Image Captioning with Attention

Nikita Bhargava
New York University
nb2643@nyu.edu

Anshu Tomar
New York University
at3769@nyu.edu

December 22, 2018

Abstract

Image Captioning is an interesting problem existing between the intersection of Computer Vision and Natural Language Processing. In this paper, we explore combinations of deep Convolution Neural Networks (CNN) and Recurrent Neural Networks (RNN) to generate well-formed captions for Microsoft COCO dataset of images. The CNN encodes the image into feature vectors and the RNN decodes these features into texts describing the image. We have incorporated the Attention mechanism into our models and proved that Attention is useful in labelling images more efficiently

1 Introduction

A human is capable of describing an immense amount of details about a visual scene just after having a quick glance at it. However, when we talk about visual recognition models in computer vision, it has always proven to be an impalpable task.

This field is vast and there has been tremendous amount of work done in this field. Some of the applications of automated image captioning are generating summary/essay from images for visually impaired people, social media like Facebook can infer directly from the images about the whereabouts of people in the image, pairing up similar images in huge corpus with the help of their captions etc. Researchers have mostly focused on labeling images with a fixed set of visual categories and a significant amount of progress has been done so far in that area. But such models rely on hard coded visual concepts and sentence templates which puts a restriction on their variety. They cannot compete with the vast amount of rich descriptions a human is capable of making.

A common architecture for automated image captioning consists of an encoder (Convolutional Neural Network (CNN)) which encodes image to extract features from them and a decoder (Recurrent Neural Network (RNN)) which uses these features and image description to generate captions for the images. In this project, we aimed to enhance the capabilities of such models. Specifically we aimed to make these models rich enough so that they can better reason about the contents of images in addition to just identifying objects in the image. We explored the attention mechanism and applied it to our decoder network. This made the model attend a specific part of the image at a time while generating the caption for the image sequentially. This type of attention is referred as self-attention. Also, before integrating attention, we compared different combination of CNNs and RNNs to choose the best combination for our dataset from all of them. The combinations that we experimented with are - ResNet and LSTM, ResNet and GRU, VGG and LSTM, VGG and GRU. Based on the results from these runs, we chose ResNet and GRU combination to add attention to it. This paper shows the performance analysis of all these configurations.

2 Related Work

There has been a lot of work done around automatic image captioning. [1] presents a Multimodal Recurrent Neural Network (m-RNN) model for generating sentence descriptions to explain the content of images. Their model

contains two sub networks - a Convolutional Network for image encoding and a Recurrent Neural Network for sentence generation and can be applied to the task of retrieving images or sentences from a given image. [2] presents a generative model approach on a recurrent architecture. It combines recent advances in the field of computer vision and machine translation and uses it to generate natural sentences describing an image. The paper explains experiments done on several datasets and shows their accuracy in describing an image. [3] paper presents a novel approach of using multiple instance learning to train visual detectors for words occurring commonly in captions, including many different parts of speech such as nouns, verbs, and adjectives. [4] is one of the recent works in this field. This paper integrates attention mechanism along with standard back propagation techniques and shows through visualization how the model is able to automatically focus on certain parts of the image while generating the corresponding words in the output sequence.

3 Data

We have used Microsoft COCO (Common Object in Context) dataset for training and evaluation of the model [5]. The dataset contains images to train and validate the mode. It also contains annotations for the images. There are 82,784 images in the training set and 40,505 images in the validation set. For each image in the dataset, there are five captions. After downloading the dataset, the images were preprocessed to make them of equal size. This was required for feeding them to the CNN while training and evaluation of the model. Another part of the preprocessing involved building the vocabulary from captions of the images present in the dataset. This vocabulary contains words which are present more than a particular threshold in the whole dataset. A number is assigned to each word that would be used by the CNN and RNN in the model. The reason to choose the COCO dataset is that this dataset set is rich in different categories of images and is proven to train the CNNs and RNNs models better than any other existing datasets.

