Semantic Image description using Deep Learning

Bhargav Kulkarni, Rohith Nallamadi, Pravallika Tallapragada, Abhishree Thittenamane

December 27, 2017

1 Abstract

In this project, we have worked on the problem of Image description. Deep learning is being used extensively in the field of Image procecssing and Computer Vision, especially for handwritten character recognition and classification problems. In recent years, there has been a resurgence of interest in the use of various variants of Convolutional neural network for this purpose, mainly because of simplicity of the method, generalization of independent of number of training data. Our project mainly consists of two stages: Features from image classification network and Recurrent Neural Network based image captioning network. The Image classification features are extracted from CNN based Inception architecture. The last stage is an Image captioning problem, wherein we make use of RNN based LSTM units to train on the labelled images from the MS COCO dataset. Each word of the description will be automatically aligned to different objects of the input image when it is generated. Experimental results showcase and represent graphically as well as with images, the various methods we have used in our model.

2 Introduction

In the past few years, deep neural network has made significant progress in the areas of image classification, object detection and so on. The main aim in such image understanding tasks is deep image understanding such that whole image scenario is understood and not just individual objects. Image captioning follows this path which requires describing the content of an image using properly formed English sentences. Image captioning is the task of generating text descriptions of images. This is a quickly-growing research area in computer vision, suggesting more intelligence of the machine than mere classification or detection. Describing the contents of an image is a challenging task that involves Machine Learning and Natural Language Processing (NLP). Image caption generation is a fundamental problem that involves Computer Vision, Natural Language Processing(NLP), and Machine Learning. It can be analogous to translating an image to proper sentences. A description must capture not only the objects contained in an image, but it should also express how these objects relate to each other as well as their attributes and the activities they are involved in. Moreover, the above semantic knowledge has to be expressed in a natural language like English, which means that a language model is needed in addition to visual understanding.

RNNs use encoder decoder architecture for sequence to sequence learning. RNNs are variants of the Neural Networks that make predictions based on the sequence of inputs. For every current sequence, the outputs from previous elements are used as inputs, along with the new data. This gives a sort of memory to the network, which makes it more effective in attaining informative and context related captions.

A deep convolutional neural network (CNN) is used to produce a rich representation of the input image by embedding it to a fixed length vector. So a CNN is used as an image encoder by pretraining it for classification task and then using its last hidden layer as the input to the RNN. This inception feature vector is then given to an RNN decoder as an input and it generates sentences. This model is called the Neural Image Caption, or NIC.

LSTM is a special type of RNN which solves the vanishing gardient problem by introduceing something called the memory cell. However, inspite of having memeory cells which have accumulate a history of information, they are still limited to several time steps.

3 Literature Survey

Natural language descriptors has been a long studied topic but the problem of image captioning in natural language is something which is being studied with more interest in recent times. Recent advances in object detection and recognition and attribute recognition has been used to drive natural language generation systems but these are limited in their expressivity.

Ross Girshick et al [1]proposed a simple and scalable detection algorithm for object detection that improved mean average precision (mAP) significantly. High-capacity CNNs were used to bottom up region proposals and localize and segment objects. Hence, this method is called Regions with CNN features (R-CNN). When there is less labeled training data is available, supervised pre-training and domain-specific fine-tuning, gives a high increase in performance.

It is very time consuming to learn to store information over extended time intervals via recurrent backpropagation. This is mostly because of insufficient, decaying error backflow. In 1997, Sepp Hochreiter and Jurgen Schmidhuber came up with a novel, efficient, gradient-based method called Long Short-Term Memory" (LSTM) that solves this problem [2]. The gradient is truncated when harmless to do so and by doing this, LSTM learns to bridge minimal time lags by enforcing constant error flow. Multiplicative gate units learn to open and close access to the constant error flow. LSTM is a computationally efficient method, leads to many more successful runs, and learns much faster.
Zhongliang Yang et al [3] uses an object detection and localization model to extract information about the objects and their spatial relationship respectively. For sentence generation they use a Recurrent Neural Network based LSTM with attention mechanism similar to that of the human visual system. We follow a similar method for implementing image captioning by using LSTM as described in the sections below. We make use of a CNN-RNN model to represent an image as a single feature vector which is the output of the top layer of a pre trained convolutional network as used in [4]. Junhua Mao et al [5] propose a multimodal RNN called M-RNN where the two subnetworks one for images and the other for sentences interact in a multimodal layer to form the whole M-RNN model.

