Deep Learning for Image Captioning

Siddesh Pillai
srp4698@rit.edu
Under the guidance of Prof. Jeremy Brown (jsb@cs.rit.edu)
Rochester Institute of Technology

ABSTRACT

Image Captioning is an application which provides a meaningful textual summary of the scene in the image. The application is developed on the Android platform. When using the application, the user takes a picture and sends it to the server. The server has a model which generates captions for the image as a whole, which can be read out to the user via text to voice feature of Android devices. Flickr30k [36] and MSCOCO [4] image data-sets were used to develop the model. Since, its a supervised learning the data-sets comprise of images with 4 textual annotations for every image. The model was developed using Neural Talk which is a deep learning framework which performs operations such as convolutional neural network for object recognition in images and a bi-directional recurrent neural network for captioning the image [18]. Experiments were carried out to test the effectiveness of the model with respect to scale and color augmentation. These experiments provided some fascinating results on how the natural language descriptions were generated based on image characteristics. This application was developed with the intent of providing aid to visually impaired individuals.

1. INTRODUCTION

A normal human-being just needs a glance for a few seconds over an image to provide intricate details about the scene [9] whereas, a visually impaired person may require significant amount of time to understand an image, when there are lots of objects present in the image. This problem is compounded when the objects in the image are not everyday objects and are uncommon or rare. Problems also arise if the background has unnatural properties such as a blurry background or when the background is in motion. Attempts have been made in the past to empower visually challenged individuals [19] and with the advancement of Computer Vision along with the evolution of deep learning, object recognition has been on the rise. Some recent applications and devices in the field of Virtual Reality [10] and Augmented Reality [24] include: Google Goggles [11], Hound [33], Shazam [31], Hololens [25] have paved way for innovations which can bridge this gap for people with visual and hearing disability.

Currently, the applications for image scene detection are nascent and there exist many challenges since every image has different complex features and each image also depends on hardware components which are used to capture the image. For e.g. the resolution of an image captured by analog devices varies a lot from the resolution of the same image when captured by digital devices. Even in digital images, you can find quality spectrum of pixels from 500 x 480 of a Digital 8 image to 15360 x 8640 of a 16k image. Lets look at a scenario where we want to understand the context of an image. As you can see in figure 1 [21] previous models may provide captions such as, a person on a bike, which is a normal descriptions of objects in the image but it ignores other important details and characteristics, such as riding on the road or riding fast etc. This fails to take into account action features or to ignore other characteristics such as a blurry or in motion background. Hence, it doesn't quite describe the manner of the action for the image.

Image Recognition is a sub-problem of Image Processing which involves object recognition, content based image retrieval [6], optical character recognition [26], and facial recognition [1]. However, this is only the first step of our solution. Now we have a model in place which will classify all the real objects of an image but it doesn't quite gives us the context of the image such as what is the image about. Earlier experiments which aimed to generate a model capable of providing innate descriptions from the image [8] [5], where each technique is different from the other and the captioning power is enforced with restricted set of vocabulary; hence, the captions generated from these models proved to be less decisive with limited capabilities.

When we compare and contrast the 2 images as shown in

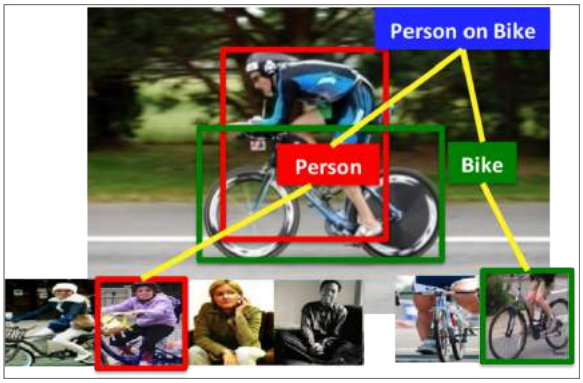

Figure 1: Labelling: Description for an image

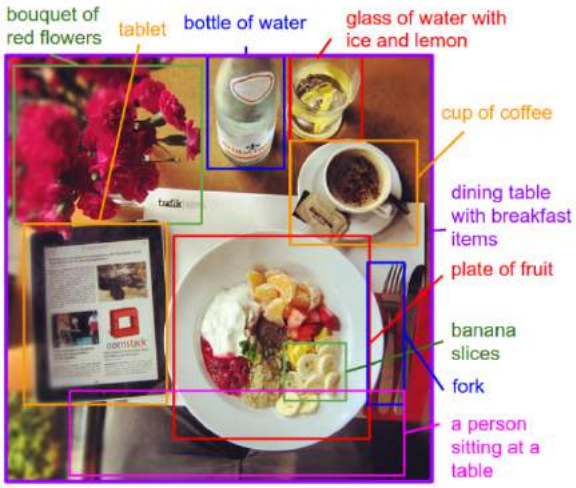

Figure 2: Goal: Rich Descriptions for an image

the figures 1 [21] and 2 [18] where both generate captions. Figure 2, however, provides us with a richer description and beats the first image in providing instinctive details about the image, by far. It is an arduous task to mimic the power of human eye but with Deep Learning there have been breakthroughs in terms of accuracy for such experiments.

The following content in this project report consists of the background of the problem, the approach used to solve the problem, system design architecture, experiments performed, results obtained, conclusion and future work.