4 Model

There are three major components to our final model. They are discussed in brief below with some illustrations [6]

4.1 Encoder

An encoder encodes the input image into a smaller image with learned channels. Convolutional Neural Networks perform amazingly well in encoding the images. As there are already trained CNNs represent images, we chose RenNet-152 and VGG-11 to experiment with. These networks are trained on ImageNet and available in PyTorch. Figure 1 is a overview of how an Encoder works.

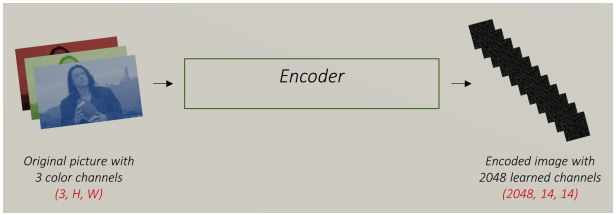

Figure 1: Encoder

4.2 Decoder

A decoder looks at the encoded image and generate caption for the image word by word. A decoder needs to be a recurrent neural network. There are already trained RNNs to generate sequences. We chose LSTM and GRU to experiment with. These networks are already present in PyTorch. Figure 2 is a overview of how an Decoder with Attention works.

4.3 Attention

With attention, the decoder will look at different parts of the image at different time steps to generate words in the sequence. By adding attention, the weighted average of the encoded image is passed through the decoder at every time step. This weighted representation of the image is concatenated with the previously generated word to generate the next word in the sequence. The attention network calculates these weights for the image. The part of

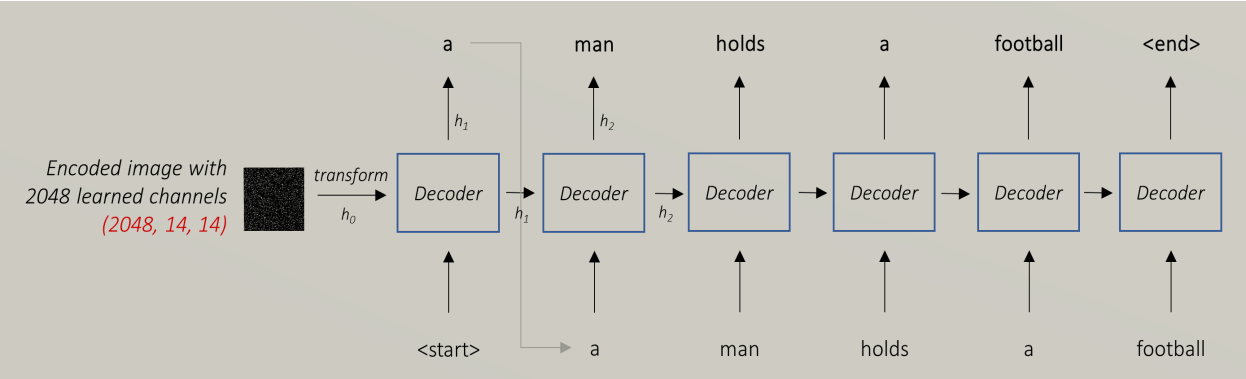

Figure 2: Top Level Model of how a Decoder works on Images

the image with higher weights will be the focus of the network to generate the next word in the sequence. Figure 3 is a overview of how an Decoder with Attention works.

5 Experimentation

5.1 Initial Model

For our initial model, we chose an architecture with ResNet acting as a convolutional network followed by LSTM as the Recurrent Neural Network [ref: yunjey]. The input images were passed through a ResNet-152 model. This was Pytorchs pre-trained ResNet-152 model. But before using it, we removed the last fully connected layer from it . The feature vectors obtained were further worked upon by a fully connected layer which converted them to the desired embedding size. These embedded vectors were then passed to the LSTM working in a many to many mode. At each time-step, a new token was provided and a word from the vocabulary predicted. The output of the LSTM was then passed through a fully connected layer, output of which was a well-formed caption for the image. Figure x shows how the training loss reduced with time. For the evaluation of the model, we observed the minimum loss from the training logs and took the model corresponding to that. The best encoder and decoder ckpt files along with the training logs can be found at [ref:]. This location also has the generated captions for this model for the Microsoft COCO dataset.

