Image Captioning using Deep Learning

Abhisek Konar
McGill ID: 260779907
abhisek.konar@mail.mcgill.ca

Aashima Singh
McGill ID: 260782301
aashima.singh@mail.mcgill.ca

Abstract— **The aim of this project is to provide automatic meaningful description of images. Image classification and object recognition have been relevant trends in deep learning and computer vision community. It is proposed here to take the task one step further by providing semantic descriptions to images from the Flickr8k dataset by using combinations of Convolutional neural networks and various flavors of recurrent neural networks. In doing so, niceties of multiple deep neural network architectures such as LSTM, Gated Recurrent Unit and Bidirectional LSTM have been explored. The results obtained are then compared to give a comprehensive analysis of the model architectures that are best suited to give state-of-the-art results.**

I. INTRODUCTION

With the advent of large scale deep neural networks, there has been a flurry of work done in the domain of Computer Vision. CNNs and the availability of large annotated datasets of images like the IMAGENET[14], along with the enhancement in computation capabilities of our computers, has somewhat created an ideal environment which has lead to a lot of breakthrough in the vision domain. Complex neural net architectures has be created that can match the accuracy of humans when it comes to recognizing objects given an image(top 5 error rate in the Imagenet Large Scale Visual Recognition Challenge)[15]. A natural, and a lot more challenging extension to the task of image recognition is the task of image captioning. Image captioning is the task of generating a sentence given an image, that meaningfully describes it. It involves not only recognizing the objects in the image given, but also arranging words in a proper syntax based on the spatial description of the image given. Many end to end image captioning models has been proposed [1],[7]. From these it can be said that in general terms, a model consists of three parts. An image model that deals with the image information, a language model that captures the relationships between the words present in the captions and, a third part, which consists of a multimodal decoder which combines the information from the above two models to predict the caption for a given image.
In this project different architectures has been explored and comparative study has been made involving them. This project concentrates on 2 parts of the model.

1) *Image encoding part*
   For this part experiments with different image representations are done. In particular, the two CNN architectures, VGG19 and RESNET152 are used to encode the images. As training these nets is very computationally exhaustive, weights of both the architectures pretrained on IMAGENET have been used.
2) *Multimodal decoder*
   The different architectures experimented with in the multi-modal decoder architecture are:
   a) LSTM
   b) Bidirectional LSTM
   c) Gated Recurrent Unit
   d) Multi-layer LSTM

II. RELATED WORKS

In recent developments in image captioning tasks, myriad approaches have been undertaken such as the model containing multi-modal Recurrent Neural Network, introduced by Karpathy et al. [7]. It successfully aligned visual and language modalities through a common multi-modal embedding. Vinyals et al. [1] also took an end-to-end neural approach with a Convolutional Neural Network followed by RNN was used. Xu et al. [8] created a different attention-based image captioning model which automatically learns image content and gave state-of-the-art results when considering evaluation metrics. The datasets that have mostly been used in image captioning are the MSCOCO, Flickr30k and Flickr8k datasets.

III. MODEL DESCRIPTION

The algorithms and architectures used are as follows:

A. VGG19

The VGG19 model was introduced by Simonyan and Zisserman [2] and involved the usage of very small 3×3 convolutional filters and very deep layers of weights of up to 19. A set of 3 smaller 3×3 convolutional filters were used instead of a bigger 7×7 window because this greatly reduced the number of the convolution weights from ${}^{49}C_2$ to ${}^{27}C_2$. Also, it was found that a set of 3 smaller 3×3 convolutional windows provided a similar receptive field performance to that of the bigger windows. This configuration also increases the non linearity within the architecture. The VGG19 has a 19 layer deep architecture consisting of 5 pooling layers. In the pooling layers, spatial pooling is carried out where MAX pooling is done over a 2×2 pixel window with a stride of 2.
At the end of the 19 weight layers are 3 fully connected dense layers. The first two layers take up input dimensions of size 4096 and the final layer performs a 1000 way classification which is then fed to a final softmax layer to produce the final class of the input image. In all the fully connected layers

present, the ReLU is the choice of the activation function used.