2. BACKGROUND

As Yoshua Bengio [34] stated "Computers are now pretty good at understanding images, but in terms of understanding the meaning of a sentence, a paragraph, a document, we still have a lot of progress to make, and there's now a lot of people around the world designing new algorithms based on deep learning for exactly that". Karpathy et al. [18] stressed on the fact the object recognition is the first step but you need to have a strong label space, which is basically arranging words in a sequence to generate complete representation of the image. Alternatives to label space are using some fixed set of categorical language models which should be complete and provide natural language, using the vast amount of knowledge present on the Internet. Despite the improvements in Visual Semantic representations by Chen et al. [5], Bernard et al. [2], [8], [30], the technology is still far behind the power of humans.

My motivation for this project was to make an application which would be able to aid in image understanding. If I was also able to add a feature which could narrate the descriptions for people having trouble reading the captions it would be even better. To make this happen I needed to build a very strong model which would be able to describe images under different physical parameters. Hence, I chose the Neuraltalk framework [18] which has already been showcased at the Microsoft Image COCO, detecting and captioning challenge [27]. I emulated the training model which is used by Neural Talk. Once the model was built, I tested and validated it, to ensure the correctness of the model in order to use it for the application. My experiments to test the effectiveness of the model via scale and color augmentation gave me insights on what filters could prove efficient when the image scene is constrained in terms of light or other external parameters which hinder the scene quality. Lastly, I wanted this to work for real-time images hence I needed to build a seamless server which would be able to process concurrent requests and process the images to generate the results. Since, I had experience building android applications, I chose Android as a platform where the aforementioned concept was validated.

Let us understand the basics before we dive into our model creation.

2.1 What is a Neural Network?

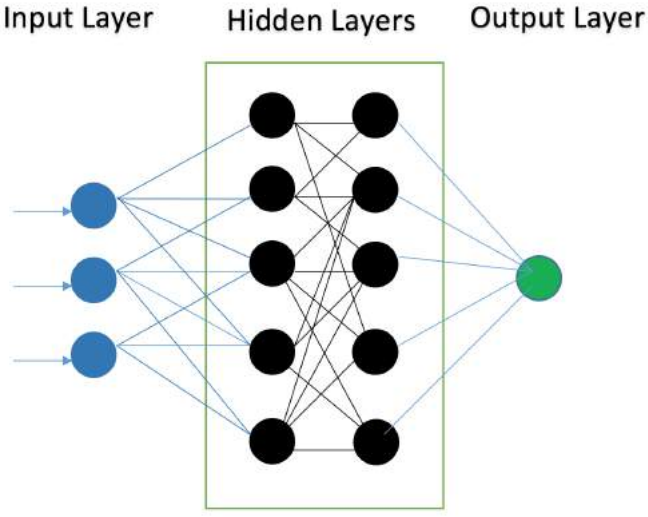

Figure 3: A Neural Network

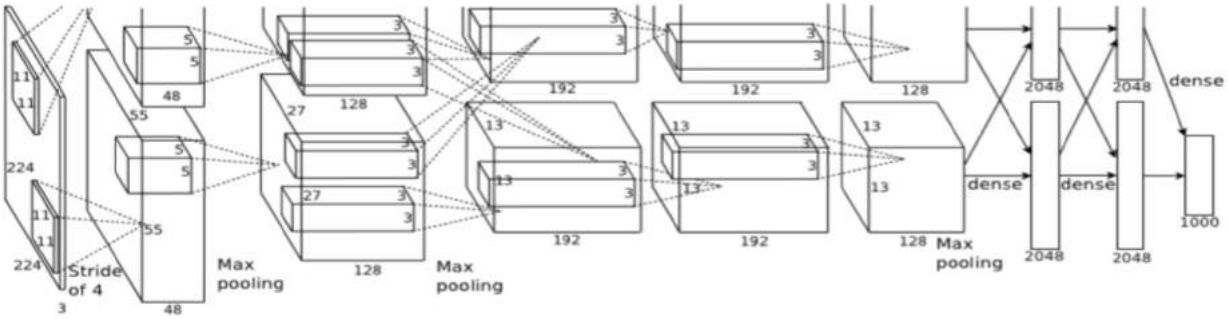

Figure 4: AlexNet CNN Architecture

An Artificial Neural Network consists of input nodes, hidden nodes and output nodes. Each node is called a neuron. These neurons are triggered by an activation function which then calculates the adaptive weights between those neurons. The final layer is the output layer which classifies based on the learning type. The figure 3 gives an idea of a neural network model. These are mainly used for classification problems. The model is learned through forward and backward propagation process where the weights are adjusted at every epoch to correct the error.

2.2 What is a Convolution Neural Network?

A Convolutional Neural Network (CNN), unlike the Artificial Neural Network, is a more powerful network which consists of a series of hidden layers often termed as Sub-Sampling and fully connected output layer. These hidden layers often have specific features to learn. Convolutional layers mostly consists of feature learning and are mostly used for image and voice recognition.

The figure 4 [20] is an architecture of AlexNet [20] which is most commonly used style of CNN. It consists of convolutional layers and max pooling. Max Pooling basically crops the image into half which only consists of important features of an image from the previous layers. The activation function used is RELU in most cases but there are others such as STEP or SIGMOID. The number of neurons for such network consists of about 60 million neurons, with 5 convolution layers and 3 max pooling layers and fully connected output layers.