5.2 Model Exploration

In order to determine the best model for our image dataset, we ran exploratory experiments across the other popular CNN i.e the VGG model. We used the VGG-11 model and ran experiments with both the Decoders - LSTM and GRU. We observed that the results were better when we used ResNet-152 and hence our final model has ResNet-152 as the Encoder.

GRUs are a popular alternative to LSTMs as they are less complex than LSTMs on account of having 2 gates instead of 3 and thus can train faster. For our RNN network we decided to explore GRUs in addition to LSTMs. As we have trained our models on Prince, we can not validate whether the GRU model trained faster than the LSTM model but post training both models till 30 epochs, we saw that the loss was lesser for the GRU based models as compared to the LSTM based models. More is explored regarding this in Section 6: Evaluation and Results.

Figure 4 and Figure 5 shows how the training loss reduced with time for two of the above described models. For the evaluation of the model, we observed the minimum loss from the training logs and took the model corresponding to that. The best encoder and decoder ckpt files along with the training logs can be found at [ref:] for each of the models]. This location also has the captions generated by each model for the Microsoft COCO dataset.

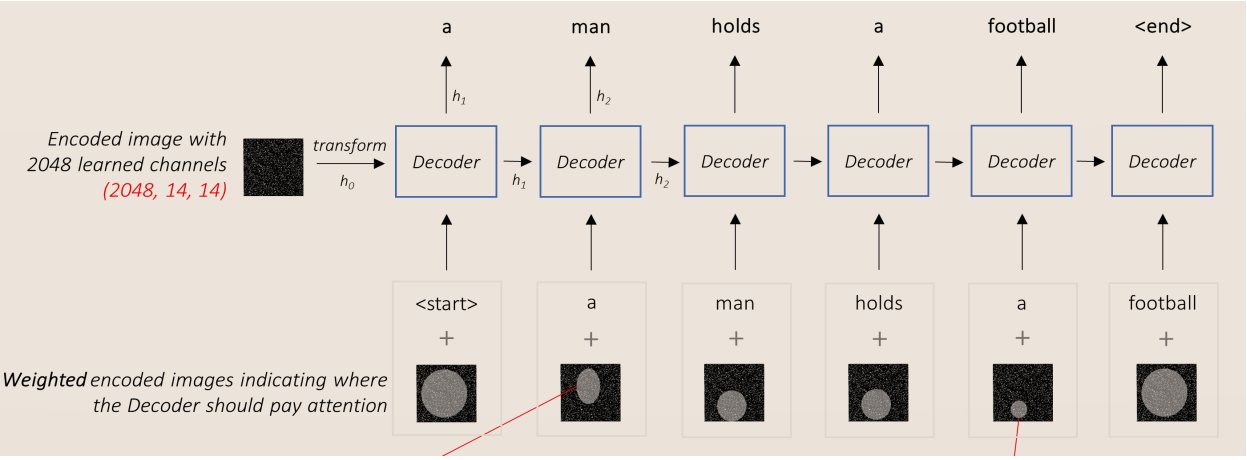

Figure 3: Top Level Model of how a Decoder with Attention works on Images

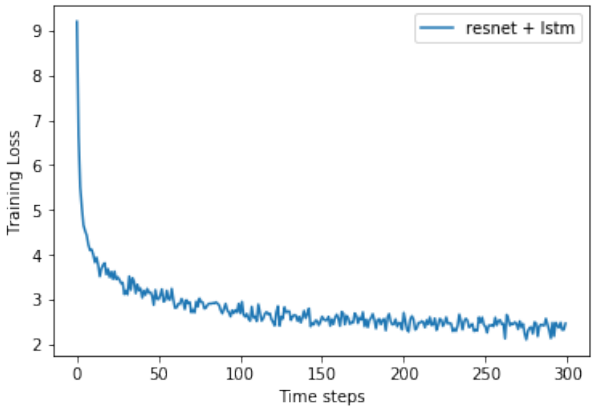

Figure 4: Training Loss for ResNet and LSTM model

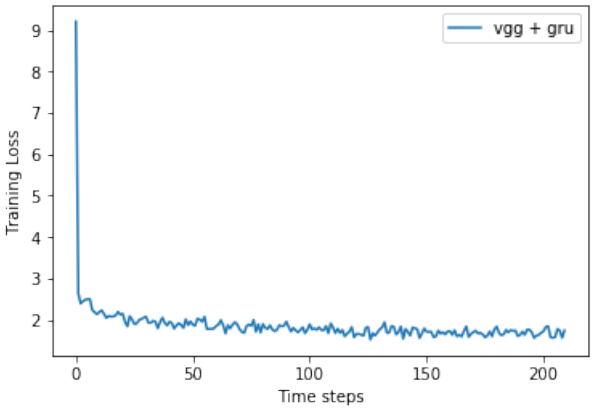

Figure 5: Training Loss for VGG and GRU model

6 Evaluation and Results

6.1 Comparative Study between Models

We evaluated the models on different types of scores. Following are the scores which we used for our evaluation [6]:

BLEU (Bilingual Evaluation Understudy) Score: This score works by counting matching n-grams in the candidate translation to n-grams in the reference text, where 1-gram or unigram would be each token and a bigram comparison would be each word pair. The comparison is made regardless of word order. Depending on the number of tokens , the BLEU score is called as BLEU_1(unigrams), BLEU_2(biagrams), BLEU_3 (trigrams) and BLEU_4(4grams) [8]

ROGUE (Recall-Oriented Understudy for Gisting Evaluation) Score: ROUGE tries to assess the adequacy, by simply counting how many n-grams in your generated summary matches the n-grams in your reference summary.There are different types of ROUGE scores as

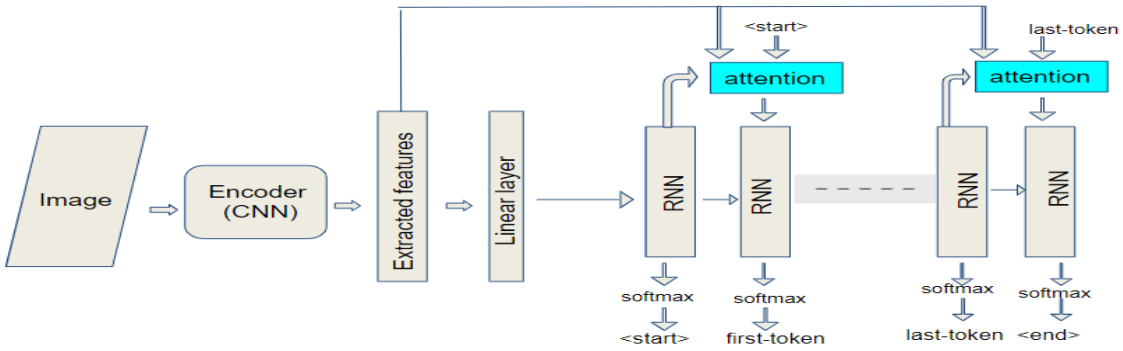

Figure 6: Top Level Representation of Our Final Model

well such as ROUGE-1(unigrams), ROUGE-2(bigrams), ROUGE-L(Longest Common Subsequence) and many more.