B. ResNet152

The ResNet152 was an architecture introduced by He et al.[3] and it consists of weight layers of a depth of 152, that is, 8 times deeper than the VGG neural network, yet it has lesser complexity and fewer number of filters. Like the VGG19, it uses 3×3 small convolutional filters, however it adds shortcut connections which converts the original network into a residual network. A batch normalization layer is used after each convolution layer. For the pooling layer, it uses a 3×3 window with a stride of 2. The weight layers of the Resnet finally connects to a single fully connected layer that takes a 1000 dimensional input and uses a softmax activation function to make a 1000 way classification of the input images. This architecture also takes input images of size $224 \times 224 \times 3$.

C. LSTM

The Long Short Term Memory, as denoted in Fig 1, is a special kind of Recurrent Neural Network which involves learning of long term dependencies. In cases of language models, often vanilla RNNs do not perform well in cases where long-term contexts need to be considered but LSTMs, as introduced by Hochreiter et al. [4], do not face such problems. Unlike a simple RNN containing a single layer in the repeating module, LSTM has 4 interactive layers. LSTM manipulates the information in the cell state, the core of LSTM, with the help of three gates. The first is the 'sigmoid' layer which makes the decision about what information will be forgotten and what will be updated.

$$f_t = \sigma(W_f.[h_{t-1}, x_t] + b_f) \tag{1}$$

After this there is a combination of an input sigmoid layer, with a 'tanh' layer makes the decision of what new values will be added to the cell state.

$$i_t = \sigma(W_i.[h_{t-1}, x_t] + b_i) \tag{2}$$

$$C'_t = tanh(W_c.[h_{t-1}, x_t] + b_c) \tag{3}$$

$$C_t = f_t * C_{t-1} + i_t * C'_t \tag{4}$$

The cell state is updated as shown by equation (4). Then we run a sigmoid and a tanh layer which pushes the output values between -1 and 1 and it is multiplied with the output of the sigmoid layer which finally outputs the previously decided values.

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{5}$$

$$h_t = o_t * tanh(C_t) \tag{6}$$

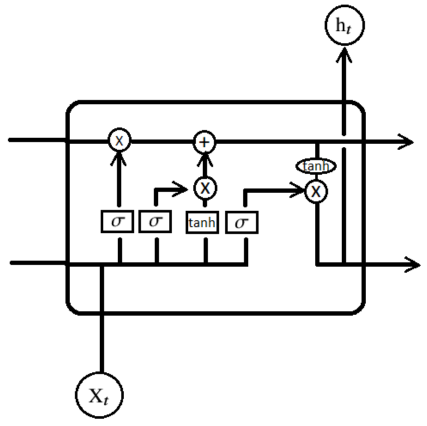

Fig. 1. Long Short Term Memory

D. Bidirectional LSTM

Bidirectional LSTM, as introduced by Schuster et al. [5], is an extension of LSTM such that there are two LSTMs and while the first LSTM is similar to the one described above, the second LSTM is applied on the reverse of input sequence. The network goes through the input sequence in both directions at the same time.

E. Gated Recurrent Unit

Gated Recurrent Unit, introduced by Cho et al. [6] and represented in Fig 2, is a relatively new form of LSTM where like the LSTM there are gates that manipulate the flow of information through the cell state but the difference is that the input and 'forget' gate layers are combined into a single update layer.

$$z_t = \sigma(W_z.[h_{t-1}, x_t]) \tag{7}$$

$$r_t = \sigma(W_r.[h_{t-1}, x_t]) \tag{8}$$

Instead of controlling the degree to which each cell state is exposed, GRU exposes the whole state each time. The *tanh* activation is similar to an LSTM.