4 Model

The literature shows that by maximizing the probability of the correct translation of words in the input sentences, it is possible to attain an accurate model for image captioning. This is possible by making use of Recurrent Neural Networks (RNNs)[6] which converts the variable length input to a fixed dimensional vector and uses this representation to decode to the desired output sentence.

Maximizing the probability of the correct description given the image is done using the following formulation :

$$\Theta^* = argmax \sum_{(I,S)} log(p(S|I,\Theta))$$

where θ are the parameters of our model, I is an image, and S its correct transcription. Since S represents any sentence, its length is unbounded. Applying chain rule to model the joint probability over S0, . . . , SN , where N is the length of this particular example gives:

The dependency on θ is dropped. Stochastic Gradient Descent is used to optimize the sum of the log probabilities over the training set of the form (S, I). The probabilities p(St—I, S0, . . . , St1) is modelled with a Recurrent Neural Network (RNN), where the variable number of words we condition upon up to t1 is expressed by a fixed length hidden state or memory ht. This memory is updated after seeing a new input xt by using a nonlinear function $f : h_{t+1} = f(h_t, x_t)$. The design choice parameters for RNN includes the exact form of f and the way images and words are given as input xt. For f a Long-Short Term Memory (LSTM) net is used, which has shown state-of-the art performance on sequence tasks such as translation. The representation of images is done using CNNs popular for image tasks such as object recognition and detection. CNNs are capable of producing rich representation of input images in the form of feature vectors as seen in [7]

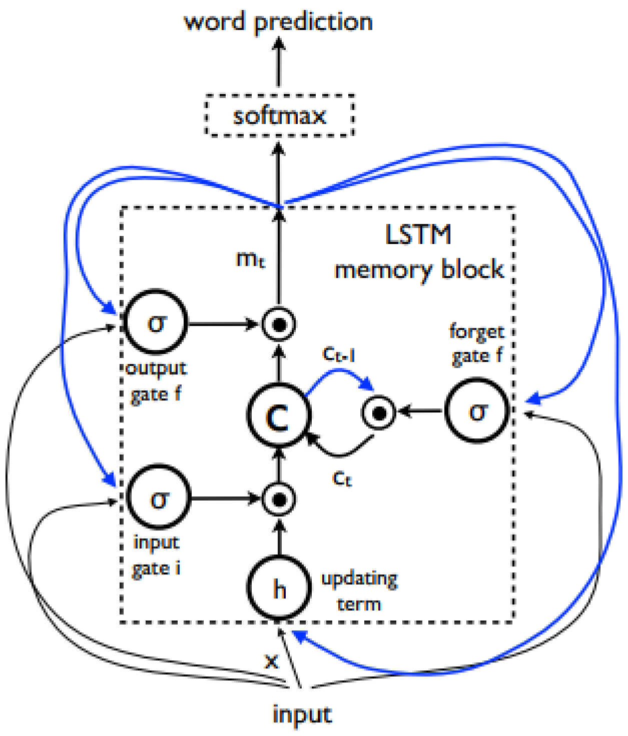


Figure 1: LSTM: The memory block contaning cell c controlled by 3 gates, Source: [4]