CIDer(Consensus-based Image Description Evaluation) Score: This metric measures the similarity of a generated sentence against a set of ground truth sentences written by humans and it shows high agreement with consensus as assessed by humans. Using sentence similarity, the notions of grammaticality, saliency, importance and accuracy (precision and recall) are inherently captured by this metric. [9]

SPICE(Semantic Propositional Image Caption Evaluation) Score: The creators of this metric hypothesize that semantic propositional content is an important component of human caption evaluation. They also claim that SPICE captures human judgments over model-generated captions better than other automatic metrics. [10]

Based on our experiments, we saw that ResNet152 as Encoder and GRU as Decoder was giving the best performance even before we added our Attention Layer to the model. Post adding our Attention Layer, performance improved on all scores. Also almost for all model runs we observed that as compared to LSTM networks, GRU networks trained significantly faster as was expected as they are less complex on account of having one gate lesser. In fact, of all the models we experimented with **ResNet-152 as Encoder** and **GRU as Decoder with Attention** performed best on all metrics. Table 1 shows the scores from the most successful runs of our main models.

6.2 Analysis of the Final Model

6.2.1 Some Captions Generated by our Final Model

Figure 7 to 11 are three of the results from our Final Model (with Decoder Attention). Specifically above are the results from our ResNet + GRU + Attention model. We can see that even in images with lot of content, the model captured the main essence well.

Figure 7: Generated Caption: ”a horse standing in a grassy field with a horse in the background.”

MODELS	BLEU_1	BLEU_2	BLEU_3	BLEU_4	ROUGE_L	CIDer	SPICE
ResNet + LSTM	0.658	0.477	0.335	0.235	0.488	0.758	0.157
VGG + LSTM	0.610	0.423	0.285	0.193	0.452	0.588	0.128
ResNet + GRU	0.662	0.481	0.337	0.234	0.490	0.760	0.158
VGG + GRU	0.610	0.425	0.289	0.199	0.456	0.600	0.129
ResNet + GRU + Attn	0.664	0.482	0.340	0.240	0.491	0.770	0.159

Table 1: Evaluation Scores For Main Models

Figure 8: Generated Caption: ”a man dives for a frisbee on a beach.”

Figure 9: Generated Caption: ”a remote control sitting on top of a table.”

6.2.2 Comparison With Initial Model

In Figure 11, we can see a comparison of captions generated by the Initial Model (specifically ResNet + LSTM) and the final Attention Model (specifically ResNet + GRU + Attention). The caption generated by the final model correct and it is much more descriptive.

Figure 10: Generated Caption: ”a plane flying through a cloudy sky above a city.”

6.2.3 Some Non-satisfactory Results

We can see that in the Figure 9, though the model identified the object (remote control), it failed to output a well-formed description. We saw that our model used the sitting verb for a lot of objects. In Figure 10 as it can be seen - the model captioned the image wrong. There were 3 objects in the air and it labelled it as just one flying plane.

References

[1] Explain Images with Multimodal Recurrent Neural Networks, https://arxiv.org/pdf/1410.1090.pdf

[2] Show and Tell: A Neural Image Caption Generator,

Figure 11: Initial Model : "a group of people standing around a horse drawn sled."
Final Model (With Attention) : "a black and white photo of a group of people with tennis rackets."

https://arxiv.org/pdf/1411.4555.pdf

[3] From Captions to Visual Concepts and Back, https://arxiv.org/pdf/1411.4952.pdf

[4] Show, Attend and Tell: Neural Image Caption Generation with Visual Attention, https://arxiv.org/abs/1502.03044

[5] Microsoft COCO: Common Objects in Context, https://arxiv.org/pdf/1405.0312.pdf

[6] Github for the Dataset, https://github.com/tylin/coco-caption

[7] https://github.com/sgrvinod/a-PyTorch-Tutorial-to-Image-Captioning

[8] A Gentle Introduction to Calculating the BLEU Score for Text in Python, https://machinelearningmastery.com/calculate-bleu-score-for-text-python/

[9] https://www.cv-foundation.org/openaccess/content-cvpr_2015/papers/Vedantam_CIDEr_Consensus-Based_Image_2015_CVPR_paper.pdf

[10] SPICE: Semantic Propositional Image Caption Evaluation, https://arxiv.org/abs/1607.08822