Image Captioning

Vishal Motwani
vpm238@nyu.edu

Vikram Mullachery
mv333@nyu.edu

Abstract

Here we discuss and demonstrate the outcomes from experimentation on Image Captioning. Image captioning is a much more involved task than image recognition or classification, because of the additional tasks of recognizing the interdependence between the objects/concepts in the image and the creation of a succinct sentential narration.

Experiments on several datasets show the accuracy of the model and the fluency of the language it learns solely from image descriptions. With a handful of modifications, three (3) of our models were able to perform better than the baseline model by A. Karpathy. The best BLEU and CIDEr scores that we achieved at 28.1% and 0.848 compare favorably to the baseline model's 26.8% and 0.803, on MSCOCO dataset

Related Work

Recent work in this area includes Show and Tell by O. Vinyals et. al.[1], Show Attend and Tell: Neural Image Caption Generation with Visual Attention by K. Xu et. al.[2]. There have been other more recent implementations that extend the use of Deep ResNet, for e.g. by R. Lou[3]. A highly educational work in this area was by A. Karpathy et. al., Neuraltalk2.

More recent advancements in this area include Review Network for caption generation by Zhilin Yang et al[4]. and Boosting Image Captioning with attributes by Ting Yao[5]

Datasets

We use three (3) different datasets to train and evaluate our models. These datasets contain real life images and each image in these datasets is annotated with 5 captions

MSCOCO[6]	contains 120K images with 5 captions for each Split : 80k images for Training and 40k images for Validation
Flickr30k[7]	contains 30K images with 5 captions each Split : 28K images for Training and 2k images for validation
Flickr8k[8]	8K images with 5 captions each Split : 7k images for training

[1] Show and Tell: A Neural Image Caption Generator. O. Vinyals et. al.
[2] Show Attend and Tell: K.Xu et. al.
[3] Neuraltalk2 using Deep ResNet by R. Luo
[4] Review Networks for Caption Generation. Yang et. al.
[5] Boosting Image Captioning with Attributes. T. Yao et. al.
[6] MSCOCO image captioning dataset
[7] Flickr30K dataset: From image descriptions to visual denotations
[8] Flickr8K dataset: Framing image description as a ranking task

	and 1k images for validation

General Architecture

The goal is to maximize the probability of the correct description given the image by using the following formulation:

$$\theta^* = argmax_\theta \sum_{I,S} logP(S|I;\theta)$$

Figure 1: Objective function 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. Thus, it is common to apply the chain rule to model the joint probability over $S_0, \ldots, S_N$ where N is the length of this particular example as

$$logP(S|I;\theta) = \sum_{t=0}^{N} log(S_t|I, S_0, S_1 \ldots S_{t-1};\theta)$$

Figure 2: Modeling Sentence Probability

At training time, (S, I) is a training example pair, and we optimize the sum of the log probabilities as described in Figure (2) over the whole training set using adam optimizer. It is natural to model $P(S_t \mid I, S_0, ., ., S_{t-1})$ with a Recurrent Neural Network (RNN), where the variable number of words we condition upon up to t – 1 is expressed by a fixed length hidden state or memory h_t. This memory is updated after seeing a new input x_t by using a nonlinear function $f : h_{t+1} = f(h_t, x_t)$

For f we use a Long-Short Term Memory (LSTM) net. For the representation of images, we use a Convolutional Neural Network (CNN). They have been widely used and studied for image tasks, and are currently state-of-the art for object recognition and detection.

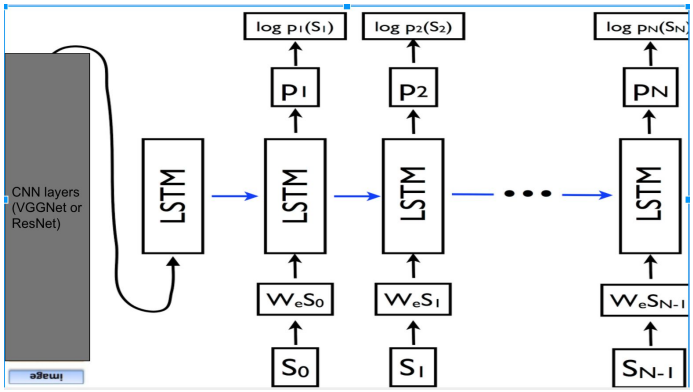


Figure 3: LSTM decoder combined with CNN image encoder. The unrolled connections between the LSTM memories are in blue and they correspond to the recurrent connections. All LSTMs share the same parameters