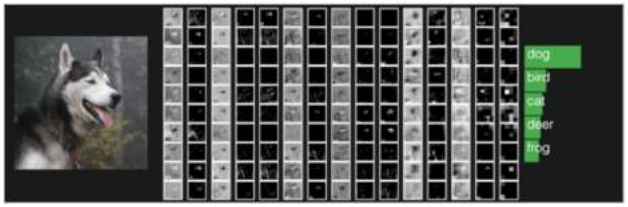

Figure 5: Illustration of Convolution

Convolution is basically application of filters which help understand the image features in order to distinguish them from another set of pixels. These are very helpful to understand features over layers of convolution as you may notice from figure 5 [17]. At every layer an image gets activated differently over a filter. We perform series of this operation and the filters are assigned randomly and the model is set to train. Error is calculated based on the label, if it classifies correctly at every epoch, else it back propagates to correct the errors by Soft-max derivatives.

2.3 What is a Recurrent Neural Network?

Recurrent Neural Network is another type of Neural Network. Unlike a traditional Neural Network, neurons are not independent of each other. These are mainly used in Language modelling to predict the next word, like we see on a Google search bar, where, whenever we type a word, a series of words shows up. The words are predicted from the sequence of words hence it knows the previous words in sequence. So basically, we feed the input a sequence of words which converts these words to a vector in the form of one hot encoding and every step is considered as time steps to predict the most effective word. When it comes to back propagation, the network knows the gradients of previous time step as well as those of the current time step.

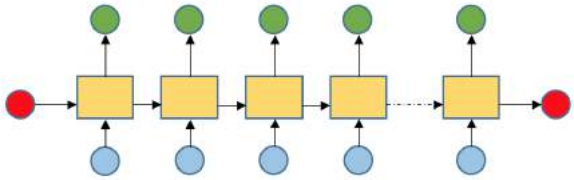

Figure 6: Illustration of RNN

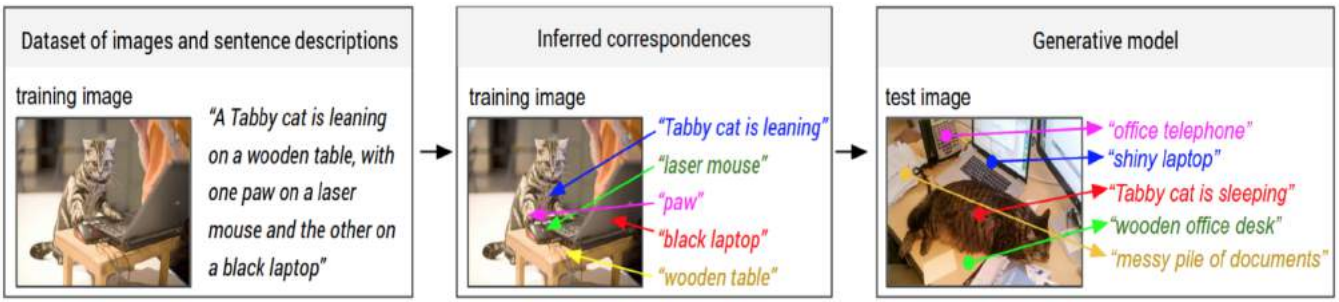

Figure 7: Overview of the model

2.4 What is Multimodal Learning?

Multimodal learning is basically a scenario in deep learning where more than one modelling takes place over an action. For e.g. if we have video and audio recording of a person talking and we want to create a model where it can read the lip movements to predict the words, we would employ Multimodal learning. Here, the lip movements are visual and corresponding audio embedding as a multi-modal learning. Cross modelling is a nature where the audio and video are correlated. Here, one model usually acts as a cue to the other. In our scenario, we use image and captions in the training data to be embedded via a multi-modal embedding to generate sentences.

2.5 Deep Visual-Semantics Alignments for Generating Image Descriptions

Karparthy et al. [18] presented a model that generates natural language descriptions of images which is incorporated in my base reference paper for my capstone project. Karparthy [16] who is currently a PhD student and lecturer at Stanford University developed the model for image captioning. Their approach is novel which has 2 parts Alignment Model which is a combination Convolution Neural Network for image regions and bi-directional recurrent neural network for captions and aligns these 2 models via multi-modal embedding and a Multimodal Recurrent Neural Network model which uses the inferred alignment model to generate image descriptions. Figure 7 [18] shows the overview of the model creation steps.

2.5.1 Aligning Image with Language

Since, it is a supervised learning the captions are related with the objects from the image. In figure 8 [18], we can see Tabby cat is leaning on a wooden table with one paw on a laser mouse and the other on a black laptop. Here Tabby, paw is referred to cat, laser mouse, black laptop is referred to laptop and wooden table is referred to the table. Here the objects detected from the image are mapped to the regions with the text from the captions via a multi-modal embedding.

The Bi-directional Recurrent Neural Network (BRNN) is responsible for aligning the words with respect to the image areas based on the score that was allocated for each combination of words. Each word is rated based on the context around the region through a vector space. The image-sentence score is computed from vectors of words and the regions in the image as shown in figure 9 [18].