$$h'_t = tanh(W.[r_t * h_{t-1}, x_t]) \tag{9}$$

where there is element-wise multiplication between r_t and h_{t-1} and r_t are a set of reset gates which when off, causes the unit to forget the previously computed state. The final output can be represented as follows:

$$h_t = (1 - z_t) * h(t-1) + z_t * h'_t \tag{10}$$

IV. IMPLEMENTATION DETAILS

A. Dataset Description

There are a number of publicly available image datasets that could have been used for image captioning tasks. The dataset used in the project is the Flickr8k dataset[10]. The rational behind selecting this particular dataset consists of images and their corresponding annotations, which are vital

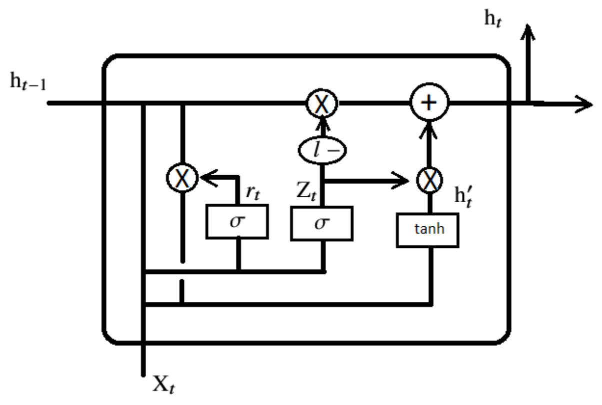

Fig. 2. Gated Recurrent Unit

for the task of image captioning. Also, compared to other available image repositories, like the MSCOCO and the Flickr30K, the Flickr8K is a relatively smaller database and so running experiments on it will be relatively faster.
The Flickr8k dataset consists of 8000 RGB images of subjects from a wide variety of categories with 4 captions describing each of the images. The images present do not have a fixed predefined size. And so before using the images, each of the images were preprocessed to be of the same shape, $224 \times 224 \times 3$. The captions given per image too as expected are of arbitrary length and necessary steps in preprocessing are taken up to make the data compatible with the captioning architecture. This is described in further details in the language model subsection.

B. Model used

The baseline model, represented in Fig 3, as inspired by the end-to-end approach taken by Vinyals et al. [1], consists of the following three components:

1) Image Model: The image model can be broken down into two parts:

1) Image encoding using CNN:
   The $224 \times 224 \times 3$ resized image obtained from preprocessing the images from the dataset is encoded using the architectures mentioned above. The procedure followed for each of the architectures are as follows:
   a) VGG19:
      The VGG19 accepts image of dimension $224 \times 224 \times 3$, so the preprocessed images are directly passed onto the CNN. Pre-trained weights of VGG19 has been used in the project to save time and computation effort. For VGG19 two different encoding strategies has been explored:
      - The last fully-connected layer is popped from the architecture and the 4096 dimensional feature vector representation of the images are obtained from penultimate layer is then used as an input to the following dense layers present in the captioning model.
      - The entire architecture is used including the final fully connected layers to obtain a 1000 dimensional feature vector containing posterior probabilities of the ILSVRC dataset classes. This is then passed on to the adjacent dense layer.

   b) RESNET152:
      Similar to the VGG19, this architecture also accepts image of dimensions $224 \times 224 \times 3$. And pre-trained weights for the RESNET are used for reasons stated above. The two different encoding strategies followed for the RESNET152 are:
      - The model up to the penultimate layer is considered, leading to a 2048 dimensional resultant image feature vector to be fed onto the subsequent dense layer.
      - The entire architecture including a global average pooling layer and a 1000 dimensional fully-connected softmax layer.
2) The Adjacent Dense layer:
   The feature vector obtained above is then passed through a dense layer with a ReLU activation function,that compacts it from a 4096 or 2048 dimensional vector in case of VGG19 or ResNet152 respectively to a 128 dimensional feature vector. This is then mapped to the same input space of the decoder module as the language model. The ReLU activation function is selected because they are easy to optimize due to their similarity with linear unit (Goodfellow et al 2016) because of their consistency in their gradient. The ReLU function has a first order derivative of 1 everywhere the unit is active.