The LSTM model is trained to predict each word of the sentence after it has seen the image as well as all preceding words as defined by $P(S_t \mid I, S_0, ., ., S_{t-1})$. For this purpose, it is instructive to think of the LSTM in unrolled form – a copy of the LSTM memory is created for the image and each sentence word such that all LSTMs share the same parameters and the output $m_t - 1$ of the LSTM at time $t - 1$ is fed to the LSTM at time t (see Figure 3). All recurrent connections are transformed to feed-forward connections in the unrolled version. In more detail, if we denote by I the input image and by $S = S_0, \ldots, S_N$ a true sentence describing this image, the unrolling procedure reads:

$$\begin{aligned} x_{-1} &= \text{CNN}(I) \\ x_t &= W_e S_t, \quad t \in \{0 \ldots N-1\} \\ p_{t+1} &= \text{LSTM}(x_t), \quad t \in \{0 \ldots N-1\} \end{aligned}$$

where we represent each word as a one-hot vector S_t of dimension equal to the size of the dictionary. Note that we denote by S_0 a special start word and by S_N a special stop word which

designates the start and end of the sentence. In particular by emitting the stop word the LSTM signals that a complete sentence has been generated. Both the image and the words are mapped to the same space, the image by using a vision CNN, the words by using word embedding W_e. The image I is only input once, at $t = -1$, to inform the LSTM about the image contents.

The loss function is as follows :

$$L(I, S) = -\sum_{t=1}^{N} log\, p_t(S_t).$$

The above loss is minimized w.r.t. all the parameters of the LSTM, the top layer of the image embedder CNN and word embedding W_e.

Further, to generate sentence, beam search is used. It iteratively consider 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. This approximates $S = argmax_S P(S' | I)$. We use beam size of 20 in all our experiments.

Baseline Model

We use A. Karpathy's pretrained model as our baseline model. This model is trained only on MSCOCO dataset. The model uses a 16-layer VGG Net for embedding image features which is fed only to the first time step of the single layer RNN which is constituted of long-short term memory units (LSTM). The RNN size in this case is 512. Since words are one hot encoded, the word embedding size and the vocabulary size is also 512.

The two parts, CNN and RNN, are joined together by an intermediate feature expander, that feeds the output from the CNN into the RNN. The feature expander allows the extracted image features to be fed in as an input to multiple captions for that image, without having to recompute the CNN output for a particular image.

In VGG-Net, the convolutional layers are interspersed with maxpool layers and finally there are three fully connected layers and softmax. The softmax layer is required so that the VGGNet can eventually perform an image classification. But for the purpose of image captioning, we are interested in a vector representation of the image and not its classification. And so, the last two layers are eliminated and the output from the fully connected layer can be extracted and expanded to feed into the RNN part of the architecture. The code and and further details of the baseline model are explained here

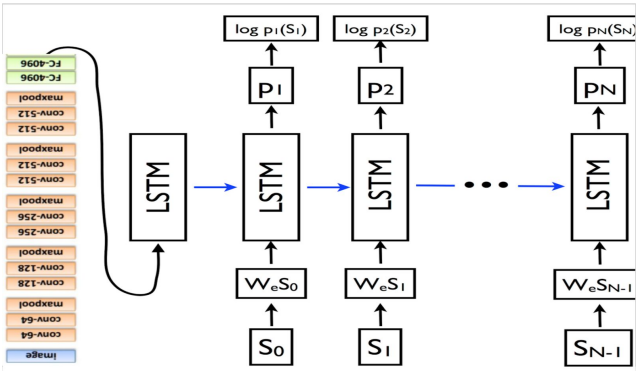


Figure 4: Baseline Model

Experiments and Modifications

We attempted three different types of improvisations over the baseline model using certain variants to the architecture.