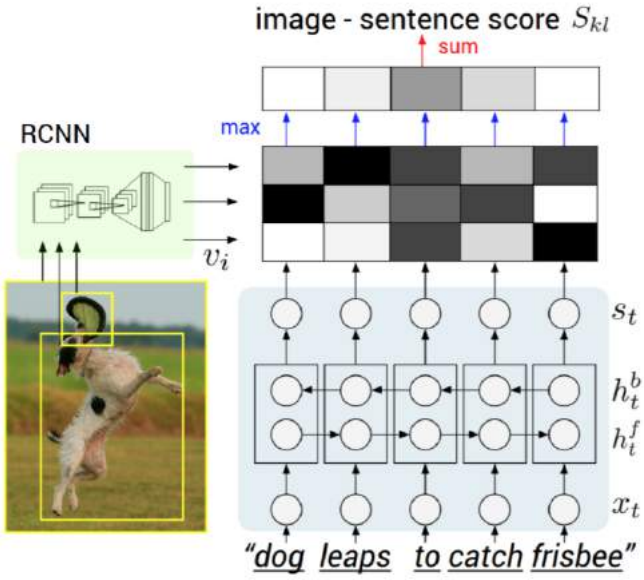

Figure 9: Evaluating the sentence creation score

2.5.2 Multi-modal Recurrent Neural Network for Generating Image Descriptions

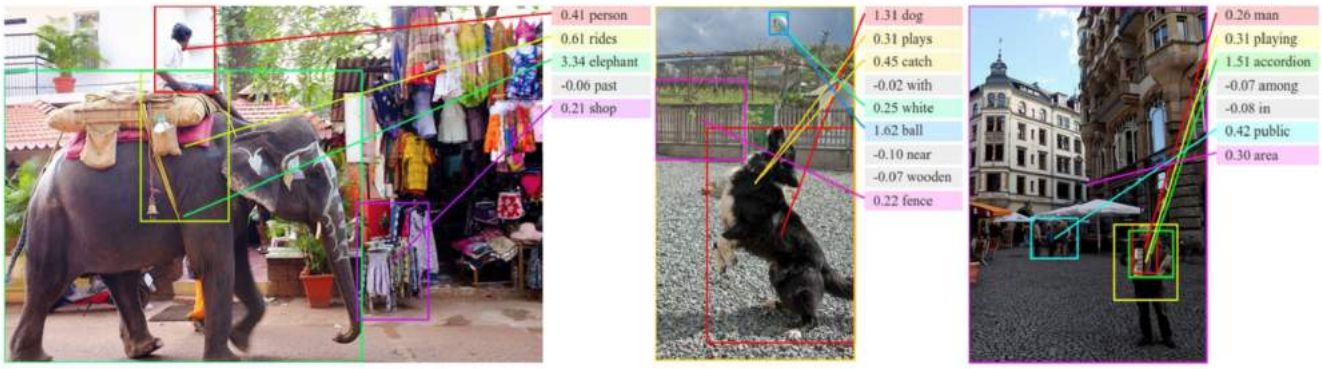

Figure 8: Aligning Image with Language

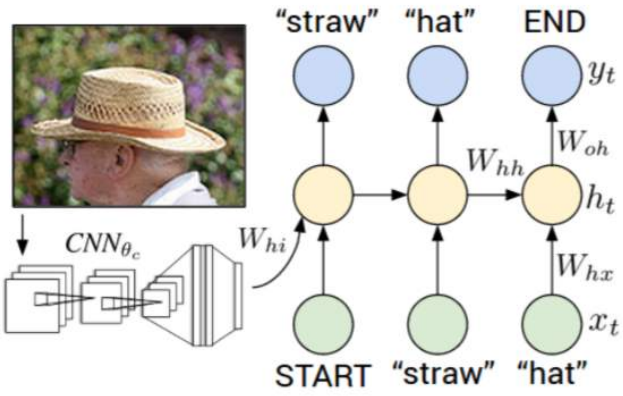

Figure 10: Evaluating the sentence creation score

The RNN takes a word which are generated by the CNN and the context from previous time steps as it proceeds and defines a distribution over the next word in the sentence [18]. The RNN is conditioned on the image based on the information from the first time step. START and END are tokens as shown in figure 10 [18] indicating the start and end of a sentence which is generated here.

3. SOLUTION

To build an effective image captioning system, I divided my project into sub-modules since every module was independent in terms of development. Starting with Neural Talk setup, followed by Model Training, Server, and Application.

3.1 Neuraltalk

Neuraltalk is a framework developed by Andrej Karpathy et al. [18] which is built using Torch [7]. The good thing about Neural Talk is training of image data can be batched, it supports fine tuning and supports CUDA [28] for GPU computations. Torch [7] is scientific computing framework which supports GPU and widely used by Machine Learning scientists because it provides maximum flexibility to developer to tinker with the fine tuning of the models.

3.2 Model

Neural Talk which comes with default VGGNet-16 pre-trained model for CNN. I used another model ResNet-18 which is way more powerful than VGGNet as per the ImageNet Challenge [15]. Therefore, I used both the pre-trained checkpoints of VGGNet-16 [32] and ResNet-18 [14] CNNs for the project.

3.3 Server

A multi-threaded server architecture was needed if, in order to handle multiple requests. I chose the server design to be in Java using the Network API.

3.4 Application

I was motivated to choose Android since it contains a lot of open source packages. The application makes use of Camera API, Text to Speech API which comes with the Android Framework, Network API which is a part of Java Network API.