2) The Language Model: There language model primarily deals with 2 things.The preprocessing of the image captions and converting the processed captions to a suitable 128 dimensional representation that are then fed into the final multimodal decoder.

- The Flickr8k dataset contains five possible captions for each image. The first task of the language model is to pre-process the data by tokenizing the captions such that keywords 'start' and 'stop' demarcate the beginning and ending of each caption. The key core of the language model is the LSTM (Long Short Term Memory). Hence whenever an 'end' keyword is encountered the LSTM signals the end of the caption. Thereafter, partial captions are generated by keeping in track the preceding words encountered earlier. It is padded so as to make the length equal to the largest caption. Next words are maintained with the help of one-hot encoding with the total size of the vocabulary or unique words. These processed captions are then passed down into the LSTM network. These are then passed through an embedding layer and an LSTM to generate a 128 dimensional word embedding.

- The Language model consists of two layers. The first layer used is an embedding layer. This converts the above preprocessed captions to a dense representation of 256 dimensions which is then passed through an LSTM layer combined with a time-distributed dense layer(keras implementation) to produce the final desired 128 dimensional word embeddings, which, combined with the 128 dimensional image representations are passed onto the multimodal decoder model.

3) Decoder Model: The decoder model was constructed by first merging the image and language models to obtain a 256 dimensional vector representation. Then the following approaches were used:

1) LSTM:
The LSTM is a special kind of RNN that predicts the next word after it has seen the current image and the previous words. It computes the following conditional probability as stated by Vinyals et al. [1], $P(S_t|I, S_0, S_1..., St-1)$. The output dimension from the LSTM layer is 1000 which is then followed by a dense layer that produces a 8256 dimensional output which is the vocabulary size or the number of unique words.

2) Bidirectional LSTM:
Bidirectional LSTM was implemented in Keras with the help of the Bidirectional layer wrapper that takes the original LSTM as the argument. Hence the output from the Bidirectional LSTM is 512 dimensional and it is passed through a dense layer like the previous case to get 8256 dimensional output.

3) Gated Recurrent Unit:
The GRU implemented in Keras takes the 256 dimensional merged word embedding and image model vector as input and outputs the 1000 dimensional vector which after passing through the Dense layer is converted to 8256 dimensional output.

4) Multi-layer LSTM:
When a 3 layer LSTM is used as a Decoder model then the language model does not include the LSTM and the dense and time distributed layers. Instead, the output of the embedding layer along with the image model is merged in the decoder and 3 layers of LSTM are constructed. Here several hyper-parameters were experimented with. L1 and L2 Regularization function was used and the He normal initializer was used to initialize the kernel weights matrix. The dropout rate or the fraction of units to be dropped for linear transformation of inputs was set to around 0.13. This model ultimately ends with the dense layer producing a vocabulary size dimensional output and a softmax activation.

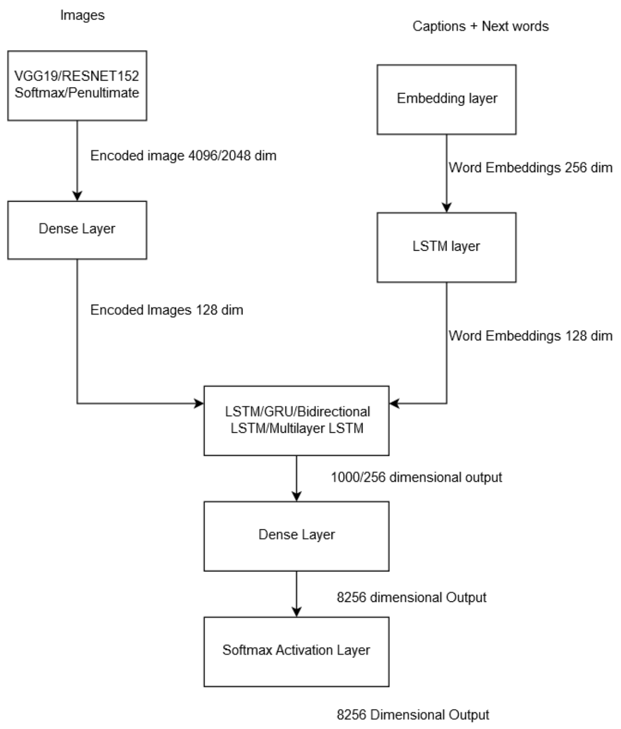

Fig. 3. The baseline model

C. Loss function and optimization

1) Loss function: The loss function used in the model is categorical cross entropy. The mathematical formula for the same is:

$$H(p,q) = -\sum_{x} p(x)log(q(x)) \tag{11}$$

The categorical cross entropy function is preferred over a mean squared error because, with the usage of mean squared loss function it might be possible that with the reduction in the loss, the weight adjustment factor shrinks leading to the stalling of the learning of the model. This can be avoided using a categorical cross entropy function.

2) Optimization function: The optimization method used in the model is RMSprop (Tieleman & Hinton et al 2012)The RMSprop is a adaptive learning rate method which maintains a uniformity across mini-batches. It is proportional to the decaying average of the square gradient. It has been found that RMSprop performs well in recurrent neural networks.

V. EXPERIMENTS AND RESULTS

A. Evaluation metrics

The evaluation metric used is nltk based corpus BLEU score which was introduced by Papineni et al. [9]. BLEU or Bilingual Evaluation Understudy evaluates quality of textual data based on the similarity between a machine translation

and human generated sentences. BLEU compares the candidate sentences with the multiple reference texts.
All the BLEU scores that has been reported as been obtained from running the captioning model trained using the predefined training set and testing on the predefined testing set. All the training sets are run for 10 epochs and a batch size of 1024. These has been chosen keeping a balance between computation time and accuracy.

B. *Different Image representations*

Representation	Architecture	B-1	B-2	B-3	B-4
Penultimate	VGG19	0.8524	0.7455	0.6410	0.5962
	ResNet152	0.8937	0.7896	0.6788	0.5889
Softmax	VGG19	0.9158	0.7864	0.6462	0.5346
	ResNet152	0.9216	0.7973	0.6652	0.5598

TABLE I
CORPUS-BLEU SCORES USING BEAM=1 FOR IMAGE REPRESENTATIONS

From the above obtained results it can be seen that considering across all the n-grams of the BLEU metrics there is no fixed trend in increase or decrease in accuracy of the model with the use of penultimate or Softmax layer in either of the image representations. But, for higher values of n (4 and 3), the removal of the softmax layer from the representation causes a significant amount of rise in the BLEU score. This can stem from the fact that image representations obtained from the penultimate layer of the architecture, especially the VGG19 (fc7 layer) produces a better generalized representation that are well suited for other tasks.

C. *Decoder Model Experimentations*

Image Representation	Decoder Model	B-4
VGG19	LSTM	0.5962
	Bidirectional LSTM	0.51519
	GRU	0.5293
	Multi-layer LSTM	0.1083
ResNet152	LSTM	0.5889
	Bidirectional LSTM	0.6106
	GRU	0.6186
	Multi-layer LSTM	0.5598

TABLE II
CORPUS-BLEU SCORES USING BEAM=1 FOR DECODER MODELS

It is observed that in Table II, GRU performs marginally better than the other decoder models. This is because GRU has a simpler structure than LSTM and is computationally efficient. Its performance is closely followed by bidirectional LSTMs which perform better than unidirectional LSTMs because they involve past and future dependencies exploitation. One reason behind multi-layer LSTM's poor performance as compared to normal LSTM, could be inappropriate hyper-parameters and the results could be improved by tweaking the hyper-parameters.

From Fig. 4 it can be seen that while the other three architectures follow a similar trend, the multilayer LSTM architecture has a slightly different loss pattern. This can be caused due to the non linearity introduced in the model with layering. This pattern can be found in both loss and validation accuracy curves as shown in Fig. 5,6 and 7.