4.1 Inception based feature extraction from CNN

Inception based feature extraction from CNN Inception network has an advantage of computational efficiency over other networks. So, it is feasible to use inception networks in big-data where huge amounts of data needs to be processed at reasonable cost and in cases where memory or computational capacity is inherently limited. In case of inception v2, 7 7 convolution is reduced into three 3 3 convolutions.The inception network consists of 3 traditional inception modules at the 3535 with 288 filters each. This is reduced to a 17 17 grid with 768 filters using the grid reduction technique by reducing d*d to (d/2)*(d/2). This is followed by 5 instances of the factorized inception modules. This is reduced to a 8 8 1280 grid with the grid reduction technique depicted. The coarsest 8 8 level, again has 2 inception modules with a concatenated output filter bank size of 2048 for each tile. The layout of the inception v2 network is given below:

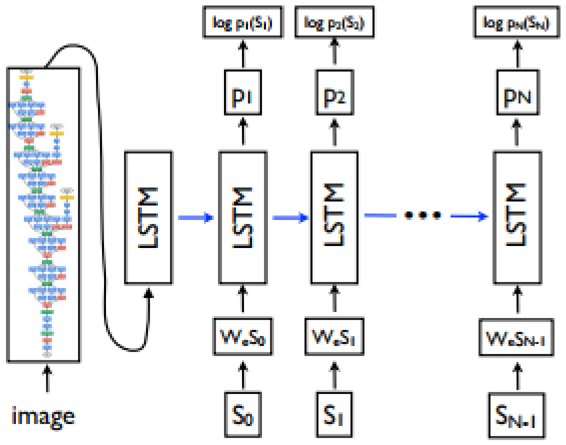


Figure 2: LSTM model combined with a CNN image embedder, Source: [4]

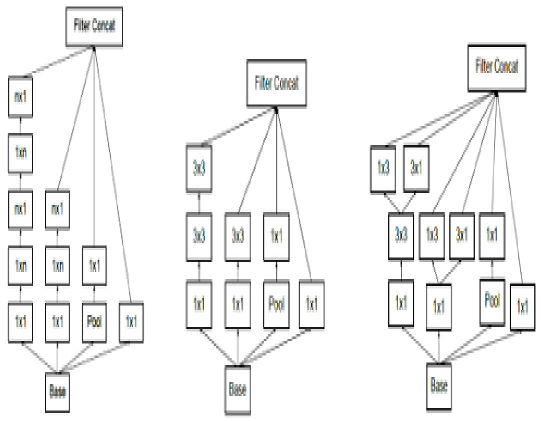


Figure 3: (From left to Right):Inception modules with expanded filter bank outputs(Source:[8]), Inception model after the factorization of the nxn convolution(Source:[8]), Inception modules with expanded filter bank outputs (Sources:[8])

type	patch size/stride	input size
conv	3x3/2	299x299x3
conv	3x3/1	149x149x32
conv padded	3x3/1	147x147x32
pool	3x3/2	147x147x64
conv	3x3/1	73x73x64
conv	3x3/2	71x71x80
conv	3x3/1	35x35x192
3xInception	As in Figure 3	35x35x288
5xInception	As in Figure 4	17x17x768
2xInception	As in Figure 5	8x8x1280
pool	8x8	8x8x2048
linear	logits	1x1x2048
softmax	classifier	1x1x1000

Inception-v3 is a variant of Inception-v2 which adds BN-auxiliary layer. BN auxiliary refers to the version in which the fully connected layer of the auxiliary classifier is also-normalized, not just convolutions. The model Inception-v2 + BN-auxiliary layer is called Inception-v3.

4.2 LSTM-based Sentence Generator

The most common problem in designing and training RNNs is vanishing and exploding gradients and the choice of f is governed by its ability to deal with it. LSTM which is a form of recurrent nets overcomes this problem [2]. The core of the LSTM model is a memory cell c containing knowledge at every time step of what inputs have been observed until that step. The behavior of the cell is controlled by gates layers which are applied multiplicatively. If the gate is 1, the gates keep a value from the gated layer and the value is dropped if the gate is 0. Three gates are generally used which control whether to forget the current cell value (forget gate f), if it should read its input (input gate i) and whether to output the new cell value (output gate o). The definition of the gates and cell update and output are as follows:

$$i_t = \sigma(W_{ix}.x_t + W_{im}.m_{t-1})$$

$$f_t = \sigma(W_{fx}x_t + W_{fm}m_{t-1})$$

$$o_t = \sigma(W_{ox}x_t + W_{om}m_{t-1})$$

$$c_t = f_t \bigodot c_{t-1} + i_t \bigodot h(W_{cx}x_t + W_{cm}m_{t-1})$$

$$m_t = o_t \bigodot c_t$$

$$p_{t+1} = Softmax(m_t)$$

where various W matrices are trained parameters. The problem of vanishing and exploding gradients is dealt well with these multiplicative gates and it helps to train the LSTM robustly. Sigmoid $\sigma()$ and hyperbolic tangent h() are non linear. The softmax layer finally produces a probability distribution pt over all words.

5 Dataset

To train the deep learning models, Microsoft Common Objects in Context(COCO) dataset has been used. The dataset has 118287 training images and 5000 validation images split across the following super categories: outdoor, food, indoor, appliance, sports, person, animal, vehicle, furniture, accessory.

electronic, kitchen. It is interesting to note that some of the super categories will have an overlap. For example, animals are present in the outdoor which are included in animals and outdoor super categories. Due to computational power limitations, in this implementation of caption generation of images, only super categories indoor and animals are considered which has 17 subcategories('sheep', 'hair drier', 'horse', 'cow', 'clock', 'dog', 'bear', 'cat', 'toothbrush', 'book', 'teddy bear', 'giraffe', 'zebra', 'elephant', 'scissors', 'vase', 'bird'). This filtered dataset has training images of 30000 and validation images of 2000.

5.1 Training

Training of LSTM model is done to predict each word of the sentence after it has seen the image as well as all preceding words given by $p(S_t|I, S_0, ..., S_{t1})$. Copy of the LSTM memory is created for the image and each sentence word such that all LSTMs share the same parameters. The output m_{t1} of the LSTM at time t 1 is fed to the LSTM at time t. All recurrent connections are transformed to feed-forward connections. For example, Let I be the input image and let $S = (S_0, ..., S_N)$ be a true sentence describing this image, then:

$$x_{-1} = CNN(I)$$

$$x_t = W_e S_t, t \in 0, ..., N - 1$$

$$p_{t+1} = LSTM(x_t), t \in 0, ..., N - 1$$

where each word is as a vector St of dimension equal to the size of the dictionary. The start word S_0 and stop word SN are assigned separately to denote the start and end of the sentence. The LSTM signals that a complete sentence has been generated by outputting the stop word. The image is mapped using a vision CNN and words are mapped to the same space by using word embedding We. The image I is only input once, at t = 1, to inform the LSTM about the image contents. Feeding the image at each time step as an extra input yields inferior results, as the network can explicitly exploit noise in the image and overfits more easily. Loss function is given by the sum of the negative log likelihood of the correct word at each step:

$$L(I, S) = -\sum_{t=0}^{N} log(p(S_t))$$

$$logp(S|I) = -\sum_{t=0}^{N} log(p(S_t|I, S_0, S_1, ...S_{t-1})).$$

This loss is minimized w.r.t. all the parameters of the LSTM, the top layer of the image embedder CNN and word embeddings We.

Implementation: Our training implementation involves two steps listed below.

- Extract inception features from the last but one fully connected layer which is a 2048 feature vector. These features are extracted for all the training and validation data.
- Pass the extracted features as the initial state to the LSTM unit. From the next time step, the LSTM unit receives a batch size of word vector. The network unrolls as many times as the number of time steps present in the input caption vector which is 20 in our case. If the size of caption of any of the image is less than 20, then zeros are appended to the vector.

To pass the word vectors to the LSTM, the words are first transformed to integer space using an word to integer mapping by assigning an unique integer to each word. Additionally, since the language data is very sparse, the integer vector is converted in to an embedding space, a standard approach used in natural language processing. This is done using Tensorflow function.