4. SYSTEM ARCHITECTURE

Figure 11 shows a high level architecture of the system. The app will capture the image and it is sent to the server and the server processes the image by calling the model and gives the caption. There are few things which have been tweaked the way app interacts with the server in the course of development. Since the model invokes Torch methods, it needs a command line interface therefore, I used exec as part in server which will call the torch model.

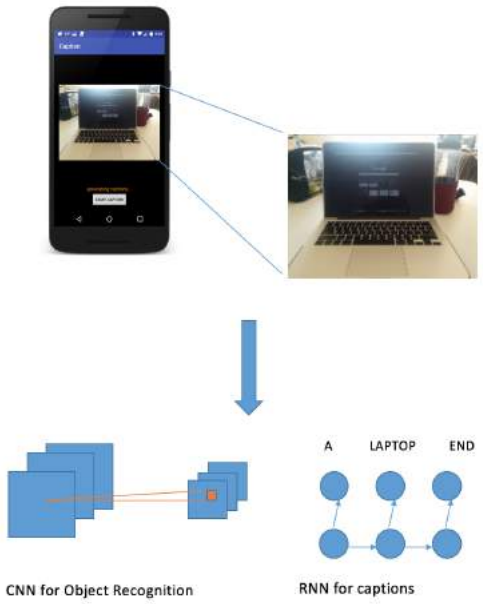

Figure 11: System Architecture

4.1 Server Component

The server is designed using Network API of Java which uses Socket I/O where the server listens for a connection and when the connection is established it receives the file and using the exec it calls the model. It is a multi threaded server environment where it handles the concurrent connection request; it can also handle asynchronous caption requests.

Once the connection is established the server then saves the image file in the directory which is pointed to the model. The server calls the torch and loads the checkpoint instance along with the directory which points to the image and then fetches the dumps from the model and image run which is a json file and the fetches the captions and sends the output as a plain-text.


```
[{"caption":"a dog is sitting on a skateboard in a room","image_id":"1"},
{"caption":"a cat is sitting on a chair with a stuffed animal","image_id":"2"},
{"caption":"a cat is sitting on the floor next to a pair of scissors","image_id":"3"},
{"caption":"a close up of a person holding a cell phone","image_id":"4"},
{"caption":"a cat sitting on top of a pair of shoes","image_id":"5"}]
```


Figure 12: Vis Json

Figure 12 is a sample json caption representations where every image associated with an id and corresponding captions generated with model interaction.

4.2 Model Training

The model training is a part of Neuraltalk framework [18] - there are a lot of pre-requisites for it to function properly. Firstly, we need to install the torch framework [7], which is the crux of our modelling. Torch [7] is a Lua [23] based machine learning framework has lot of packages in Computer Vision, Images and Machine Learning etc. which are necessary to build the model. After installing torch we need to add nn, nngraph and image packages which are essential for pre-processing and model parametrization. I installed loadCaffe to use a pre-trained VGGNET model [32] or ResNet model [14]. Other dependencies include torch-hdf5 [13] and h5py [29] which are the formats to store preprocessed data. This is the basic setup to train the model. This framework also supports CUDA [28] powered GPU support for batch training but I have used CPU checkpoint of the model.

Now, we have captions and images in their respective directories and I have written a python script, which will take all the images and captions and generate the HDF5 format for images and json file for the captions associated with the images, since it is supervised learning. There are different metrics to evaluate the training models such as BLEU, METEOR, CIDEr [35]. CIDEr metric is used to evaluate the correctness of the captions generated as said by the authors [18]. These are the initial setup for the model interaction.

To load the image for generating captions we can enforce some pre-requisites such as batch size where we can load more than one image to generate the captions. Once we load the model which has the pre-trained weights and the output of this operation is then parsed to get the output as shown in figure 12.

4.3 Mobile Application

I chose Android as a platform for developing the mobile application. Android is Unix based operating system for mobile and tablet devices and coverage in terms of people across the globe is much higher and lots of the components are open source. The application requires core image permissions and uses camera utilities to capture the image. It also needs network permissions for it to interact with the server and send the image and receive the captions. I have also added voice accessibility, which is a feature for partially blind users who have a lot of difficulty in reading the texts. The application will help them by narrating the captions to them. This application is designed for a wide range of users all of who can benefit from the app. It has a very simple user interface requiring only the capture of images; while the rest of the functionality is taken care asynchronously.

5. EXPERIMENTATION

The experiments involved evaluating the performance of the model, to see how the model is resistant to negative effects of scale and color augmentation. The intention of these experiments is to know how the model behaves in real time on different android devices. Since, the camera of every android device has a different quality based on the lenses used and the filters employed, which comes with default settings in the camera; it was necessary to test whether the model would be successful in generating the most effective captions across all devices.

5.1 Scale Augmentation

To see how the model reacts to different resolutions of the same image, I used Matlab to generate the images for different resolutions of the image. I regenerated the image into 3 different resolutions high, medium and low resolution.

Original image size: 640 x 478 pixels
High Resolution size: 1600 x 1600 pixels
Medium Resolution size: 720 x 720 pixels
Low Resolution size: 120 x 120 pixels

While performing the scaling some of the images were pixelated. I ignored those images which wouldn't work well.