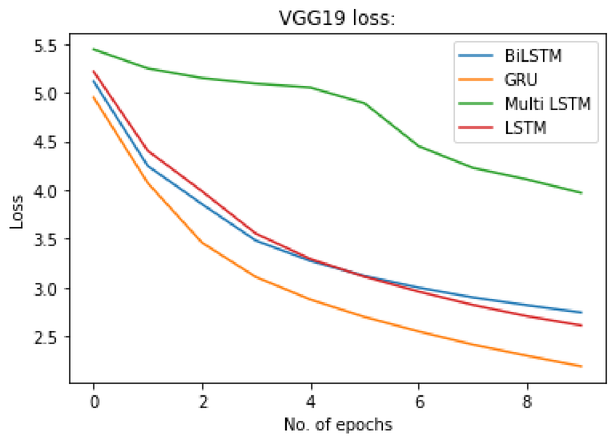

Fig. 4. Loss function for different decoder models using VGG19

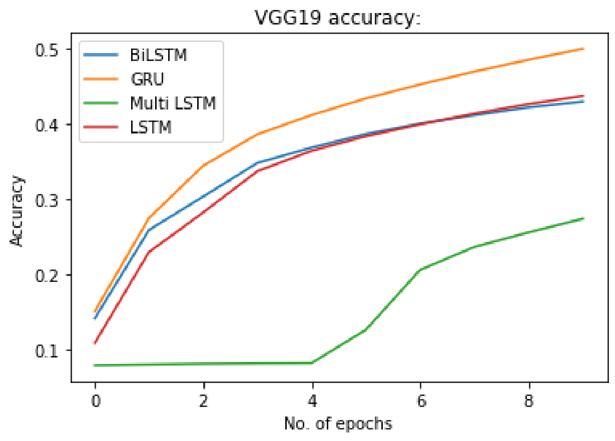

Fig. 5. Validation accuracy for different decoder models using VGG19

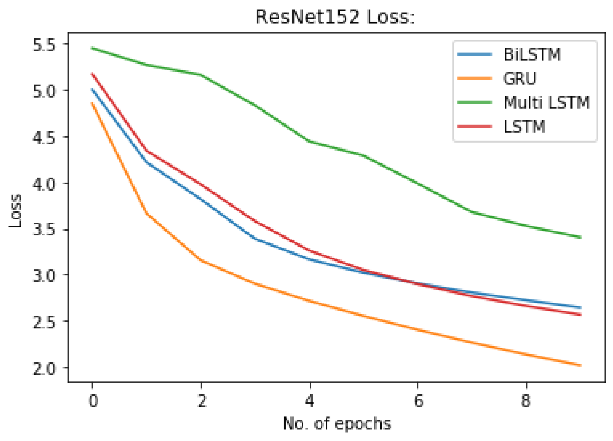

Fig. 6. Loss function for different decoder models using ResNet152

A few examples of successful image captioning can be

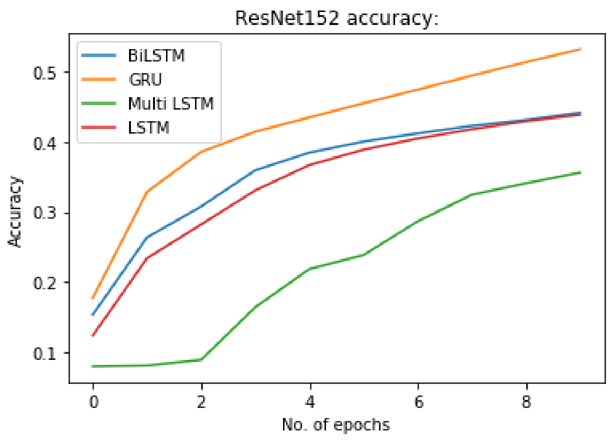

Fig. 7. Validation accuracy for different decoder models using ResNet152

observed in Fig 8.