5.2 Inference

Multiple approaches can be used to generate a sentence given an image. We use sampling based inference technique. In sampling, the first word is sampled according to p1, then the corresponding embedding is taken as input to the next state. On the other hand, in beam search we keep track of k best captions until time t and at time t+1 generate a new k best set of captions.

6 Evaluation and Results

To evaluate the model, it is difficult judge whether a caption is considered right or wrong since the description can be formulated in multiple ways. The most reliable thing to do is have the human evaluators decide whether the caption is right or wrong. Although, there are some automatic metrics to evaluate the results. The most popular one is the BLEU(Bilingual Evaluation Understudy). BLEU uses a modified form of precision to compare a candidate translation against multiple reference translations. We achieved a BLEU-1 score of 64.8 percent. The authors in the [4] have reported BLEU of 66 percent which is achieved using beam search for inference. In our implementation, we used Sampling based inference technique.

The results obtained as a result of LSTM network are shown. We trained our LSTM network with 512 hidden units for 10 epochs with about 50000 examples per each epoch. A vector of 2048 image features was used for each of these examples and we ran our code for a total of 60000 iterations using Stochastic gradient descent Optimizer. The trained model then randomly picks some images for validation and saves them for every 3000 iterations. Below are some images that are picked from each of these files for displaying. We have tried to pick images from three categories: Images and captions perfectly match, Captions are somewhat related to the images, Images and captions do not have correlation.