5.2 Color Augmentation

To see how the model reacts to different filters of the same image, I used Matlab to generate the images for different color augmentations. I refer to them as Filters. I regenerated images on different filters namely the Gaussian Filter, High Light Filter, Low Light Filter and Complement Filter.

Figure 13: Original Image

Figure 14: Image with Gaussian Filter

Figure 15: Image with High Light Filter

Figure 16: Image with Low Light Filter

Figure 17: Image with Complement Filter

5.3 Semantic Similarity

To gather insights of captions generated from each of these filters and resolutions, it was necessary to compare the semantic similarity of the captions. Since, these captions generated could often be very similar and know the closeness

of the caption generated for each of the images with filters and resolutions, I used Semantic Textual Similarity System [12] to compute the textual relevance between these captions and caption generated by the original image.

The semantic textual similarity for sentences were computed by vectoring the words in each of the 2 sentences. Word weights were assigned based on similar words for each of the vectors formed by the 2 sentences. Natural Language Toolkit (NLTK) [22] a library in python offers Word Net for computing the similarity index for the word vectors.

5.4 Classification on Image Meta-data

Today, most of the mobile devices have Geo-tagging features in the meta-data for an image. I tried to extract meta-data to work on creating a classifier for the images based on location and time of the day. But, it was an unsuccessful attempt to explore this in my current data-set since all the images were studio images and all of them had the same meta-data properties.

6. RESULTS

Model is trained, where one epoch consists of 7500 iterations for all the images which has a validation loss of 2.5 [18] and CIDEr score [35] of 0.6 with the existing setup of Neural Talk. To test the model accuracy in terms of adding external image, filters such as bright light, low light, ultra violet, gray scale, complement to view the performance of the behavior of the models and I found out that the high light filter works best when augmented over the natural image when requested for captions. Ultra-violet, gray-scale filters were ignored later on since some of the images failed while processing and it was an arduous task to locate these image in 500,000 images. However, with these 4 filters I found some interesting results as follows:

Gaussian	Complement	Low Light	Bright Light
0.2066153	584360	0.5304865	0.68249434
0.022539543	0.7245675	1.0	0.93347347
0.5486957	0.6164745	0.72526854	0.6597918
0.45356426	0.26592883	0.26592883	0.47931468
0.21508801	0.40779695	0.6504272	1.0
0.18436246	1.0	1.0	1.0
0.3046647	0.27067313	0.8791307	0.8791307
0.19016795	1.0	0.43320605	1.0
0.10900717	0.86354446	0.86354446	0.8388492
0.13962337	0.3837621	0.07842342	0.41652742
0.40373594	0.416749	0.4091154	0.8627525
0.2023001	0.17715827	0.7906016	0.9520686
0.35792506	0.20865658	0.8923272	0.75882375
0.33117455	0.43603274	0.2504242	0.43248424
0.09305642	0.5512116	0.3763584	0.6603746
0.34327427	0.71454877	0.56631196	0.7952623
0.069164366	0.10046709	0.5933321	0.5557013
0.055851806	0.11462467	0.7554655	0.86048937
0.21823758	0.18996765	0.48923883	0.31487086

Table 1: Snippet: Semantic Score comparisons for various filters

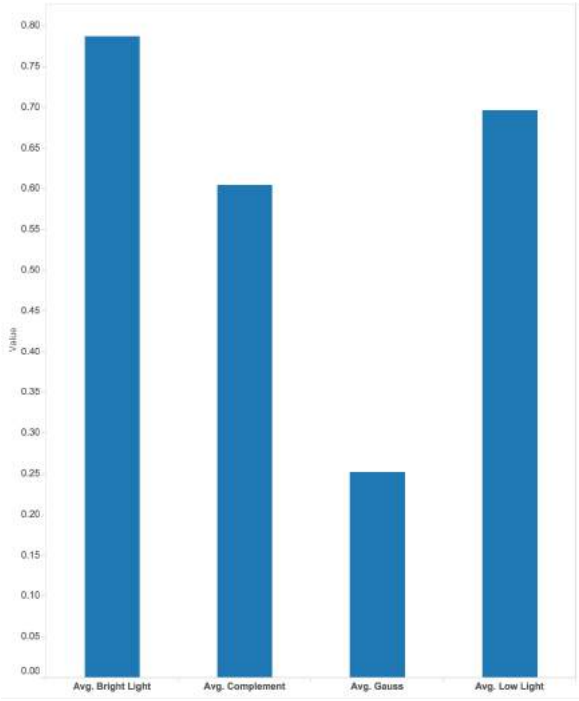

Figure 19: Similarity score vs Filters

The reason behind working with different filters was to gain insights which could be used in the application. One hypothesis was to also find the behaviour when the nature of the images was changed since one of the situations in which this application can be used is when the physical property of the environment is dynamic since the training images are mostly designed using studio images in the COCO data-set [4] and Flickr [36] has macro photographic images.

I also performed the 10-fold cross validation to measure the classification rate. From the following figure 18, you get the idea that the images with bright light filters performed very well but when the images are at low light, they perform poorly, in giving us the Semantic Similarity score [12].