Added Training on Flickr8k and Flickr30k

First improvisation was to use further training of the pretrained baseline model[9] on Flickr8K and Flickr30k datasets. After building a model identical to the baseline model, we initialized the weights of our model with the weights of the

[9] Downloadable Baseline Model (A. Karpathy)

baseline model and additionally trained it on Flickr 8k and Flickr 30K datasets, thus giving us two models separate from our baseline model

Increased RNN hidden layers

Second improvisation was increasing the number of RNN hidden layers over the baseline model. When we add more hidden layers to the RNN architecture, we can no longer start our training by initializing our model using the weights obtained from the baseline model (since it consists of just 1 hidden layer in RNN architecture). Hence in this case we pre-initialize the weights of only the CNN architecture i.e VGGNet by using the weights obtained from deploying the same 16 layer VGGNet on an ImageNet classification task. Thus using this method, we were able to increase the number of hidden layers in the RNN architecture to two (2) and four (4) layers.

Using ResNet in place of VGGNet

The third improvisation was to use ResNet in place of VGGNet. Our Motivation to replace VGG Net with Residual Net (ResNet) comes from the results of the annual Imagenet classification task. Following are the results for the imagenet classification task over the years

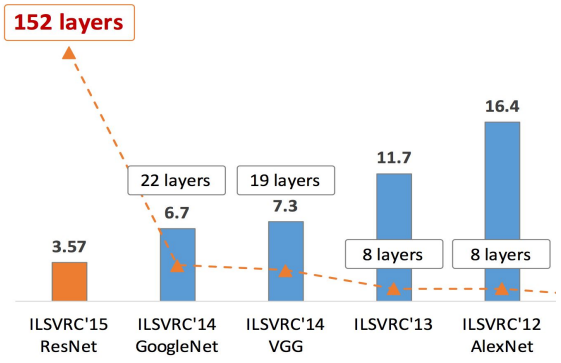


Figure 5: Image Classification improvements over the years

It is obvious from these results that ResNet (or Residual Net) can encode better image features.

ResNet architecture is a 100 to 200 layer deep CNN. To account for the problem of vanishing gradients, ResNet has the following scheme of skip connections.

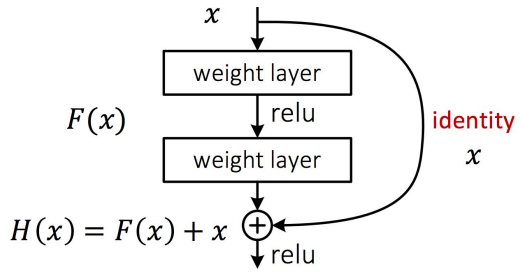


Figure 6: Skip Connections in ResNet

Inspired from the results of ResNet on Image Classification task, we swap out the VGGNet in the baseline model with the hope of capturing better image embeddings.

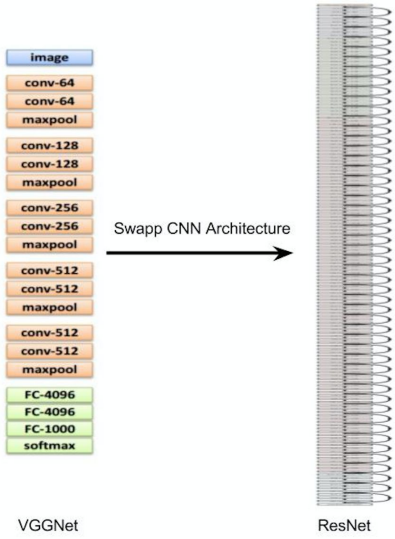


Figure 7: Swapping VGGNet with ResNet

We use 101 layer deep ResNet for our experiments.

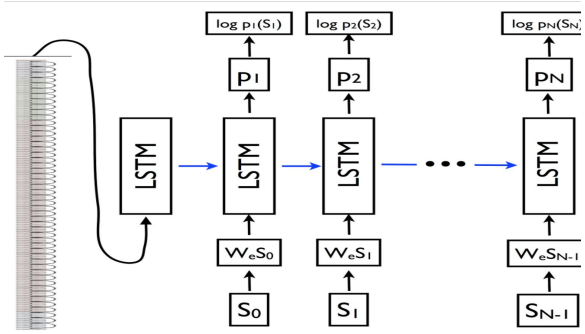


Figure 8: Architecture using ResNet for Image encoding

We pre initialize the weights of only the CNN architecture i.e ResNet by using the weights obtained from deploying the same ResNet on an ImageNet classification task[10]. Note that there are no changes to the RNN portion of the architecture for this experimentation choice.