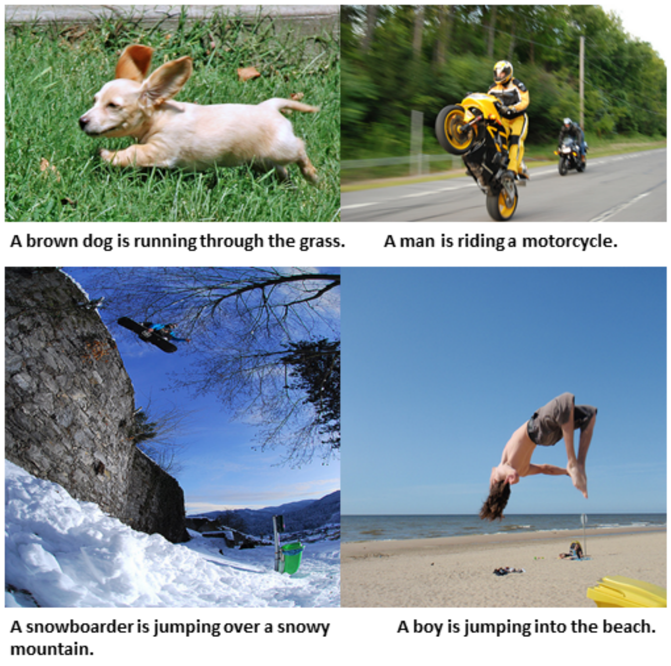
A brown dog is running through the grass.
A man is riding a motorcycle.
A snowboarder is jumping over a snowy mountain.
A boy is jumping into the beach.

Fig. 8. Successful image captioning results

VI. Discussion

From the different combinations of image representations and decoder models tried in the experiment, the combination of ResNet152 image representations paired with a GRU decoder seeem to perform the best out of the lot. This can be attributed to the fact that the ResNet152 has a deeper but simpler architecture when compared to the VGG19 enabling it to encapsulate the spatial image information better than the shallower VGG19. And so, even with image representation vectors of half the size (2048 as compared to 4096 in VGG19) the ResNet152 is able to impart more image information. The gated recurrent unit model or the

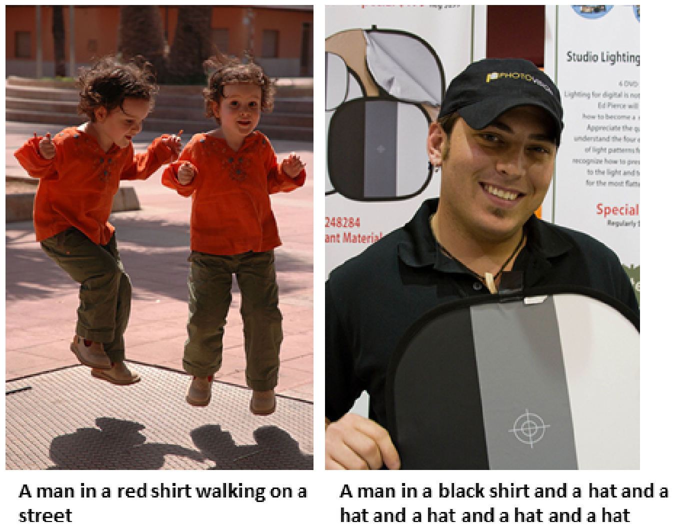
A man in a red shirt walking on a street
A man in a black shirt and a hat and a hat and a hat and a hat and a hat