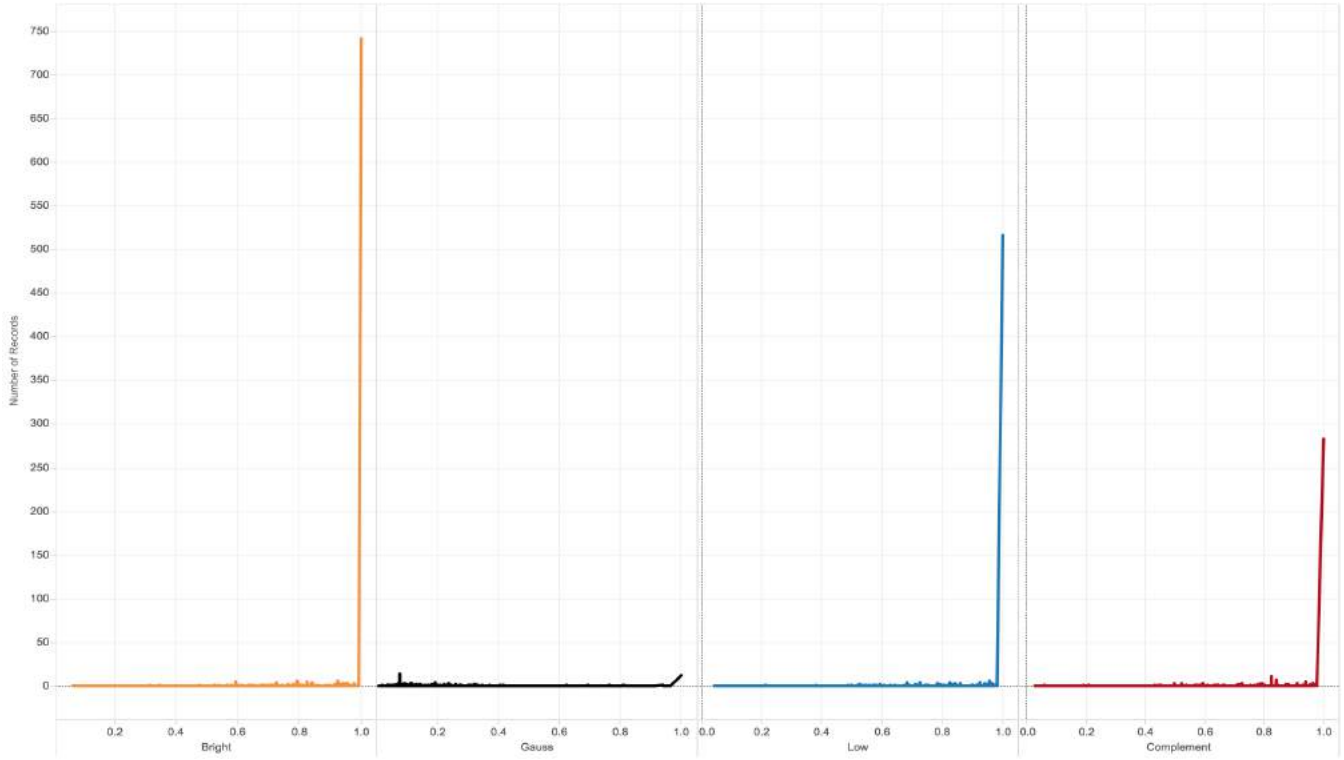

Figure 18: Histogram representation of Accuracy vs Filters

High	Medium	Low
0.3349537	0.21600127	0.040493354
0.33539537	0.2167335	0.040860724
0.34960675	0.21883325	0.048114963
0.35250682	0.22581546	0.049221758
0.3615237	0.22716723	0.053482547
0.36647725	0.22923549	0.055820897
0.396349	0.23092155	0.05985157
0.41323292	0.23326282	0.060182136
0.4133984	0.23676965	0.064108185
0.41697812	0.24851947	0.0673506
0.41806504	0.2560858	0.10288555
0.42137182	0.2791866	0.103754826
0.424739	0.2886546	0.10891973
0.43963566	0.29692376	0.10936365
0.4416984	0.30013916	0.10956612
0.44961908	0.30902258	0.1114843
0.45842206	0.32140827	0.11579245
0.46665382	0.34999007	0.2873589
0.48494866	0.38744482	0.564743

Table 2: Snippet: Semantic Similarity Score comparisons for different resolutions

From Table 2, we also observed that images with higher resolution the model performs better in terms of generating the description which are close to that of the original image.

7. CONCLUSION

The model was build using 70,0000 images comprising of both MSCOCO [4] and Flickr [36] data-sets using Neural Talk [18] and Torch framework [7] which gives us natural language description of the image scene. For Validation data-sets of 500,000 images, I used Matlab to generate various filters and resolutions on images and explored the image semantics similarity [12] with the original image. Moreover, I learned the factor of similarity for images with high, medium and low resolutions. I was successful in understanding that bright light filters work the best in generating effective captions. Moreover, some interesting results also came to light when using the complement filters for low light images. Also these gave better results than the low light images using the semantic similarity of the captions generated but it was limited to certain images.

8. FUTURE WORK

Parametrization of images with filters at the convolutional step improves the quality of trained model. Variants of CNN could be tried to maximize score with transfer learning models where some layers of CNN are trained using set of data and other layers of CNN are trained using completely different data set. For related work in this field, this approach has proved to generate better results. ResNet-152k is currently state of the art technology for image captioning. With the right computing resources it is possible to achieve the most

accurate results for image detection and captioning. From the application standpoint, we can incorporate lots of use cases such as American Sign Language where people can communicate by reading through captions and reverse engineering the process where the system is capable of understanding the gestures to generate captions and incorporating sound as well. To further the goals of this application, we can introduce the concept of peripheral vision which is still not used much in mobile computing world, where we can leverage the front camera to track eye movement and focus the region of an image to generate caption. Live video captioning applications can also be integrated, which has a very similar setup.