Evaluation Metrics

There are two evaluation metrics of interest to us.

First, a caption language evaluation score, BLEU_4 score[11] (bilingual evaluation understudy), which is an algorithm for evaluating the quality of text which has been machine-translated from one natural language to another. It ranges from 0 to 1, with 1 being the best score, approximating a human translation.

Second, CIDEr score[12], which is a consensus-based evaluation protocol for image description evaluation, which enables an objective comparison of machine generation approaches based on their "human-likeness", without having to make arbitrary calls on weighing content, grammar, saliency, etc. with respect to each other. This score is usually expressed as a percentage or a fraction, with 100% indicating human generated caption for an image.

Results

Following is a listing of the models that we experimented on:

Baseline	PreTrained model - A.Karpathy's work (Trained only on MSCOCO)
Model 1	Additional Training of Baseline on Flickr8k
Model 2	Additional Training of Baseline on Flickr30k
Model 3	VGGNet 16-layer with 2 layer RNN (Trained ONLY on MSCOCO)
Model 4	VGGNet 16-layer with 4 layer RNN (Trained ONLY on MSCOCO)
Model 5	ResNet 101-layer with 1 layer RNN (Trained ONLY on MSCOCO)

Following are a few key hyperparameters that we retained across various models. These could be helpful for attempting to reproduce our results.

RNN Size	512
Batch size	16
Learning Rate	4e-4
Learning Rate Decay	50% every 50000 iterations
RNN Sequence max length	16
Dropout in RNN	50%

[10] Downloadable ResNet for Image Classification
[11] BLEU
[12] CIDEr: Consensus-based Image Description Evaluation

Gradient clip	0.1

Following are the results in terms of BLEU_4 scores and CIDEr scores of the various models on the different datasets.

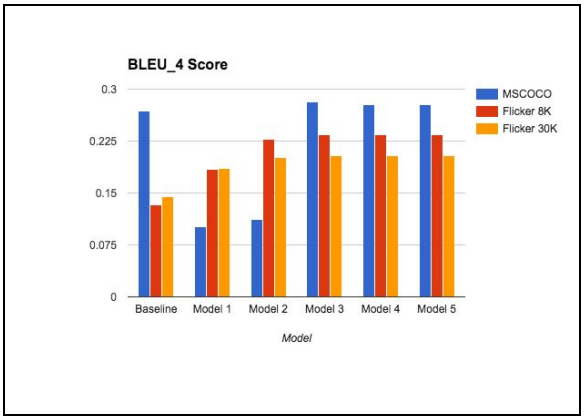


Figure 9 – BLEU_4 Score

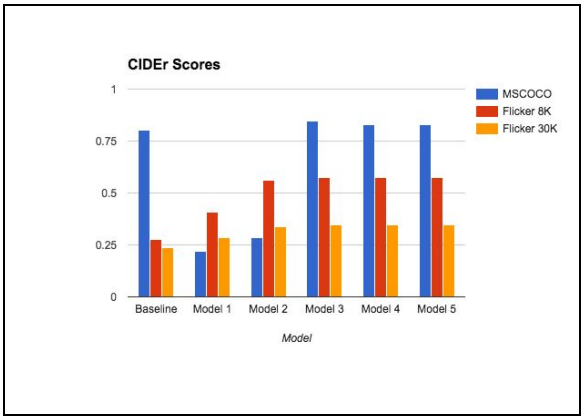


Figure 10 – CIDEr Score

Following graph shows the drop in cross entropy loss against the training iterations for VGGNet + 2 RNN model (Model 3). Note that each iteration corresponds to one batch of input images. In our experiments, Model 3 outperformed all the other models.