Fig. 9. Slightly off-the-mark results

GRU has a simpler architecture compared to the LSTM and empirically seen to perform at par with the LSTM and it is quicker to train. In the experiments conducted, it is observed that in a GTX 1060 with a batch-size of 1024 the model using GRU took around 513 seconds per epoch to train whereas the LSTM took around 630 seconds per epoch. And on top of being faster than the LSTM, for this dataset, the GRU outperforms the LSTM, albeit by a small margin.
Among the captions that were 'bad', on inspection, can be categorized into groups, one, where the model predicts something meaningful, but what it predicts is not what is given in the image and the second, where the predicted caption does not have a meaningful sentence structure.
For the first case, an example of which is shown in Fig 5, it can be seen that although the prediction done by the model is wrong, and it is a boy jumping on the street rather than a man walking, but it can be understood that where the model has gone wrong. It might have presumed the boy to be a man, and the action of jumping on the street as walking, which probably is a more common action. This explanation by no means can be generalized for every such inaccurate predictions, but attempts to provide a perspective as to why the model is predicting what it is predicting given the image.
For the second case, the most common error that has been observed, is that the model fixates on one particular word and starts repeating it thus producing a gibberish sentence. But a careful analysis of some of those reveals that the words that gets repeated in most of the cases, describes an object found in the image, so, it can be concluded that in such cases the model is able to decode the image contents, but is unable to form a meaningful sentence out of the information it has gained. A possible solution to this kind of bad captioning can be to penalize the captions that uses too much repetition of words.
Finally, unlike object recognition, image captioning does not have a single ground truth. A particular image can be

represented in several different ways but conveying similar meaning. As a result of which, evaluating captions generated from these models becomes somewhat a tricky task in itself. A drawback in this project is that only BLEU metrics has been used to judge the credibility of the generated captions. But in a task where there is no single ground truth, a better approach to judge a caption can be done by taking a general consensus of score using different captioning metrics like CIDEr[11] , METEOR [12] , ROGUE [13] and others, in addition to the one being used.

VII. Conclusion and Future Work

In this project, different decoder models involved in image captioning were experimented with, evaluated and compared with each other. The model was based on Convolutional Neural Networks for encoding images and exploration of LSTM, Bidirectional LSTM, Gated Recurrent Units and Multi-layer LSTM for the decoder module. In future, this implementation of image captioning could be expanded into domains of video captioning or caption-based image searches. As far as image captioning is concerned, more detailed captions in a narrative manner could be generated in future.

References

[1] Vinyals, O., Toshev, A., Bengio, S., & Erhan, D. (2015). Show and tell: A neural image caption generator. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 3156-3164).

[2] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

[3] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

[4] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780.

[5] Schuster, M., & Paliwal, K. K. (1997). Bidirectional recurrent neural networks. IEEE Transactions on Signal Processing, 45(11), 2673-2681.

[6] Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555.

[7] Karpathy, A., & Fei-Fei, L. (2015). Deep visual-semantic alignments for generating image descriptions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 3128-3137).

[8] Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., ... & Bengio, Y. (2015, June). Show, attend and tell: Neural image caption generation with visual attention. In International Conference on Machine Learning (pp. 2048-2057).

[9] Papineni, K., Roukos, S., Ward, T., & Zhu, W. J. (2002, July). BLEU: a method for automatic evaluation of machine translation. In Proceedings of the 40th annual meeting on association for computational linguistics (pp. 311-318). Association for Computational Linguistics.

[10] M. Hodosh, P. Young and J. Hockenmaier (2013) "Framing Image Description as a Ranking Task: Data, Models and Evaluation Metrics", Journal of Artificial Intelligence Research, Volume 47, pages 853-899 http://www.jair.org/papers/paper3994.html

[11] Vedantam, R., Lawrence Zitnick, C., & Parikh, D. (2015). Cider: Consensus-based image description evaluation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4566-4575).

[12] Lavie, A., & Denkowski, M. J. (2009). The METEOR metric for automatic evaluation of machine translation. Machine translation, 23(2), 105-115.

[13] Lin, S. H., & Chen, B. (2010, July). A risk minimization framework for extractive speech summarization. In Proceedings of the 48th annual meeting of the Association for Computational Linguistics (pp. 79-87). Association for Computational Linguistics.

[14] Olga Russakovsky*, Jia Deng*, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg and Li Fei-Fei. (* = equal contribution) ImageNet Large Scale Visual Recognition Challenge. IJCV, 2015

[15] http://karpathy.github.io/2014/09/02/what-i-learned-from-competing-against-a-convnet-on-imagenet