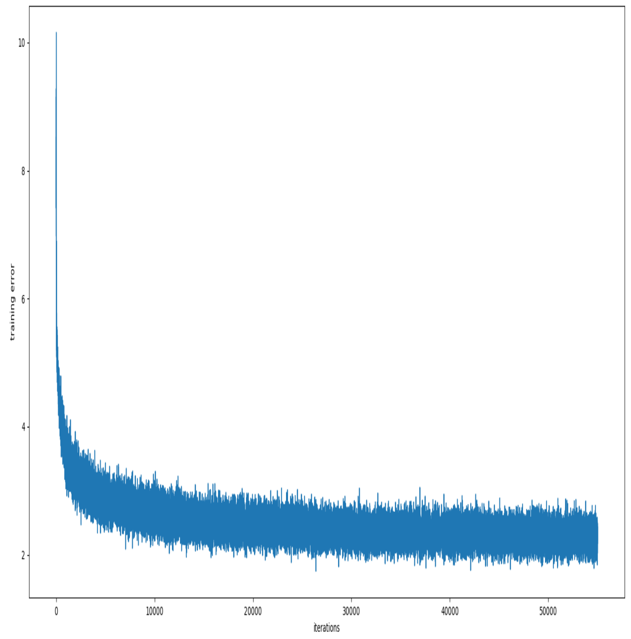
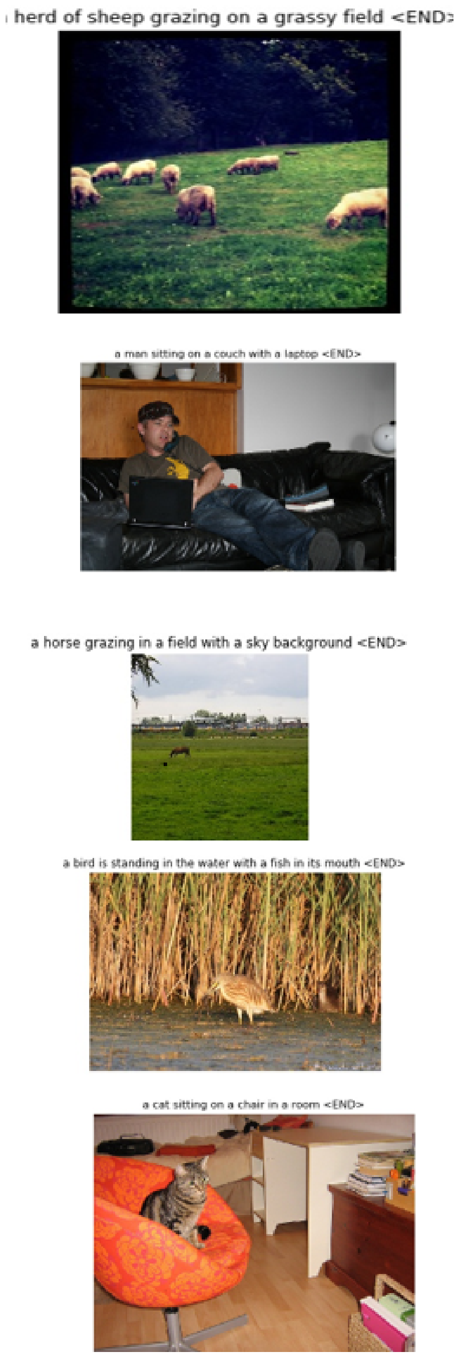
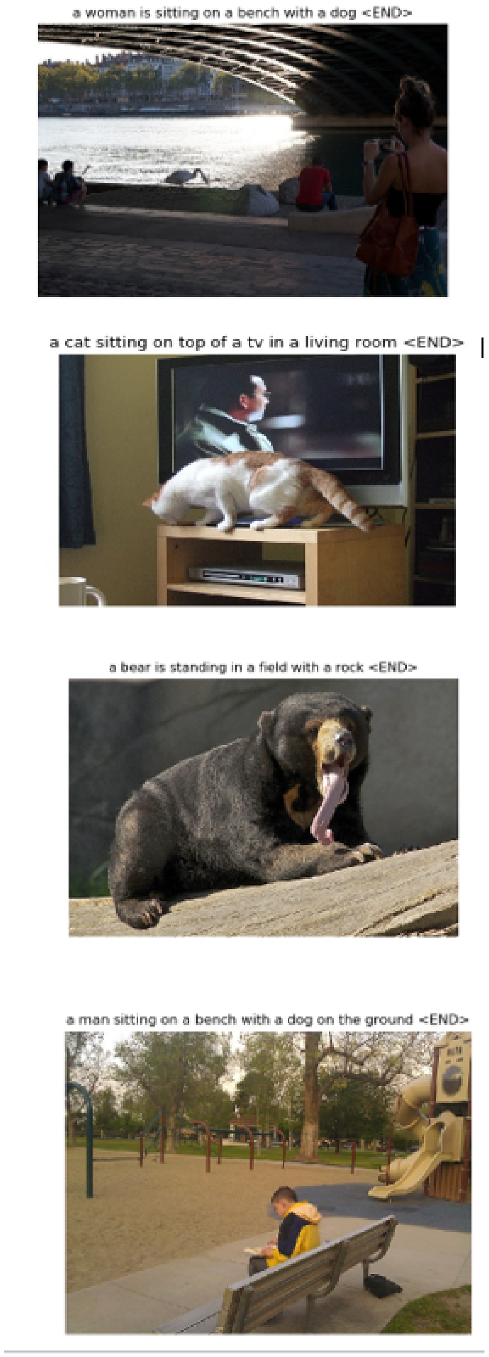


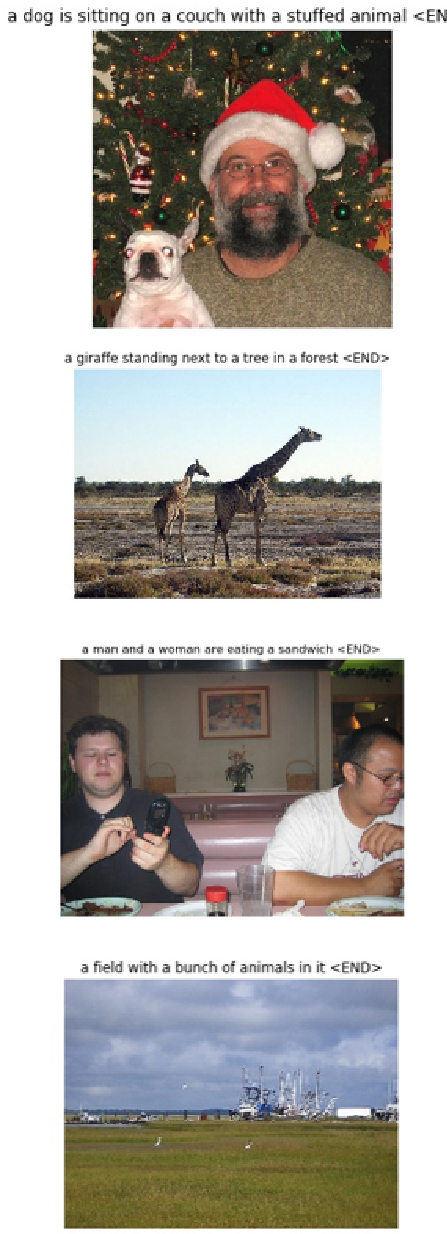
Figure 4: The above plot shows the training loss curve against the number of iterations


herd of sheep grazing on a grassy field <END>
a man sitting on a couch with a laptop <END>
a horse grazing in a field with a sky background <END>
a bird is standing in the water with a fish in its mouth <END>
a cat sitting on a chair in a room <END>

(a) fig i


a woman is sitting on a bench with a dog <END>
a cat sitting on top of a tv in a living room <END>
a bear is standing in a field with a rock <END>
a man sitting on a bench with a dog on the ground <END>

(b) fig ii


a dog is sitting on a couch with a stuffed animal <EN
a giraffe standing next to a tree in a forest <END>
a man and a woman are eating a sandwich <END>
a field with a bunch of animals in it <END>

(c) fig iii

Figure 5: The first column (i) represents images that have been captioned very well. The second column (ii) represents images that have have some relation with the captions. The third column (iii) represents images that are have no correlation between objects and the captions

7 Conclusion

We have presented an end to end deep neural network that can automatically view an image and generate a description. We were able to successfully implement image captioning and generate natural sentences as captions that were meaningful. The results are demonstrated on MS COCO dataset. Model performance can further be improved by using beam search instead of sampling for inference. We can further improve the accuracy and make the model perform better by using ResNet instead of Inception v3 for image classification.

8 References

References

[1] Ross Girshick, Jeff Donahue, Trevor Darrell and Jitendra Malik,
"Rich feature hierarchies for accurate object detection and semantic segmentation, arXiv:1311.2524v4 [cs.CV] 9 Jun 2014.

[2] S. Hochreiter and J. Schmidhuber, Long short-term memory,
in Neural Computation, vol. 9, no. 8, 1997.

[3] Zhongliang Yang, Yu-Jin Zhang, Sadaqat ur Rehman, Yongfeng Huang, Image Captioning with Object Detection and Localization, https://arxiv.org/pdf/1706.02430.pdf.

[4] Oriol Vinyals, Alexander Toshev, Samy Bengio, Dumitru Erhan,
"Show and Tell: A Neural Image Caption Generator, arXiv:1411.4555v2 [cs.CV] 20 Apr 2015.

[5] Junhua Mao, Wei Xu, Yi Yang, Jiang Wang, Zhiheng Huang and Alan Yuille, Deep Captioning with Multimodal Recurrent Neural Networks (M-RNN),,
in arXiv:1412.6632v5 [cs.CV] 11 Jun 2015.

[6] K. Cho, B. van Merrienboer, C. Gulcehre, F. Bougares, H. Schwenk,and Y. Bengio, Learning phrase representations using RNN encoder-decoder for statistical machine translation, EMNLP,2014.

[7] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. Le-Cun, Overfeat: Integrated recognition, localization and detection using convolutional networks, arXiv preprint arXiv:1312.6229,2013.

[8] Christian Szedegy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, Zbigniew Wojna, Rethinking the computer architecture for computer vision, arXiv preprint arXiv: 1512.005676v3 [cs.cv], 11 Dec 2015.