9. ACKNOWLEDGEMENTS

I want to thank Professor Jeremy Brown [3] for his inputs and guidance through all phases of the project and who helped make it possible. Andrej Karpathy [16] for his wonderful research on Deep Visual-Semantic Alignments for Generating Image Descriptions [18] which cultivated interest in me to develop the application based on deep learning techniques via Neural Talk.

10. REFERENCES

[1] F. Ahmad, A. Najam, and Z. Ahmed. Image-based face detection and recognition:" state of the art". *arXiv preprint arXiv:1302.6379*, 2013.

[2] K. Barnard, P. Duygulu, D. Forsyth, N. De Freitas, D. M. Blei, and M. I. Jordan. Matching words and pictures. *The Journal of Machine Learning Research*, 3:1107–1135, 2003.

[3] J. Brown. https://www.rit.edu/gccis/jeremy-brown.

[4] X. Chen, H. Fang, T.-Y. Lin, R. Vedantam, S. Gupta, P. Dollar, and C. L. Zitnick. Microsoft COCO Captions: Data Collection and Evaluation Server. *ArXiv e-prints*, Apr. 2015.

[5] X. Chen and C. L. Zitnick. Mind's eye: A recurrent visual representation for image caption generation. In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*, pages 2422–2431. IEEE, 2015.

[6] R. Datta, J. Li, and J. Z. Wang. Content-based image retrieval: approaches and trends of the new age. In *Proceedings of the 7th ACM SIGMM international workshop on Multimedia information retrieval*, pages 253–262. ACM, 2005.

[7] Facebook. Torch - http://torch.ch/.

[8] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding*, 106(1):59–70, 2007.

[9] L. Fei-Fei, A. Iyer, C. Koch, and P. Perona. What do we perceive in a glance of a real-world scene? *Journal of vision*, 7(1):10–10, 2007.

[10] E. Gobbetti and R. Scateni. Virtual reality: Past, present, and future. *Virtual environments in clinical psychology and neuroscience: Methods and techniques in advanced patient-therapist interaction*, 1998.

[11] Google. Goggles - https://play.google.com/store/apps/details?id=com.google.android.apps.unveil&hl=en.

[12] L. Han, A. L.Kashyap, T. Finin, J. Mayfield, and J. Weese. Umbc_ebiquitycore: Semantic textual similarity systems. In *Proceedings of the Second Joint Conference on Lexical and Computational Semantics*. Association for Computational Linguistics, June 2013.

[13] HDF5. https://en.wikipedia.org/wiki/hierarchical_data_format.

[14] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[15] ImageNet. Ilsvrc models - http://www.vlfeat.org/matconvnet/pretrained/.

[16] A. Karpathy. Andrej karpathy - http://cs.stanford.edu/people/karpathy/.

[17] A. Karpathy. blog - what a deep neural network thinks about your selfie? http://karpathy.github.io/2015/10/25/selfie/.

[18] A. Karpathy and L. Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3128–3137, 2015.

[19] K. G. Krishnan, C. M. Porkodi, and K. Kanimozhi. Image recognition for visually impaired people by sound. In *Communications and Signal Processing (ICCSP), 2013 International Conference on*, pages 943–946, April 2013.

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[21] T. Lan, M. Raptis, L. Sigal, and G. Mori. From subcategories to visual composites: A multi-level framework for object detection. In *International Conference on Computer Vision (ICCV)*, 2013.

[22] E. Loper and S. Bird. Nltk: The natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1*, ETMTNLP '02, pages 63–70, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

[23] Lua. https://www.lua.org/about.html.

[24] W. E. Mackay. Augmented reality: linking real and virtual worlds: a new paradigm for interacting with

computers. In *Proceedings of the working conference on Advanced visual interfaces*, pages 13–21. ACM, 1998.

[25] Microsoft. Hololens - https://www.microsoft.com/microsoft-hololens/en-us.

[26] A. F. Mollah, N. Majumder, S. Basu, and M. Nasipuri. Design of an optical character recognition system for camera-based handheld devices. *arXiv preprint arXiv:1109.3317*, 2011.

[27] mscocochallenge. Microsoft common object in context image captioning challenge - http://mscoco.org/dataset/#captions-leaderboard.

[28] NVIDIA. Cuda - http://www.nvidia.com/object/cuda_home_new.html.

[29] Python. H5py - http://www.h5py.org/.

[30] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[31] Shazam. http://www.shazam.com/.

[32] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[33] SoundHound. http://www.soundhound.com/hound.

[34] TechEmergence. http://techemergence.com/the-rise-of-neural-networks-and-deep-learning-in-our-everyday-lives-a-conversation-with-yoshua-bengio/.

[35] R. Vedantam, C. L. Zitnick, and D. Parikh. Cider: Consensus-based image description evaluation. In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*, pages 4566–4575. IEEE, 2015.

[36] P. Young, A. Lai, M. Hodosh, and J. Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions.