0.46408755, 0.6281653 , 0.879
       0.3063981 ], dtype=float32), '3197247245_9c93b60b8a.jpg': array([0.50261736, 0.67657393, 0.194
       0.5453143 ], dtype=float32), '2612608861_92beaa3d0b.jpg': array([0.4125007 , 0.77121854, 0.418
       0.22849633], dtype=float32), '2985439112_8a3b77d5c9.jpg': array([0.24964729, 0.40130907, 0.109
       0.13639782], dtype=float32), '3558251719_3af5ae2d02.jpg': array([0.16403103, 0.22702886, 0.283
       0.17608406], dtype=float32), '458735196_176e7df6b3.jpg': array([0.46022153, 0.41751224, 0.1984
       0.6182482 ], dtype=float32), '2429272699_8a9699775e.jpg': array([0.06069682, 0.3235488 , 0.279
       0.47740197], dtype=float32), '2272426567_9e9fb79db0.jpg': array([0.11169799, 0.12767652, 0.408
       0.8358196 ], dtype=float32), '2448270671_5e0e391a80.jpg': array([0.20587231, 0.9866344 , 0.336
       0.38728142], dtype=float32), '2998185688_8d33e4ce38.jpg': array([0.55778825, 0.52608436, 0.391
       0.94532096], dtype=float32), '3587092143_c63030ed6d.jpg': array([0.86373127, 0.23010768, 0.681
       0.07893074], dtype=float32), '125319704_49ead3463c.jpg': array([0.2624333 , 0.28762656, 0.4893
       0.29869208], dtype=float32), '2571096893_694ce79768.jpg': array([0.20958865, 0.64955443, 0.153
       0.15791404], dtype=float32), '2405599120_ec5f32af6f.jpg': array([0.20086843, 0.2286644 , 0.396
       0.11766049], dtype=float32), '3579686259_b1fe6aefc9.jpg': array([0.17461063, 0.47019017, 0.184
       0.26657566], dtype=float32), '2410562803_56ec09f41c.jpg': array([0.7420616 , 0.1411258 , 0.606
       0.27296388], dtype=float32), '3157039116_d82da4e66b.jpg': array([0.0326595 , 0.04180459, 0.339
       0.73409164], dtype=float32), '2393971707_bce01ae754.jpg': array([0.39396018, 0.70679086, 1.230
       0.26054347], dtype=float32), '2704257993_d485058a5f.jpg': array([0.29562098, 0.24403208, 0.305
       0.37067804], dtype=float32), '2562166462_b43b141d40.jpg': array([0.41892564, 0.25706095, 0.690
       0.5265349 ], dtype=float32), '2912476706_9a0dbd3a67.jpg': array([0.3017848 , 0.34467348, 0.407
       0.62248313], dtype=float32), '1262454669_f1caafec2d.jpg': array([0.45937663, 0.8678382 , 0.044
       0.44424093], dtype=float32), '150582765_bad8dec237.jpg': array([1.310666  , 0.06080621, 0.3406
       0.5456265 ], dtype=float32), '2952751562_ff1c138286.jpg': array([2.2323294 , 0.5141566 , 0.569
       0.08822186], dtype=float32), '3481884992_45770ec698.jpg': array([0.1843724 , 0.        , 0.252
       0.09679963], dtype=float32), '543102698_38e7e38bbc.jpg': array([0.36209783, 0.7530842 , 0.2356
       0.47464216], dtype=float32)}
Epoch 1/30
1211/2000 [=================>............] - ETA: 7:02 - loss: 3.5699
```


```
Epoch 9/25
2000/2000 [==============================] - 218s 109ms/step - loss: 2.3787
Epoch 10/25
2000/2000 [==============================] - 217s 108ms/step - loss: 2.3517
Epoch 11/25
2000/2000 [==============================] - 217s 108ms/step - loss: 2.3262
Epoch 12/25
2000/2000 [==============================] - 217s 109ms/step - loss: 2.3060
Epoch 13/25
2000/2000 [==============================] - 217s 109ms/step - loss: 2.2877
Epoch 14/25
2000/2000 [==============================] - 217s 109ms/step - loss: 2.2690
Epoch 15/25
2000/2000 [==============================] - 218s 109ms/step - loss: 2.2551
Epoch 16/25
2000/2000 [==============================] - 218s 109ms/step - loss: 2.2421
Epoch 17/25
2000/2000 [==============================] - 216s 108ms/step - loss: 2.2315
Epoch 18/25
2000/2000 [==============================] - 221s 110ms/step - loss: 2.2185
Epoch 19/25
2000/2000 [==============================] - 217s 108ms/step - loss: 2.2065
Epoch 20/25
2000/2000 [==============================] - 217s 108ms/step - loss: 2.1973
Epoch 21/25
2000/2000 [==============================] - 219s 109ms/step - loss: 2.1900
Epoch 22/25
2000/2000 [==============================] - 217s 109ms/step - loss: 2.1820
Epoch 23/25
2000/2000 [==============================] - 217s 108ms/step - loss: 2.1731
Epoch 24/25
2000/2000 [==============================] - 219s 109ms/step - loss: 2.1667
Epoch 25/25
 772/2000 [==========>...................] - ETA: 2:13 - loss: 1.9674
```

Evaluation of test data using BLEU

BLEU - Bilingual Evaluation Understudy (between 0 and 1)

The score indicates how similar the given text is with respect to the reference text with values given closer to 1 representing more similar texts.

A perfect score is not possible in practice as a translation would have to match the reference exactly. NLTK provides the sentence- bleu() function for evaluating a candidate sentence against one or more given sentences.

We have 5 different captions for each test image provided. So these 5 captions are given as a reference array and the caption given by the system has been passed into this function to evaluate BLEU score.

Q1 . BLEU Scores

We have generated the captions initially using the word embeddings created from the corpus of caption data and also with glove embedding.

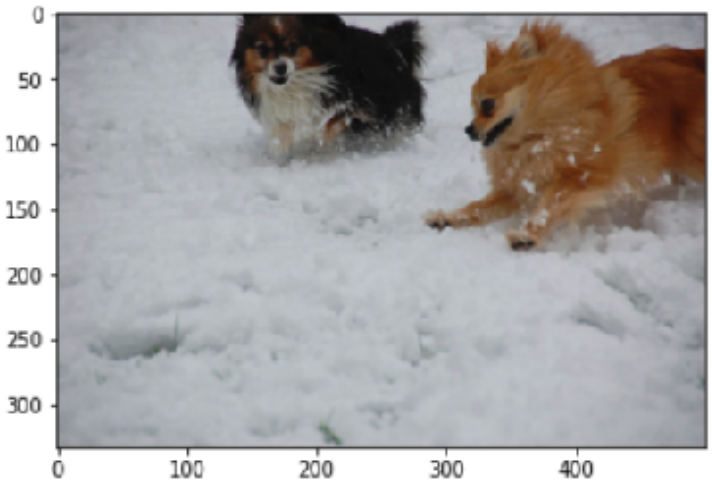
Score using the embedding created - 0.42

Score using Glove Embedding ~ 0.4

Scores using beam search prediction

Greedy search - 0.39
Beam search with k = 3 - 0.46
Beam search with k = 5 - 0.45

Output Image with captions (System) :




```
Greedy Search: a little boy be play with a toy airplane
Beam Search, K = 3: a little boy and a brown dog be play in the snow
Beam Search, K = 5: a little boy be play with a toy airplane in the snow
Beam Search, K = 7: a little boy play with a toy airplane in the snow
Beam Search, K = 10: a little boy and a dog play in the snow
```

Conclusion

- From the first comparison , we can see that the score obtained by using glove embedding is less compared to other one because glove embedding consists of the entire wikipedia corpus.These test images will contain many unrelated words in that embedding.

- The test images are similar to that of training images. So using the corpus created from caption data performed better compared to glove.

-The model using the glove embeddings will produce better captions for all images in general. So it will give better results on subjective images.

- Using beam search prediction will help in generating captions slightly better.

Q2 . Training experiment to assess the LSTM language bias

The LSTM based image captioning can blindly learn the structure of the language and predict meaning of full sentences even without learning much insight to the content of image.This is termed as ‘language bias’ of the system.

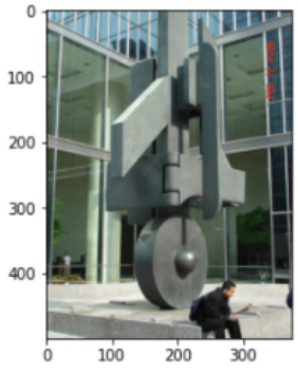
A training experiment on the Flickr 8K data to assess the language bias of the CNN - LSTM is done.

We have used greedy search and beam search with K=2,K=3,K=4

Output Images with captions (System_Modified) :


```
a little girl in a white dress be run along a dirt road
Beam Search with K = 2: a little girl in a pink tutu jump on a trampoline
Beam Search with K = 3: a little girl in a white bathe suit dance in front of a kiddie pool
Beam Search with K = 5: a little girl in a white bathe suit dance in front of a pool
```

We can observe that in the image a girl in white dress on horse is walking on the road along with a man but the captions detected doesn’t match only white dress and girl are detected correctly.

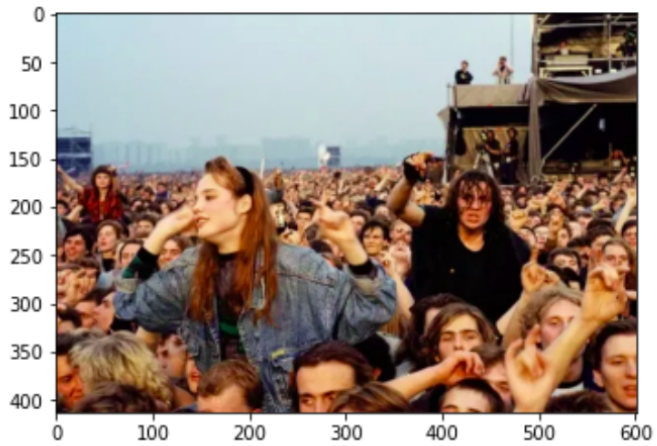



```
a boy in a red shirt be jump off a rock into a lake
Beam Search with K = 2: a boy in a red shirt be jump off a rock into a lake
Beam Search with K = 3: a boy in a red shirt be jump on a bed
Beam Search with K = 5: a little boy in a white shirt and blue jean be climb a rock wall
```

We can observe a boy with blue shirt sitting near a building beside some sculpture but no output matches with it.


```
a group of person be stand in a street
Beam Search with K = 2: a group of child be play soccer in front of a building
Beam Search with K = 3: a group of person walk down a street
Beam Search with K = 5: a group of person walk down a street
```

We can observe a group of people walking on street and person sitting on small motor cycle .The caption group of person matches but it cant detect the man with motorcycle correctly.




```
group of kid be play in firework
Beam Search with K = 2: group of kid be stand in front of the sun
Beam Search with K = 3: group of kid be stand in front of large tree
Beam Search with K = 5: group of kid be pose for picture
```

We can observe the image a group of people pushing each other but the captions given by system are wrong only group is detected.


```
a little girl in a pink tutu be climb a rock wall
Beam Search with K = 2: a little girl in a white dress be blow bubble in a wooded area
Beam Search with K = 3: a little girl in an orange swimsuit be climb a rock wall
Beam Search with K = 5: a little girl in a yellow dress be climb a rock wall
```

We can observe in the image a girl in yellow dress tring to kicki the dinosaur.Girl with yellow dress is predicted correctly.

Conclusion

- Language bias means that the learning based on captions can be modelled easily as compared to the images

- If we assume that there are 100 images in the dataset in which the images contain men walking on street and there is only one image that there is men riding on motorcycle.

- So if we consider the men riding motor cycle case then there high chance of probability that the image detected is men walking.

-So features regarding men are given then the chance be detected as walking.
-This is the cause for Language bias in LSTM .So, No architectural changes are involved