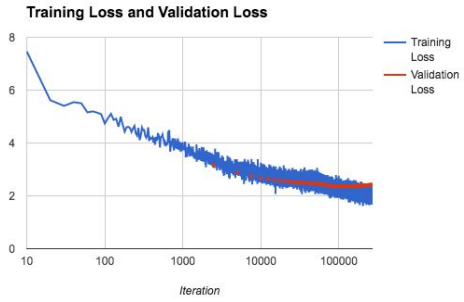


Figure 11 – Learning Rate for Model 3 (VGGNet with 2 layer RNN)

Discussion of a few results

A few instances of correct captions:

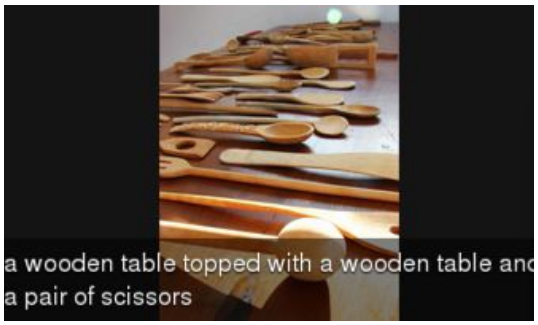
Note that this is not a copy of any training image caption, but a novel caption generated by the system.

Similar to the above, this a novel caption, demonstrating the ability of the system to generalize and learn

Figure 7 – Good Captions

Hypotheses on incorrect captions:

High co-occurrences of "cake" and "knife" in training data and 0 occurrences of "cake" and "spoon", thus engendering this caption



High occurrences of "wooden" with "table", and then further with "scissors".

Zero occurrences of word "wooden" with the word "utensils" in training data.

Further, this caption shows vulnerability of the model in that the caption could be nonsensical to a human evaluator

Figure 8 – Poor Captions

From the above examples, it is clear that the model has a bias to the words that it has seen together previously. This could have been possibly resolved by the attention mechanism in the work Show, Attend and Tell[2]

An Application - Video Captioning

As an experimentation to apply video captioning in real-time we loaded a saved checkpoint of our model and generated a caption of the video frame. Following are some interesting results

Instances of correct correct video captions	Correct Video Captions
Instances of incorrect/poor video captions	Incorrect captions

Entertaining as some of the above maybe, they teach us a few valuable things about video captioning being different from static image captioning. Empirically, one observes that there are abrupt changes in captions from one frame to the next. However, intuitively and experientially one might assume the captions to only change slowly from one frame to another. This disconnect would suggest feeding the caption from one frame as an input to the subsequent frame during prediction.

Additionally, the current video captioning sways widely from one caption to another with very little change in camera positioning or angle. This rapid change in caption appears to be akin to a highly sensitive decoder. This demonstrates a dearth of inertia or recognition of the image source as a video from a camera (as opposed to disconnected slides of individual images). Consequently, this would suggest the necessity to stabilize/regularize the caption from one frame to the next.

A third item to watch out for is the apparent unrelated and arbitrary captions on fast camera pannings. Since this is an expected real-life action on a camera, there will need to be, as yet unexplored, adjustments and accommodations made to the prediction method/model.

Future work

We observe that ResNet is definitely capable of encoding better feature vector for images. Also, taking tips from the current state of art, i.e show attend and tell, it should be of interest to observe the results that could be obtained from applying attention mechanism on ResNet.

For the decoder we currently do not use the dense embedding of words. Also, we do not initialize the weights of RNN architecture from the weights of a pre trained language model. Though Vinyals et al. mention that they do not observe any significant gain by pre-training the RNN language model, it should be of interest to observe if it's the same scenario when used in conjunction with ResNet.

Ensembling have long been known to be a very simple yet effective way to improve performance of machine learning systems. In the context of deep architectures, one only needs to train separately multiple models on the same task, potentially varying some of the training conditions, and aggregating their answers at inference time. This is another effort that should be worth pursuing in future work.

Acknowledgement

K. Simonyan and A. Zisserman. Visual Geometry Group. Very Deep Convolutional Networks for Large-Scale Visual Recognition.

A. Karpathy Neural Talk2 serving as our baseline model.

C. Olah. Chris Olah's blogs on Neural Networks. http://colah.github.io/

Bibliography

1. Oriol Vinyals, Alexander Toshev, Samy Bengio, Dumitru Erhan. Show and Tell: A Neural Image Caption Generator. April 2015
2. Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, Yoshua Bengio. Show Attend and Tell: Neural Image Caption Generation with Visual Attention. April 2016
3. Review Networks for Caption Generation. Yang et. al.
4. Boosting Image Captioning with Attributes. T. Yao et. al.
5. Image Captioning with attention . Blaine et al.
6. Deep Residual Learning Kaiming et al.
7. Very Deep Convolutional Neural Network for large scale image generation Karen Simonyan et al.