Image Descriptive Summarization by Deep Learning and Advanced LSTM Model Architecture

A DISSERTATION

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF

MASTER OF TECHNOLOGY
IN
SIGNAL PROCESSING AND DIGITAL DESIGN

Submitted by:

Tushar Aggarwal

2K17/SPD/16

Under the supervision of

PROF. N.S. Raghava

DEPARTMENT OF ELECTRONICS AND COMMUNICATION
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

OCTOBER,2019

DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

CANDIDATE'S DECLARATION

I, Tushar Aggarwal, 2K17/SPD/16, of M.Tech, hereby declare that the project Dissertation Titled "Image Descriptive Summarization by Deep Learning and Advanced LSTM model Architecture" which is submitted by me to the Department of Electronics and Communication, Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of the degree of Master of Technology, is original and not copied from any source without proper citation. This work has not previously formed the basis for the award of any Degree, Diploma Associateship, Fellowship or other similar title or recognition.

Place: Delhi **TUSHAR AGGARWAL**

Date:

DEPARTMENT OF ELECTRONICS AND COMMUNICATION
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

CERTIFICATE

I hereby certify that the Project Dissertation titled "Image Descriptive Summarization by Deep Learning and Advanced LSTM model Architecture" which is submitted by Tushar Aggarwal, Roll No 2K17/SPD/16, Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of the degree of Master of Technology, is a record of the project work carried out by the students under my supervision. To the best of my knowledge this work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere.

Place: Delhi

Date:

PROF. N.S. RAGHAVA

SUPERVISOR

H.O.D.

Department of Electronics and Communication Engineering,
Delhi Technological University,
Bawana Road, Delhi-110042

ACKNOWLEDGEMENT

l thanks GOD almighty for guiding me throughout the semester. I would like to thank all those who have contributed to the completion of my project and helped me with valuable suggestions for improvement.

I am extremely grateful to **Prof. N.S. Raghava**, Division of Electronics and Communication, for providing me with best facilities and atmosphere for the creative work guidance and encouragement.

Above all l would like to thank my parents without whose blessings, I would not have been able to accomplish my goal.

……

Tushar Aggarwal

ABSTRACT

Auto Image Descriptor is becoming a trending point of interest in current era of research among researchers. Being a great community, which is proposing a continuous and enhanced Iist of intuitive algorithms which is solving to its problems. However, still there are lot of improvement to this field. Therefore, it's becoming a field of attraction for many researchers and industries and reliable to this digital world. Of these various image descriptive algorithms, some outperform others in terms of basic descriptors requirements like robustness, invisibility, processing cost, etc.

In this thesis, we study a new hybrid model image descriptor scheme which when combined with our proposed model algorithm provides us efficient results. Following illustrative points are made to describe the thesis in a nutshell which will later on be discussed in detail.

- Firstly, we train our image in 9x9 kernels using CNN model. The idea behind this 1024 kernel of our host image is to divide each pixel of host image with lowest human value system characteristics i.e lowest entropy values and lowest edge entropy values.
- Our host image is further divided into 8x8 pixel blocks. Therefore, we'll have 64 rows and 64 columns are there of the 8x8 blocks. In total 64x64x8 blocks of host image i.e. the size of host image is 512x512.
- Now we captionate the pre trained labels to our pixelate model obtained from our CNN model. This will be used in embedding of our labels with the LSTM algorithm. Using

LSTM model algorithm, it will assign a label to each pixelate kernel which will perform embedding to the host image.

- This embedding and extraction is done Long Short Term Memory algorithm, which is explained in later chapters.
- Now using this embedded image, we our Quest Q function of our host image using embed RNN network.
- We find best value of Q function using this RNN networks. Also we get the best computed label for our host image using this algorithm.
- In a nutshell, this project has combined four major algorithms to generate best results possible. These adopted criteria significantly contributed to establishing a scheme with high robustness against attacks without affecting the visual quality of the image.

To describe it briefly the project consists of following four subsections-

1. Convolutional Neural Networks (CNN)
2. Long Short Term Memory Algorithm (RNN)
3. Recurrent Neural Networks (RNN)

CONTENTS

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

This era has witnessed a rapid growth in the availability of digital and various multimedia content. Today, digital media contents are spread via the World Wide Web among large number of people without much demanded efforts. Additionally, unlike traditional time copying, in which the quality of the duplicated content is disturbed, and on other digital tools can easily be produced large amount of perfect copies of digital data within a short span of period. This ease of digital multimedia distribution over the Internet, together with the possibility of unlimited duplication of this data, threatens the intellectual property (lP) rights more than ever. Thus, content owners are eagerly seeking technologies that promise to protect their rights.

Deep Learning is a very emerging field right now – with various different applications emerging out day by day. And the best way to get deeper into Deep Learning is to get experimenting with it. Bring out as much projects as we can, and try to do them on your own. This would help you grasp the topics in more depth and assist you in becoming a better Deep Learning practitioner. Here, we will try to a look at an one of the multi modal topic where we will encapsulate both image and text processing and try to evolve a useful Deep Learning application, aka Image Captioning. Image Captioning is defined as the process of captionate a textual description from an image which is based on the action of objects and other perceptual of the image.

It is based on intuition supervised works because neural networks that are governed by the human brain in the first place. So, re-transformed the problem should definitely work. Depending on the application, a watermark is required to survive all the possible manipulations the host data may undergo as long as they do not degrade *too much* the quality of the document. The main difference between watermarking and encryption is that encryption disguises the data and protects it by making it unreadable without the correct decryption key, while watermarking aims to provide protection in its original viewable audible form. Hence image captioning plays a vital role in this advanced era of technology. While a single bit of information is used by almost every part of field like defense, biomedical, social media etc. This information carried out many other sources of information which is further extracted by other

sources like Facebook, Google.

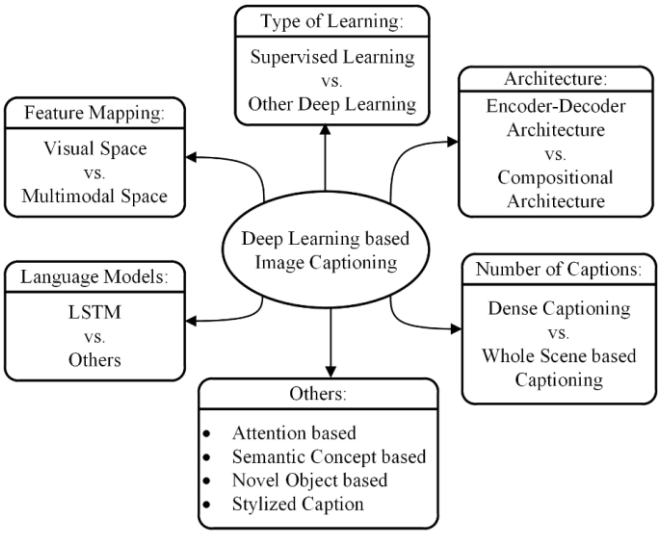


Fig. 1 Deep Taxonomy of deep learning captioning [1]

This helps them to process the large set of digital data set within a fraction of time period and provide judicious information content about the object. Captioning can be used generally used in this modern era of Artificial Intelligence to understand the need of the person and the language of the image of this digital world. With an efficient designed model of image captioning machine can predict and extract out the pin information like location, occlusion and other part of information which can't be judged by naked eyes. Earlier the captioning was done using Scaled Invariant Techniques, Support Vector Machine like techniques which proved to be inefficient with the complex modern era problem. These are of state proved to be invariant and time-consuming methods when we used them over large piece of dataset and to compute large complex format of data. On other side of coin with the evolution of deep learning techniques it become really easy part of thing to compute them over large and complex form of data. This helps the observer to observe various efficient information with high accuracy rate.

In this thesis, our main purpose is to purpose new technique of image captioning technique using the combination of various efficient neural network techniques which are present in this field of technology. These algorithms are already renowned for their markup performance. Hence, we selected few of them to accelerate our model system.

The proposed method affects the old performed model in the sense of hitting the

anomalies and other error marks as a score to compute each time to enhance the productivity of our model.

Segmentation of image pixel in to the small pixelate form arranged in the form of un-detectability and imperceptibility requirements.

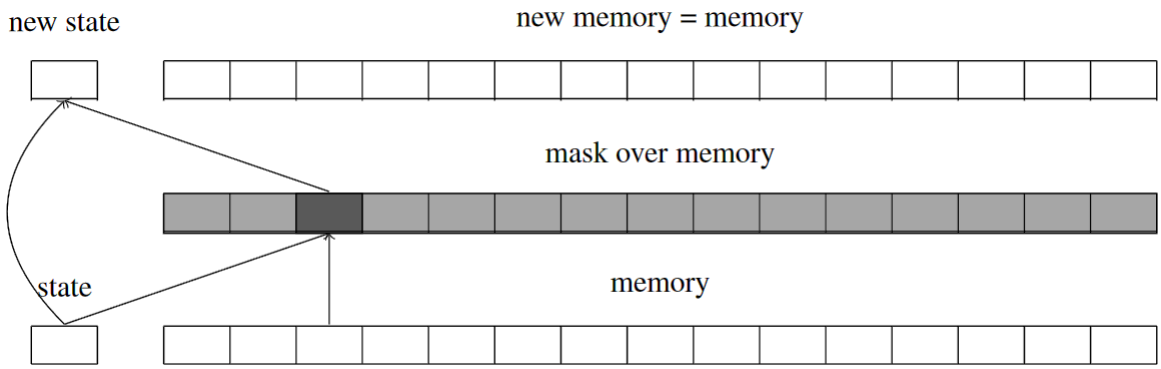


Fig. 2. State of Image generated [1]

The pixelate feature of the governed image is reproduced on basis of score prediction by model properties, improving the robustness and imperceptibility.

The CNN conversion of image to pixels is the best way to construct various part of information without touching the prominent part of information and to enhance the robustness of the judging images.

(i) The scheme employs by LSTM (Long Short-Term Memory) to the pixelates of the image by assigning a piece of the part of speech as a unsupervised labels which will be corrected out by the RNN networks which will be applied over it in our further steps.

(ii) After this we will follow up with the RNN networks as the advantage of applying to every form of image is as follows:

(iii) RNN treat each part of object as a independent form of object without depending on other layered data hence it will generate better outcome dynamically.

(iv) It is computed without large computational resources hence large dataset can be burned over minimal time period of data.

(v) The basic function of the RNN [3] is shown in equation below

$$J(\theta) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{j=1}^{|M|} y_{t,j} \log \overline{y}_{t,j} \qquad (1.1)$$

M = vocabulary , J(θ) = Cost function

Therefore, every time we run it over the object image it will assign a cost function and a score in each iteration which helps in captioning in final epochs. Below, in detail, is described the outline of all the four algorithms which are used in this thesis that are-

- Convolutional Neural Networks
- Long Short-Term Memory Algorithm
- Recurrent Neural Network

1.1 Convolutional Neural Networks

CNN is one of the widely used networks in the image processing in deep learning. The convolutional network governs the field of computer vision majorly. It takes digital image as an input to the model in which important characteristics of the image is extracted out using the kernel of the CNN model. After this it provide the image kernel weights and a bias to it so that it can easily differentiate the various images in the model. In nature, most signals are time varying. Therefore, wavelet transforms are suitable for many applications. As with any wavelet transform, a DWT is used to describe an image as small waves (called wavelets) of varying frequencies and limited durations.

A CNN model provides a COVNET architecture to the kernel architecture which is based on the pattern of human brain's neural pattern. This neural layer of COVNET assign a repetitive signal to the image kernel as a visual assistance to layered the image kernel into block pattern. This convert the image pixel to the matrix pattern of weighted pixelate. This enhance the prediction valuation of the model in a upward classical methodology of the system. As it brings out the Temporal and Spatial sources of the image which depends on the better fitting of the images which helps in the reuse of the assigned weights which reduce the dumping of large number of unnecessary garbage weights.

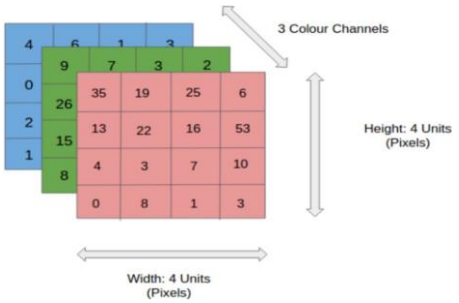


Fig. 3 Image Kernel [3]

Therefore, the weights are assigned as the power of 2. This process represents the image into 3 layered image matrix of RGB normalized vector. Image dimension is of 4X4X3 where 4 X 4 is the height and width of the image kernel and 3 is the depth of the kernel. These kernels are used for convolving the images by overlapping and iterating till each pixel gets convolved.

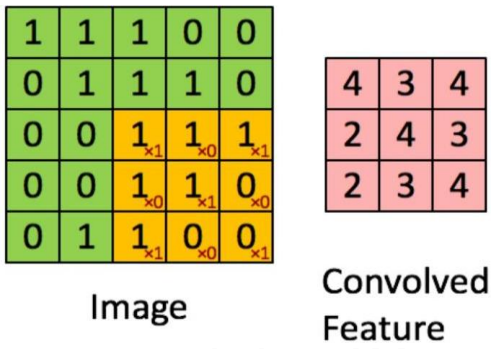


Fig. 4 Convolved Kernel [4]

After the convolution we perform the matrix multiplication over the convolved image to remove strides from then original image. The stride filter is keep shifting by optimal value till it hover up to last width pixel. In case of the multi dimension kernel we consider it as one- depth kernel such that it summed up to one bias sum Feature Output.

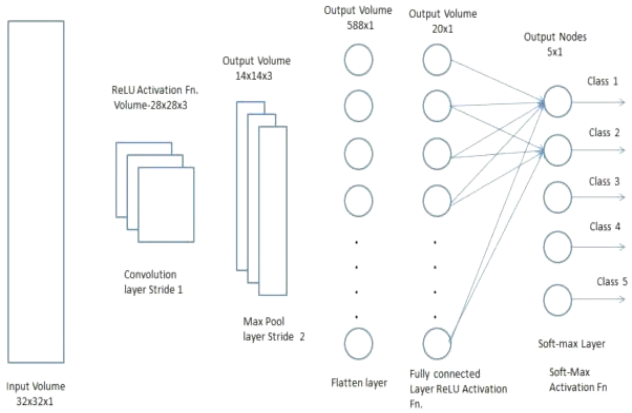


Fig. 5 Fully Connected COVNET [4]

COVNET not only summed up the bias but also helps in reefing the low feature pixels to adapt the Higher-level features which can be regulate using Padding methodology. Depending upon the complexities of images pooling layer will be increased to extract out feature variables from the complex image. After going through above complexities, we flatten the layered model to pass it through the

further Neural Networks for its classification.

Here in my thesis I have used fully connected COVNET network to reduce the complexities with less computations. Below is an image of depicts the feature mapping of the kernels to summarize the feature values of the image which can be further used for the feature scaling and classification of the application-based things for Neural Networks.

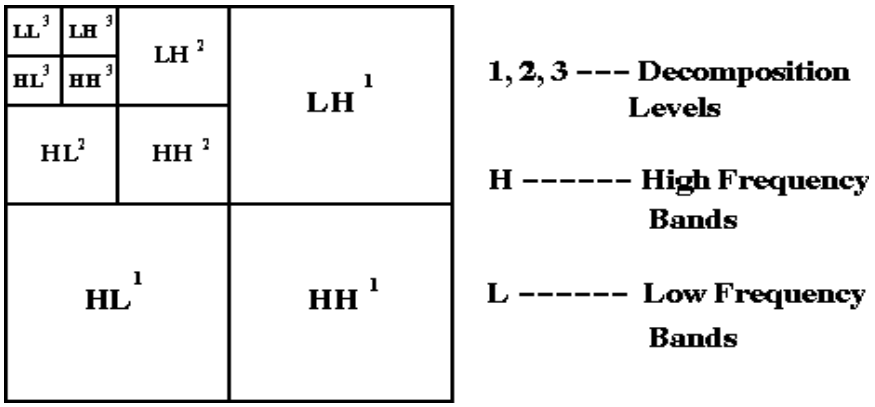


Fig. 6 Image Decomposition

1.1.1 Convolutional Layer

The linear process is considered a convolutionary layer in a Convolutionary Neural Network. With image processing function detectors, every node in the hidden layer extractions different features. The first node can, for example, extract the horizontal borders of a picture on the first layer, the second node can extract the horizontal borders, etc.. The functions are derived from a nucleus. The example of howstrided convolutions on an image is shown in Figure 3.11. The lower part is the picture and the top part is the production. The output of the convolutions also reduces the original image dimension.

1.1.2 Pooling Layer

After the convolutionary layer, the pooling layer happens. The explanation why it is pooled is because the aspect of the convolutionary layer is further decreased and features omitted to render the system more stable. Two kinds of pools are carried out: max pooling and average pooling. Max Pooling extracts a function with the largest pixel value, while the average pixel value is estimated to be removed.

1.2 Long Short-Term Memory

LSTM is one of the most widely used deep learning algorithm in the field of deep learning applications because of its robustness and its highly efficient performance. LSTM is widely used in daily life problems using the concepts of deep learning. As we now the deep learning is considered as black box in which the prediction based on iterative approach on feeding the input and getting the feedback from the model. It would be unfair if we say that neural networks don't consist of any memory as each layer of LSTM consist of brain like neurons which captures the feedback and assign the weights to the training data. Although this process is a static in process.

The figure below represents the architecture of the LSTM network. It takes input from three sources to the model.X_1 takes the input as a current time insertion. H_t-1 to embed the the third input from previous LSTM network layer as a generation of feedback. C_t-1 is the memory unit input from previous layer to store the weight of last generated iteration. H_t is the output generated by the current layer and its weights are stored in C_t for the current layer.

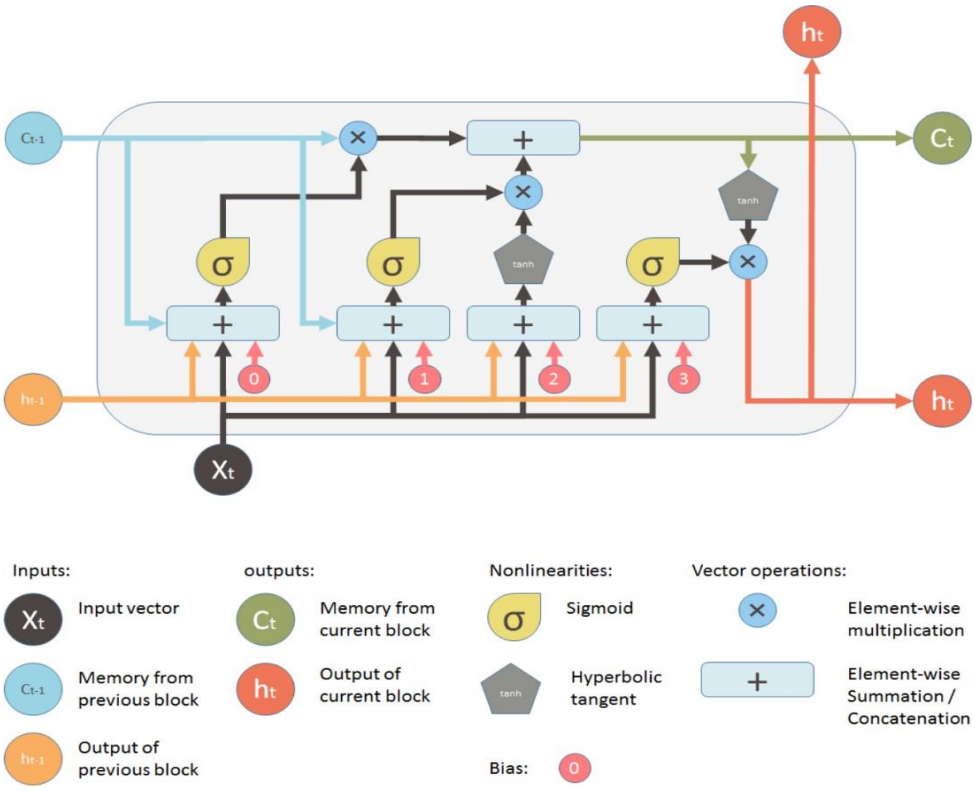
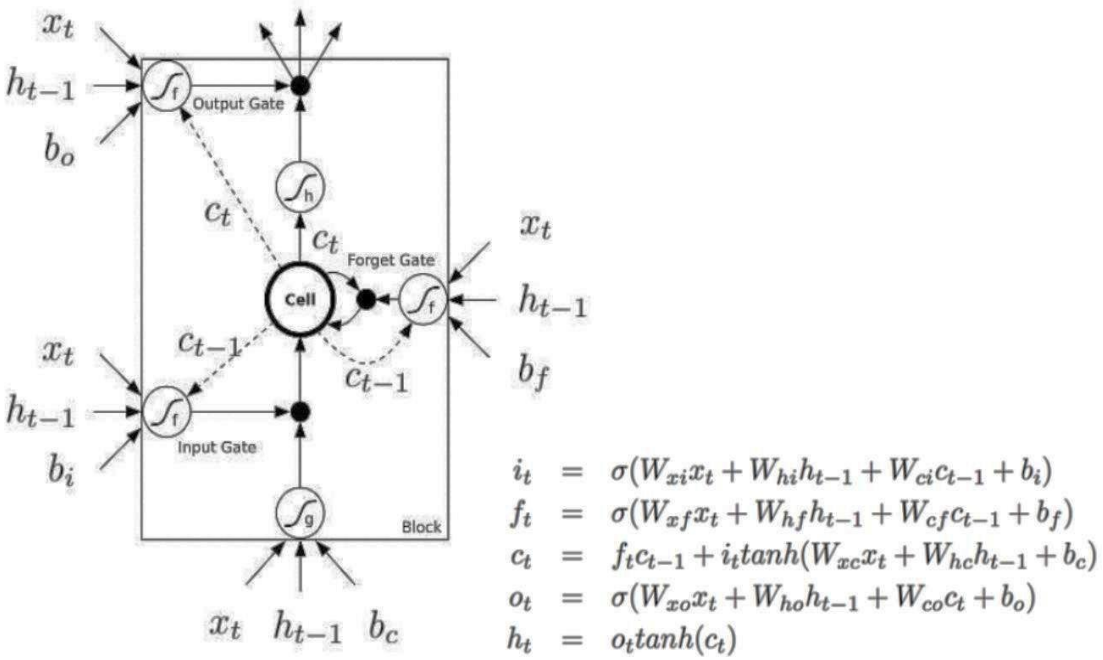


Fig.7 LSTM Architecture [5]

The above steps lead to the generation of many LSTM cells which governs the high accuracy rate for the classification model. First cell is considered to be a valve input which stores the cell memory if it gets switch off then no old memory will get stored for the next iteration hence no feedback will be generated. So, we connect second



cell to share the previous memory to the current cell to convolve the feedback and this results in the generation of new weights to the third ce

Fig. 8 LSTM Cell [4]

Analysis of Embeddings In order to represent the previous word St−1 as input to the decoding LSTM producing St, we use word embedding vectors [22], which have the advantage of being independent of the size of the dictionary (contrary to a simpler one hot-encoding approach). Furthermore, these word embeddings can be jointly trained with the rest of the model. It is remarkable to see how the learned representations have captured some semantic from the statistics of the language. Note how some of the relationships learned by the model will help the vision component. Indeed, having “horse”, “pony”, and “donkey” close to each other will encourage the CNN to extract features that are relevant to horse-looking animals.

We hypothesize that, in the extreme case where we see very few examples of a class (e.g., “unicorn”), its proximity to other word embeddings (e.g., “horse”) should provide a lot more information that would be completely lost with more traditional bag-of-words based approaches.

1.3 Recurrent Neural Networks

With the evolution of the neural networks the scenario of the deep learning completely changes as the RNN as it evolved with the intuitive application. The vanilla network leads to this significant change in RNN as it doesn't carry any predetermined vector size. In RNN type networks the weights defined are depend on the past generated parameters which influence the decision learnt on the current neurons from past inputs. For example, it remembers how a 3 looks like in p_i represents the occurrence probability of an event i with

$$\sum_{i=1}^{n} p = 1 \tag{1.2}$$

Unlikely to the traditional neural networks it can take the multi inputs while generating multi modal output without getting influenced from prior weights. Accordingly, it generates hidden state for the further network layers in order to generate multi nodal output from the same set of inputs which transform the probability distribution of the modal and not mugging up for same set of data.

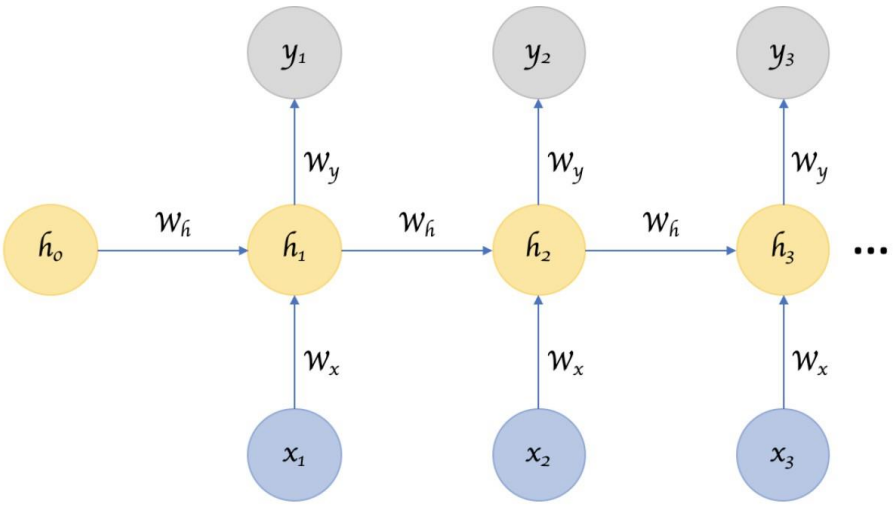


Fig. 9 LSTM Neural Networks [8]

Multiple weights are generated using RNN architecture in order to influence the set of input to carry multi weights assigning to the hidden layer in order to transform the output of the training set. This concept of RNN is known as Weight Sharing technique. By enabling this weight sharing technique in the RNN's the parameter will not be of constraint vector length so that the output will become leverage.

1.4 Deep Learning Algorithm with the Game Theory

The deep learning concept in neural networks acts as process of joining all neural networks together such that the parameters generated by each individual layer can be processed by the medium of deep learning. Intuitively this can be handled by DL so that the imperfect notations of the parameter can be tackled down by it. This can be seen by the Deep Mind Alpha Go which is proved to be best Go mind player which gets evolved through tactically advanced parameters of Deep Leaning.

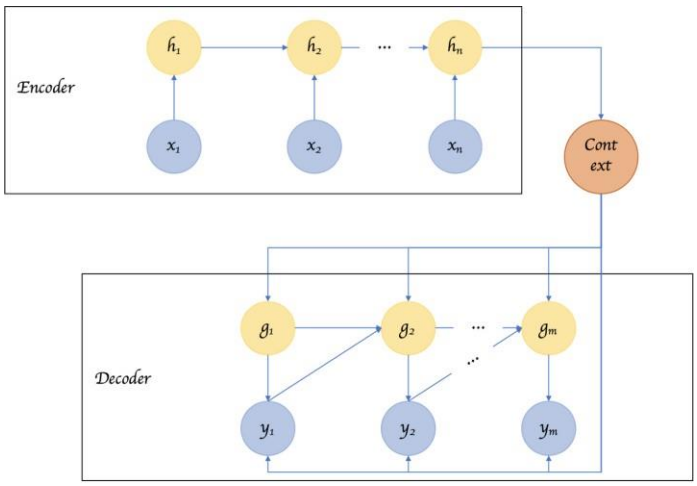


Fig. 10 RNN's Encoder and Decoder [10]

Further intuitive of deep learning can be judged by the monolithic steps of adversial network which would lead to multi coordinating of the trained parameters. Once it gets trained the network generate the fake instances of the dataset by augmenting them so that the model records the error and try to replicate the best possible network with the advancement of the game theory in deep science.

In the Game Theory of Deep learning, there are two idealized rules:

1. All the network determined by the model will be judged upon the score generated by the neural network.

2. The score calculated is further provided as a Q score formula to the multi layered of RNN so that it will be unsupervised and simply learn through the parametrized technique.
3. In this govern theory, they provided an elegant approach to the model so that it highlighted the strength and weakness of the model using graphing the parameters. In "Deep Reinforcement Learning from Self-Play in Imperfect-Information Games" they have proposed first end to end RNN technique using game theory which determine the imperfect game play without knowing the prior information of the domain [6].

$$r_{i,j} = \left\| x_i - x_j \right\| = \sqrt{\sum_{k=1}^{c} (x_{i,k} - x_{j,k})^2} \tag{1.4}$$

where $x_{i,k}$ is the kth component of the multi modal coordinate x_i of i^{th}

For a 2-D case, $r_{i,j}$ is given by Equation

$$r_{i,j} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{1.5}$$

This represents the vectorized distance between the points of vectors.

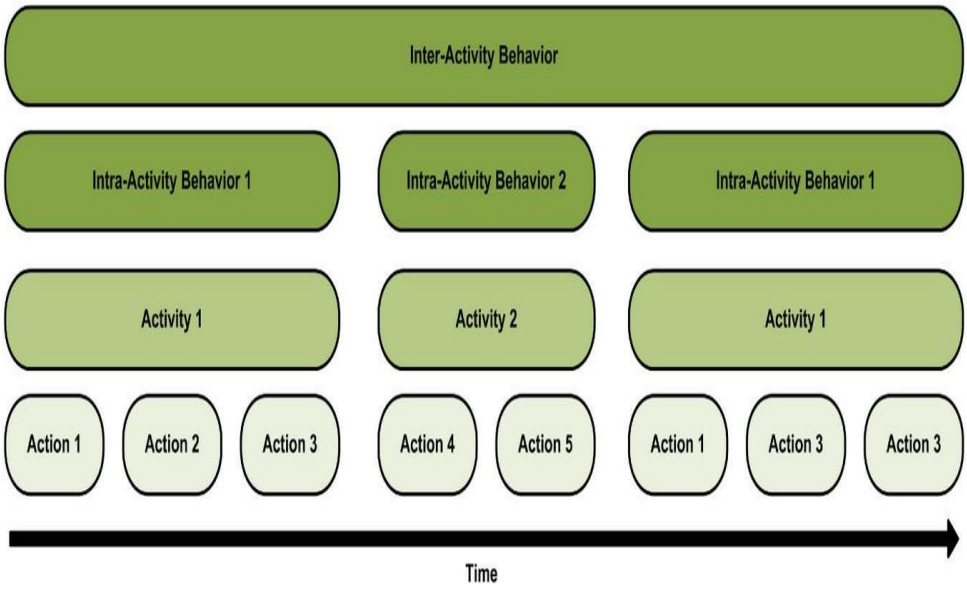


Fig. 11 Behavior of the Network

The above figure depicts the behavior of the network which learns the activity of the parameterized layered networks with the increment of time period. As we increase the epochs of the model to train over the dataset, the network continuously depicts the

behavior by modelising the parameters of the trained model. This auto learning of the network enhances the intuitive level of persistence of the model.

In this case the random information gain parameter is evaluated by Equation [1.6] as explained in [10]-

$$u_i^{max} = \left(\text{rand}2 - \frac{1}{2}\right) \tag{1.6}$$

where rand_1 = V(0, 1) and rand_2 = V(0,1) are the randomized parameters obtained through the probabilistic distribution . The model obtained can be generalized as the function of f(x).

$$La(MLE) = \sum ai \in S\ (logp(ai \mid Context(ai)))$$

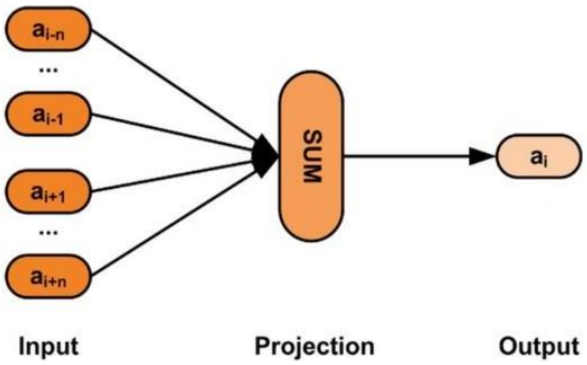


Fig. 12 LSTM Cell

CHAPTER 2

PROPOSED IMAGE SUMMARIZER

A robust technique has been implemented using CNN, RNN, LSTM and a part of game theory with the help of deep learning. These steps can be seen through the flow diagram that will depict the flow of technique which I have used to improve the computation of the image summarizer model. Before enlisting the technique in brief, I would like to enlist few points of the concept to highlight the process.

First, the host image has been sculpted as block system so that specific region of the image can be tuned (blocks); therefore, we decomposed the image into a kernel size block so that parameter tuning can be easily achieved. CNN characteristics will be used for the further steps of neural network so that it can be indulged into memory cells of the hidden layers. These two parameters will be then summed to find the weights and the bias of the feature variables. Therefore, the kernel with minimum value of entropy will be selected and used for the LSTM architecture. The kernel selected for the model will be of 5X5X3 for the process of model.

The second point is to calculate the algebraic parameterized values calculated by the RNN layered architecture so that vector length of the feature variable is of optimal value with getting convexing of the function. This helps the model to compute the feedback in the most intuitive fashion so that it can be summed up with current inputs.

The changing of last step of inclusion of game theory part of Q value helps to generate a score-based approach to increase the efficiency of the parametric steps (this can be seen in the explanation of the algorithm).

We consider the image I of size N x N. The image summarization procedure is given below:

Step 1. CNN model is applied on the whole image screen through the convolved kernels.

Step 2. The applied kernel carries out all the respected information of the image.

Step 3. The information generated from the image by CNN will be passed

through RNN Network to recursively learn the dependency of the pixel to the other pixel so that padding and shifting would be done and weights will be assigned.

Step 4. Caption information will be generated alongside with RNN network so that multiple captions are formed such that it will get saved as vector form in concurrent layers [2].

$$S` = S + \delta * S_w \quad \text{Where } \delta \text{ is a scaling factor.}$$

Step 5. The caption information generated will be fed into LSTM architecture with the image kernel so that LSTM cells get mapped with captions and image kernels.

Step 6. These mapping will be transferred to language model so that semantic modelling will be carried to the hidden layer. The embedded network of words and image kernel will be processed by LSTM architecture so that relationship will be developed between kernels and language model. This resultsinto deep image captioning.

Step 7. The language model will be trained in the stylized manner so that semantic model will become effective with every epoch.

Step 8. The language generation will be based on Step 1 so that phrases generation will be carried out by NLP model [2].

$$p_n = \frac{\sum_{C \in \{Candidates\}} \sum_{ngram \in C} Count_{clip}(ngram)}{\sum_{C' \in \{Candidates\}} \sum_{ngram' \in C'} Count(ngram')} \tag{2.1}$$

Step 9. Caption model will be dynamically iterative until the end state will be achieved by the caption model.

Step 10 The generated captions will be scored according to the game theory so that it will be ranked to select the best caption to summarize.

$$Q^{\pi}(s,a) = r(s,a) + \gamma V^{\pi}(\delta(s,a)) \tag{2.2}$$

2.1 IMAGE SUMMARIZED CHARACTERSTIC PROCEDURE

2.1.1 Image Embedding Procedure

- Feed the image dataset to the CNN COVNET model.
- Convolved the images with kernels of 9X9 shape. The neuron in the layer will learn the feature labels and assign with weights to them.
- Apply a loop function to the COVNET layer until it gets sampled to each

layer, then backdrop the error found and update the weights.

- ReLU function will be applied so that pixel of each pixel will be transferred as the probability function so that it will be distinguishable from other pixels.
- Examine the COVNET layer to rectify the feature maps so that pooling layer will be applied separately.
- Perform the error evolution from the given formula

Total Error = ∑ ½ (target probability – output probability)

- Perform the above steps iteratively over whole training set to obtain robust values of weight. Then the values are transferred to further model for mapping of language model.

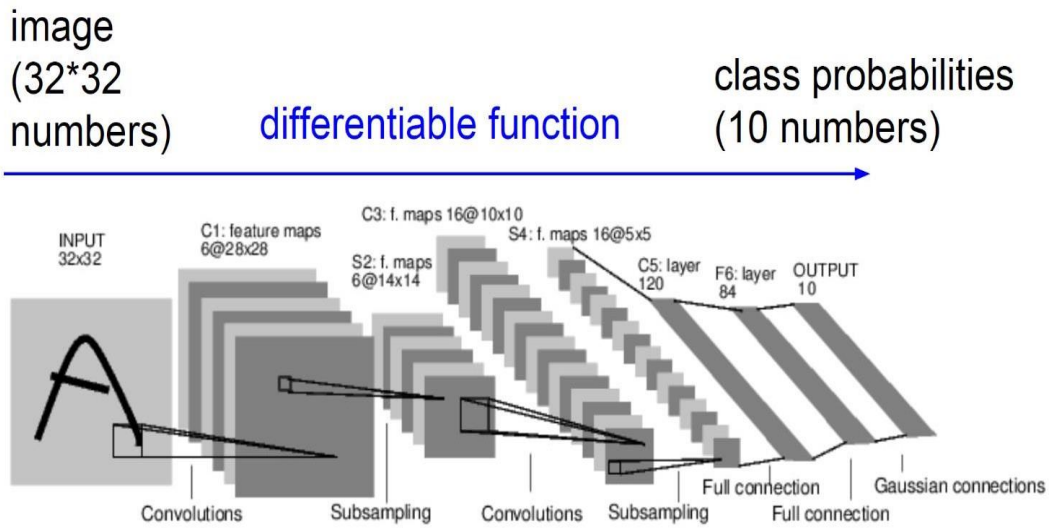


Fig. 13 Differentiable Function [12]

Time of serial execution: $(t_1+t_2)\times N+t_3=t_{serial}$

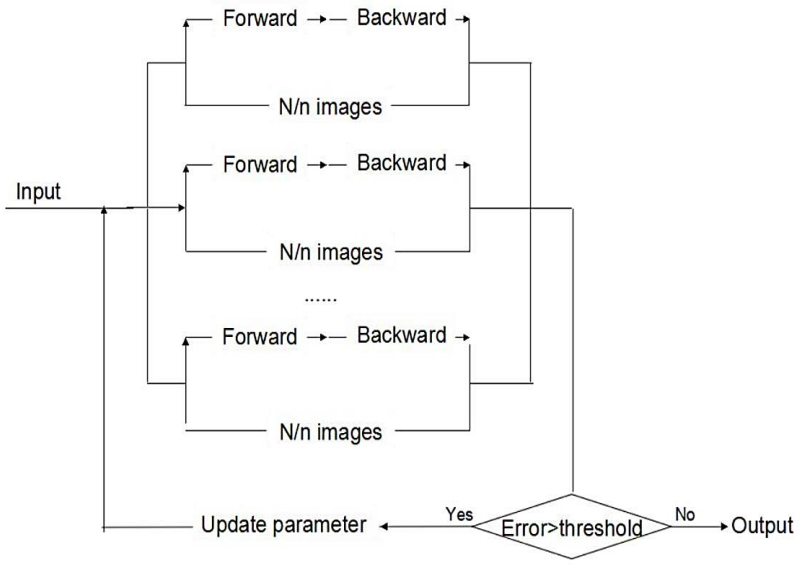


Fig 5 the parallel strategy

Time of parallel execution : $\max\{(t_1'+t_2')\}\times(N/n)+t_3'=t_{parallel}$

speed-up ratio $=t_{serical}/t_{parallel}$

Speed-up efficiency=speed-up ratio/n

N: num of images

n: num of nodes

t_1 :time of forward pass for training a picture

t_2 :time of backward propagation for training a picture

t_3 :time for updating weight and bias of convolution neural network

Fig. 14 LSTM Backpropagation [12]

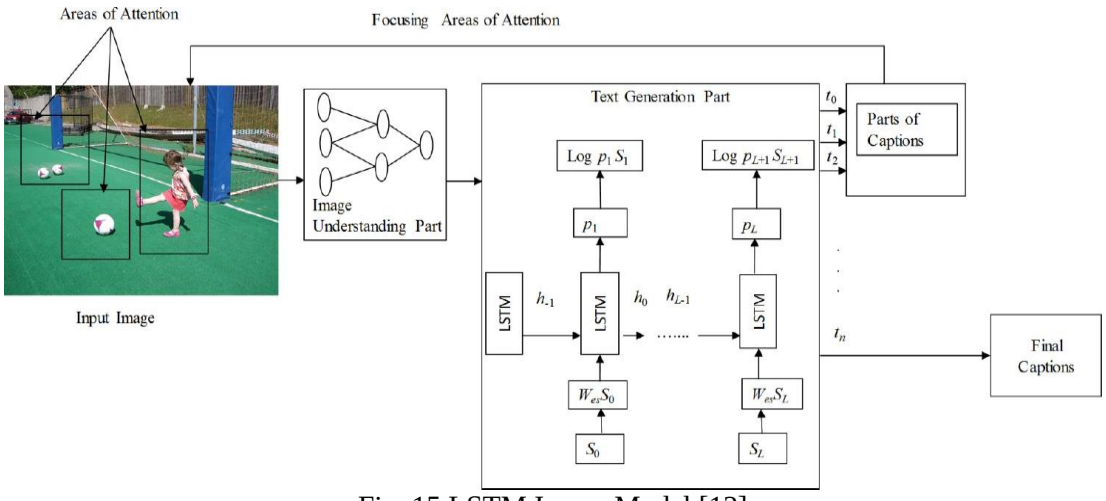


Fig. 15 LSTM Image Model [12]

2.1 Text Mapping procedure

- CNN model generates the information from the image and decompose it into LSTM-RNN model.

- LSTM-RNN model will generate the text features from the above Step 1.
- Salient feature portion will be focused in each iteration and get processed by language phase.
- Captions will be updated by the model by the ranking order of priority until the model gets updated.

2.2 PROPOSED PROCEDURE

This following scheme has been performed with three host images and two watermark images.

1. CNN model is applied on the whole image screen through the convolved kernels.
2. The information generated from the image by CNN will be passed through RNN. Network to recursively learn the dependency of the pixel to the other pixel so that padding and shifting would be done and weights will be assigned.
3. Caption information will be generated alongside with RNN network so that multiple captions are formed such that it will get saved as vector form in concurrent layers.
4. The caption information generated will be fed into LSTM architecture with the image kernel so that LSTM cells get mapped with captions and image kernels.
5. The language model will be trained in the stylized manner so that semantic model will become effective with every epoch. This will increase the effectiveness of captioning model efficiency. Find the minimum value of fitness value and its respective position.
6. Now, Caption model will be dynamically iterative until the end state will be achieved by the caption model.
7. Finally, the generated captions will be scored according to the game theory so that it will be ranked to select the best caption to summarize.

CHAPTER 3

RESULTS AND DISCUSSIONS

Several experiments are performed to enhance the efficiency of the model and to generate a robust output so that we'll get a well refined model at the end of our training. The training of the model has been performed on standardized and authenticated dataset. We have used COCO dataset of around 1.5 GB to train our model. Many ideas have been suggested for testing the accuracy of our model. Since it is a unsupervised model we have used a confusion matrix to estimate the performance of the model in this paper.

$$\mathrm{PSNR} = 10\log_{10}\left[\frac{\max(x(i,j))^2}{\mathrm{MSE}}\right] \tag{3.1}$$

Where i, j are the coordinates of each pixel of the host image x and the mean-square error (MSE) between the host image x and the watermarked image y is defined as follows [6]

$$\mathrm{MSE} = \frac{1}{m*n}\sum_{i=1}^{m}\sum_{j=1}^{n}[x(i,j) - y(i,j)]^2 \tag{3.2}$$

When good imperceptibility is achieved, the watermarked image appears nearly identical to the host image; in other words, we can say that the host image is not affected by the embedding process.

Accuracy (ability to differentiate the nodule and nonnodule cases correctly), sensitivity (ability to determine the nodule cases correctly), and specificity (ability to determine the nonnodule cases correctly) are used to measure the correctness of the classification. These metrics are widely used in binary classification problems and are defined as follows:where TP (true positive) represents the number of cases correctly identified as nodules; FP (false positive) represents the number of cases incorrectly identified as nodules; TN (true negative) represents the number of cases correctly identified as nonnodules; and FN (false negative) represents the number of cases incorrectly identified as non-nodules.

3.1 Result Description

The training of the model has been performed on standardized and authenticated dataset. We have used COCO dataset of around 1.5 GB to train our model. Many ideas have been suggested for testing the accuracy of our model. Since it is a unsupervised model we have used a confusion matrix to estimate the performance.

In the table we have shown the result of image and text in tow different table format that how we first processed the text data and with respect to it we captionized the image from raw image to a captionized format. The first section of the table consists of the text data that we processed from raw format to a vectorized format that can be understood by the model.

The text that is marked with the stopping and the ending point so that system will understand the beginning and the ending of the statement after that we passed it to the next stat where every individual statement or we can say the caption we vectorized format by genism model where the text is being processed under the removal of stop word, noise words with application of chi-square and other genism model. At last the cleaned textual data is passed to tf-idf vector model where the text is being arranged in numeric format with respect to dictionary rule and to weighted order.

Then in another table we can see the dataset image which is being passed to CNN model where COVNET will transform the image into pixel mapping of weights using kernel formatting, maxpooling and padding. This will generate a array of array which carry weights of the pixel which will mapped with textual data and LSTM layer will fed to this and get trained for the prediction.

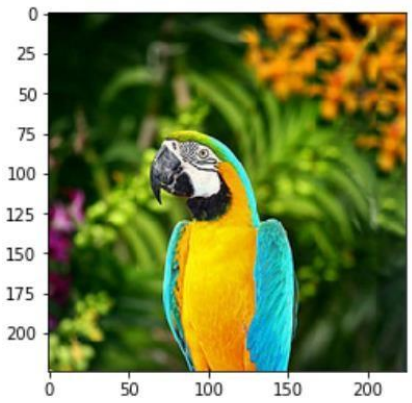
Recurrent Neural Network (RNN) controlled by shared Long-Term Memory (LSTM) parameters and produces a soft visual attention map. As shown in section 3.2, the exponentially large search complexity reduces this CAVP design to linear time. This stabilizes the conventional Monte-Carlo policy development by reducing search complexity.

CAVP and its network of subsequent language policy should also be pointed out that over time, relationships between subjects mentioned in the subsequent generation can be effectively modelled to higher-order .

Table 1 Final Processed Text after Filtering

['ssss Closeup of bins of food that include broccoli and bread. eeee', 'ssss A meal is presented in brightly colored plastic trays. eeee', 'ssss there are containers filled with different kinds of foods eeee', 'ssss Colorful dishes holding meat, vegetables, fruit, and bread. eeee', 'ssss A bunch of trays that have different food. eeee'] NLP LSTM Generated Text.
['Closeup of bins of food that include broccoli and bread.', 'A meal is presented in brightly colored plastic trays.', 'there are containers filled with different kinds of foods', 'Colorful dishes holding meat, vegetables, fruit, and bread.', 'A bunch of trays that have different food.'] Cleaned Trained Text
[[2, 841, 5, 2864, 5, 61, 26, 1984, 238, 9, 433, 3], [2, 1, 429, 10, 3310, 7, 1025, 390, 501, 1110, 3], [2, 63, 19, 993, 143, 8, 190, 958, 5, 743, 3], [2, 299, 725, 25, 343, 208, 264, 9, 433, 3], [2, 1, 170, 5, 1110, 26, 446, 190, 61, 3]] Vectorized Text through TF-IDF Tokenization

Table 2 Output of CNN Model

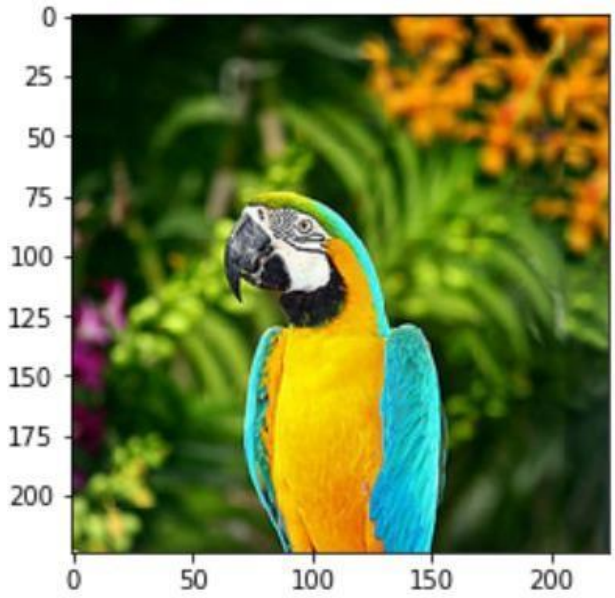


Scaled Training Image by CNN

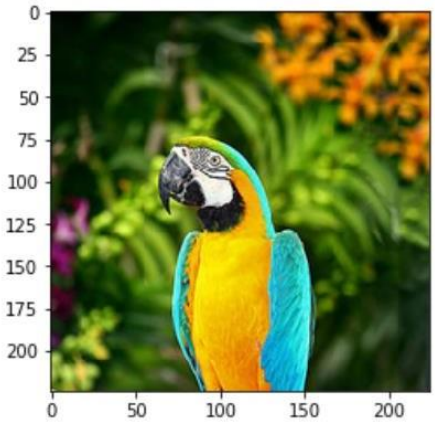
```
array([  2,   1, 126,  34,   5,   1,  29,  25,   1, 247, 116,   3,   0,
         0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
         0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0], dtype=int32)
```

CNN Vectorised Image

Table 3 Prediction Output of Model



Trained Input Image




```
Predicted caption:
 a small bird perched on top of a tree branch
```

Table 4 Prediction Output of Model

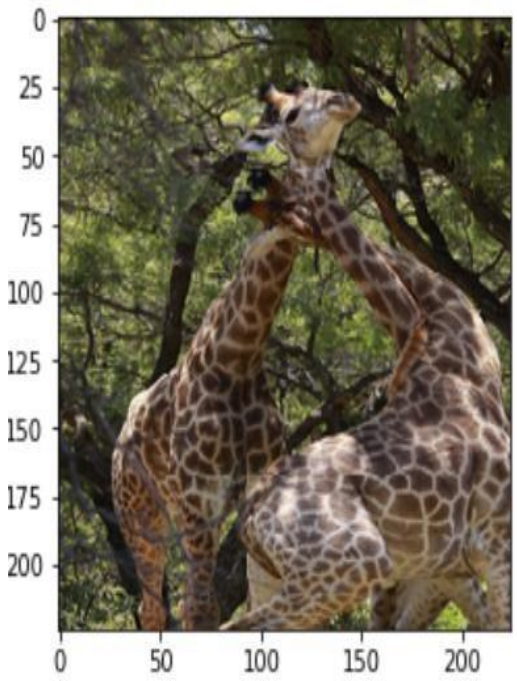
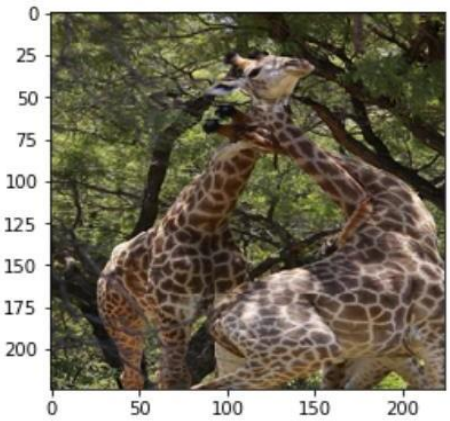


Image from COCO dataset




```
Predicted caption:
 a bird is standing in the grass near trees eeee

True captions:
A couple of giraffe snuggling each other in a forest.
A couple of giraffe standing next to some trees.
Two Zebras seem to be embracing in the wild.
Two giraffes hang out near trees and nuzzle up to each other.
The two giraffes appear to be hugging each other.
```

Table 5 Trained Parameters of CNN Model

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	(None, 224, 224, 3)	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1180160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2359808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0
flatten (Flatten)	(None, 25088)	0
fc1 (Dense)	(None, 4096)	102764544
fc2 (Dense)	(None, 4096)	16781312
predictions (Dense)	(None, 1000)	4097000

Total params: 138,357,544
Trainable params: 138,357,544
Non-trainable params: 0

Table 6 Trained Parameters of LSTM Model

Layer (type)	Output Shape	Param #	Connected to
decoder_input (InputLayer)	(None, None)	0	
hidden_state_input (InputLayer)	(None, 2048)	0	
embedding_layer (Embedding)	(None, None, 128)	3347840	decoder_input[0][0]
hidden_state_values (Dense)	(None, 300)	614700	hidden_state_input[0][0]
lstm_layer1 (LSTM)	(None, None, 300)	514800	embedding_layer[0][0] hidden_state_values[0][0] hidden_state_values[0][0]
lstm_layer2 (LSTM)	(None, None, 300)	721200	lstm_layer1[0][0] hidden_state_values[0][0] hidden_state_values[0][0]
decoder_output (Dense)	(None, None, 26155)	7872655	lstm_layer2[0][0]

Total params: 13,071,195
Trainable params: 13,071,195
Non-trainable params: 0

Above two tables consist of the parameter table of the CNN and LSTM layers. These parameters are obtained after generating the model and trained on the given dataset of images and annotations. In CNN network we can see the number of parameters increasing drastically due repeated CONV2d layer which get trained on the image with the kernel section so that it will add more number of filters to the image which can be used for image sharpening, increasing its resolution so that more information can be extracted out.

Another table is of LSTM parameters which consists of decoder, it can take the multi inputs while generating multi modal output without getting influenced from prior weights. Accordingly, it generates hidden state for the further network layers in order to generate multi nodal output from the same set of inputs which transform the probability distribution of the modal and not mugging up for same set of data. The main motive of the LSTM layer is to train the network till the second last layer so that last layer can be naived one.

Fig. 16 Loss Vs Epochs Plot

Above graph is the plot of the LOSS vs Epochs of the model that is captured while training the neural networks. We can clearly seas the number of epochs are increasing the loss of the neural network is decreasing it is due to batching used in the neural networks while training. The batching method randomly divide the data into batches of fixed length data so that in every epoch model will get some random batch. This technique of batch formation prevents the overfitting of the model hence the model will get trained efficiently with underfitting or overfitting of the parameters.

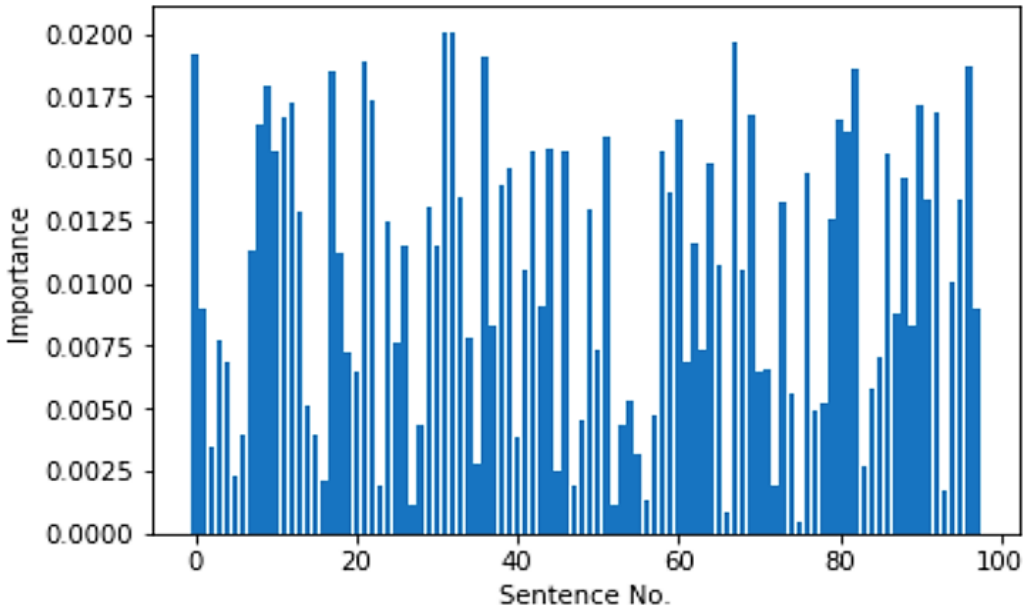


Fig. 17 Caption Priority Plot

This graph depicts the sentence number with the normalized importance of it in the y axis. This helps the model to assign the caption on the basis of priority level to the feed image to the model. As we may have some hidden features in the image which can't be depicted by naked eyes. So this priority level technique helps the model to assign most prior caption to the input image.

CHAPTER 4
APPLICATION

4.1 METHOD DISCUSSED

Above proposed method is being applied on the medical images in order to robust our method so that we can make

some prediction in the emerging researched field of medical science. Here, we have used the dicom images of CT Scan, MRI images of lung bisection images dataset with the annotations of labels of the nodules of lungs. Nodules are the piece of cancer spotted in the lung. Since its sometime difficult for the surgeon to operate the cancer part without knowing the location and the coordinates of the nodules from the images which leads to the inefficient operating process for a patient. The process of nodule prediction with its approximate co-ordinate region will be explained in the following steps-

1. The images we have fetched from the dataset are in the dicom format. Dicom format is the image format that is generated by CT Scan or MRI machines which are not readable by general computers. So, we have converted these images into system readable and processable format.
2. After the conversion we will input these images to our neural net model where it will be processed under U-NET's so that the nodule region will be traced by the model using the down sampling and then restoring the image by up sampling of it. So that we can generate the nodule mask for the general lung images.
3. Then these nodule mask will be processed with matplotlib library with respect to the annotations of the lung images so that the co-ordinates will be generated for the lung nodules as a part of feature engineering in order to generate our training set for the proposed model.
4. At last this training set will be passed to our previously trained model along with the converted lung cancer prone images so that our proposed model will be able to learn the relation of the image with its corresponding annotations.

In order to increase our computation of the network and decreasing the resource allocation while computing with images we have feezed all the layers except the second last layer so that the neural net will train the last layer in order to learn

relationship of the provided dataset.

During experimentation, we have found that if we deploy models on 32x32x32 mm crops of nodule images then they get trained much faster and generate much better accuracy. However, when we want to apply that model on actual full scan image then we have to evaluate it something large computations like 3000 times. Every time the model is get evaluated for the location then there is a high chance of a false positive, so this leads to large number of evaluations is not desirable. And on other side 64x64x64 models, will leads to take longer which isn't quite as desirable at describing nodule's information but this approach ultimately works better. On comparing the two, lesser evaluations than the 32 sized model while only less accurate.

Nearly all the nodule identification literature and almost every forum tutorial recommended that the lung tissue be segmented from CT scans. Neither segmentation strategy, however, was sufficient to manage nodules and masses concealed by the lung tissue edges. Sometimes the pictures were taken away and the nodule detector could not be found. I thought first of a U-net learning to segment the lungs correctly. This would almost certainly have better results than traditional techniques of segmentation. However, when a human examined the CT scans, lung tissue borders provided me with a good reference frame to find nodules.

The train datas are given and not interesting to discuss in a more straightforward competition. But this solution was a significant, if not most important component of the engineering trainset. I used labels given, automated labels created, actively learned automatically and manually added annotations. Although the trainset was stronger, it still took considerable tweak to train a neural network effectively. The dataset was very unbalanced still (5000:500000) and the size and form of the positive examples varied significantly.

I had suggested a U-net architecture, but 2D U-networks could not use the intrinsic 3D structure of the nodules. The primary reason for skipping U-nets was that a thin grained probability map was not necessary, but only a rough detector. It was much lighter and more versatile to have a tiny 3D convnet over the CT scans.

But it leads to reasonable question to raise - why not bigger than 64? Well we tried this too. It leads out to 64 soft spot sort. That our models rely priorly on extracting the symmetries of 3D space image. Well on experimenting the images we figure out that the lungs in general aren't that symmetric. So if we rely on making the chunk size larger then augmented data becomes less effective. We figure out that a chunk size of 128 *could* work.

One of the nice things about this architecture is that we have used was the model that can be trained on any sized input image (32x32x32 in size). The reason is because the last most pooling layer in the model is a golbal max pooling layer which generates an output fixed length output doesn't matter the input size.

Ronneberger et al 2015,[36] listed the contracting direction of a U-Net system (left side). The same as in the latter section (3x3 conv with ReLU and 2x2 max pooling), the standard Convolutional Networks. As previously mentioned, the great idea is that the network can create features that we want to detect patterns (in this case its lung cancer). The trouble with this is that the width of the object is defined by convolutionary layers. The only layout of the U-Net is in its increasing direction (right side) consisting of up-convolutions (2x2 sizes) and merge layers. The information is therefore lost as they eat. The Upgrades .The strategy we used here works when the predictions performed are both more *accurate* and largely *diverse*. Having the prior knowledge of this technique, we spent the last of the trying to build lots of diverse models with different optimised weights which were as accurate as other generated but used differently parameterised (to enable diversity).

- the data subset the model was trained on (randomly 75%)
- activation function (relu/leakly relu mostly)
- loss function and weight on loss function
- training length/batch
- model layer parameter
- model connection

Ultimately the 'ensembles' of models. The first ensemble we generted really on an ad-hoc criteria - during which the process of creating the neural network structure we

trained on the bunch of models. Most of them turned out to have replicating performance, so we dump them all into an ensemble. This ensemble had a CV score of 0.41.

4.2 RESULTS OF APPLICATION PART

The majority of every dicom scan is not seems to very useful in diagnosing lung cancer. There are various reasons for this, but the most common is that many of the scan data is covering the outside locations of the lungs. Hence the typical scan before is being cropped. The lungs are the big black spaces and the large portion of the scan doesn't get overlap with the interior of the lung at all.

The Lung Trace is the processed image from the dicom dataset into system process able format so that our model functionality can be imposed on it. After the implementation of the model we have generated the Nodule Trace i.e. the masked image of nodule area generated by the model and further it gave the co-ordinates of the nodule.
After discarding the majority of the dicom scan, it's time to use the big ensembles of neural networks to be trained. For each block idenfitied as 'abnormal' by the prior step, we run it through each of the neural nets models.

Because each model was trained with different parameters, batches, and objectives, every model gives a bit different prediction. Also every model is referring each block often number of times with some random transformations applied which we dicussed in the augmentation section. Predictions are taken as the averaged across the transformations but not across models.

Here we have found some examples which turned out to have malignant nodules. These are colored based on how important each part of the block is to the malignancy prediction for the entire block.

The output of this stage is one prediction per model per suspicious region in the image. These become the inputs to the next part of the pipeline which produces the actual diagnosis.

Forming a diagnosis from the CNN model outputs turned out to be quite easy. Remember at this point we have model predictions of several attributes (nodule malignancy, size, spiculation) at *many different places in each scan*. To combine all these into a single diagnosis, I created some simple aggregates:

- max malignancy/spiculation/lobulation/diameter
- stdev malignancy/spiculation/lobulation/diameter
- location in scan of most malignant nodule
- some other clustering features that didn't prove useful

These features are fed into a linear model for classification. Below is a feature importance plot, with the Y-axis showing the increase in log-loss when the specified feature was randomly scrambled:

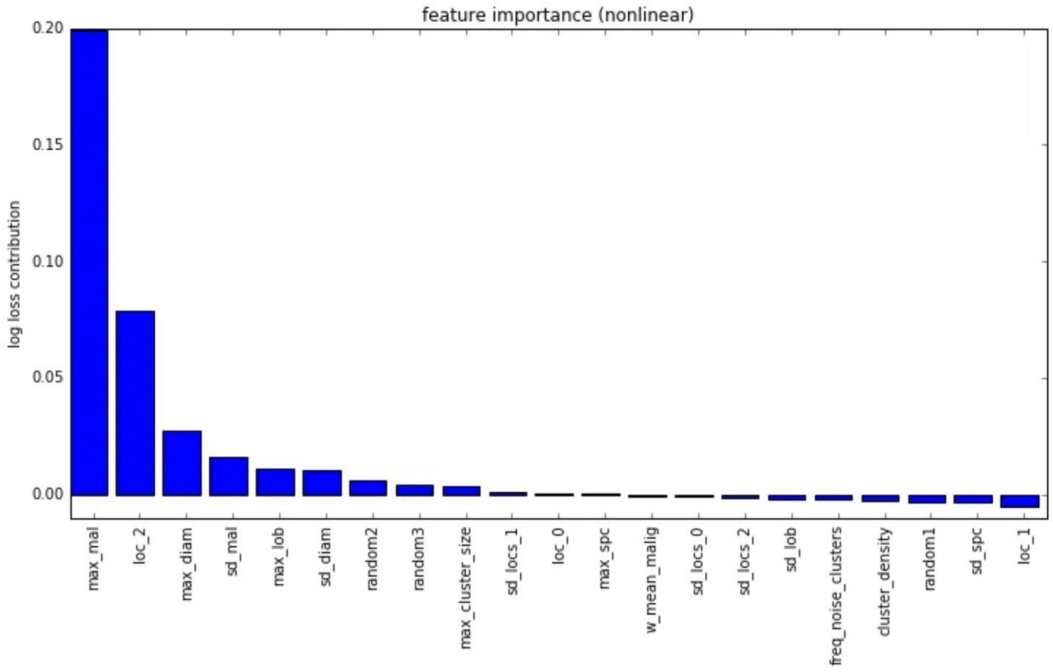


Fig. 18 Feature Importance

The graph plotted above depicts the log loss of the feature used while training the model. This can be seen as feature importance plot also. It's fairly clear from this that the max malignancy prediction is the most important feature. Another very important feature is the location of the most malignant nodule in the Z dimension (i.e. closer to head or closer to feet). This is also a finding that I saw in the medical literature so it's pretty neat to independently confirm their research.

Even with the better trainset it still took considerable tweeking to effective trained a neural network. The dataset was heavily unbalanced (5000:500000) and there was much variation in size and shape of the positive examples. We have considered U-net architectures but 2D U-nets cannot exploit them inherently 3D structure of the nodules and 3D U-nets were quite slow and inflexible. The main reason to skip the U-nets id that it is not prior to have a fine-grained probability map instead of it just a coarse detector. Having a small 3D convnet that you slide over the CT scans was much more lightweight and flexible.

Hence the goal was to train a working nodule predictor. The first thing we perform was to upsample the positive examples to a ratio of 1:20. For generalization a number of augmentation strategies were tried but somehow only loss-less augmentations helped. In the end I used heavy translations and all 3D flips.

Once the classifier is in placed we wanted to train a malignancy estimator. The provided malignancy of labels is ranged from 1 (very likely not malignant) to 5 (very likely malignant). To sllow more weight on our malignant examples we have squared the labels to a range from 1 to 25. At the beginning we are thinking about a 2 stage approach where first nodules will be classified and then another network will be trained on the nodule for malignancy. To decrease the time we have tried one network to train both at once in a multi-purpose learning approach. This worked perfectly and since the methodology used is quick and simple we havedecided to go fo this.

In order to meet the training stage for convolutionary neural networks a large number of positive and negative samples are normally required. Therefore the following augmentatory techniques are used for the generation of further lung nodulae for the training of the model:

(i) rotation from -25 ° to 25 ° with 5 ° step

(ii) vertical twisting

(iii) twisting of the model, both horizontally and vertically, Because of the importance of the nodule for the detection process, uniform transformations

such as stretching or skews should not be applied We have 16,189 optimistic patches after raise (they are discarded which are too close to the edges).

Usually the architecture of the neural network is one of the most important outcomes of a competition or case study. For this we spent relatively little time on the neural network architecture. We take this is mainly that there are already existing good baseline architectures to compare with.

We begin with out with some simple VGG and resnet-like architectures. All performed utterly the same. Then we liked to try a pre-trained C3D network. The pretrained weights can't not help at all but the architecture with training on our dataset thoroughly pretrained weights gave a optimised and good performance.
The final architecture was basically C3D with a some few modification.

The first adjustment is the receptive field which we set to 32x32x32 mm. This might like a bit too small but it worked very good with some tricks later in the pipeline. The idea was to keep everything lightweight and make a bigger net on the end of the pipeline. But network was 64x64x64 mm we try to stay at the small receptive field so that we were as complementary as possible. The second adjustment we did was to immediately average pool the z-axis to 2mm per voxel. This made the net much lighter and did not effect accurracy since for most scan the z-axis was at a more coarse scale than the x and y axes. Finally we introduced a 64 unit bottleneck layer on the end of the network. The was to do some experiments with training on the raw intermediate features instead of the predicted malignancy later in the process.

After education, the next step was to have the neural network classify nodules and assess their malignancy. I found the CT screen very useful for displaying the tests. My conclusion was that for the neural network it's an impressive job.

I missed a lot of nodules, although I saw only a few false positives. There was only one serious problem. There was a lack of some very large nodules. Occasionally 3.00 logloss of false negatives were the worst score. Two occasions in a small experiment I wanted to downsample scans to see if the scanner would then pick up the big nodules. It worked outstandingly well.

Table 7 U-Nets Parameters for Detection

Layer	Params	Activation	Output	Remark
Input			32x32x32,1	
Avg pool	2x1x1		16x32x32,1	Downsample z-axis
3D conv	3x3x3	relu	16x32x32,64	
Max pool	1x2x2		16x16x16,64	Axes are same again
3D conv	3x3x3	relu	16x16x16,128	
Max pool	2x2x2		8x8x8,128	
3D conv (2x)	3x3x3	relu	8x8x8,256	
Max pool	2x2x2		4x4x4,256	
3D conv (2x)	3x3x3	relu	4x4x4,512	
Max pool	2x2x2		2x2x2,512	
3D conv	2x2x2	relu	1x1x1, 64	Bottleneck features
3D conv	2x2x2	sigmoid	1x1x1, 1	Nodule detector
3D conv	2x2x2	none	1x1x1, 1	Malignancy estimator

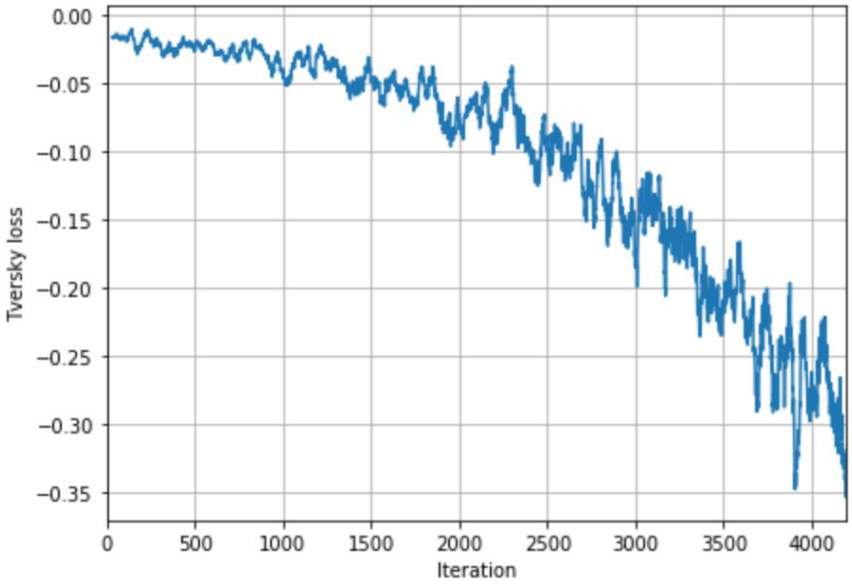


Fig. 20 Log Loss for Nodule Prediction

It may take up to several days for the entire training process. However, you can enjoy decreasing loss even after several epochs (several hours of training). It can be seen the loss is being decreasing slowly during the initial iterations as the number of iterations are increased the loss function decreases exponentially.

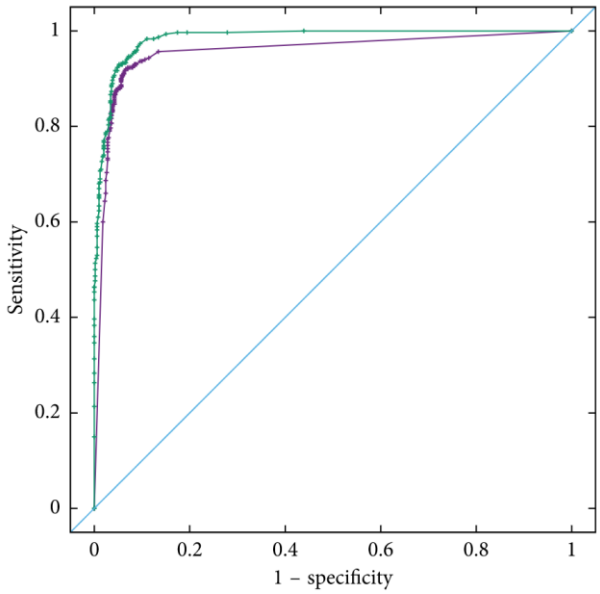
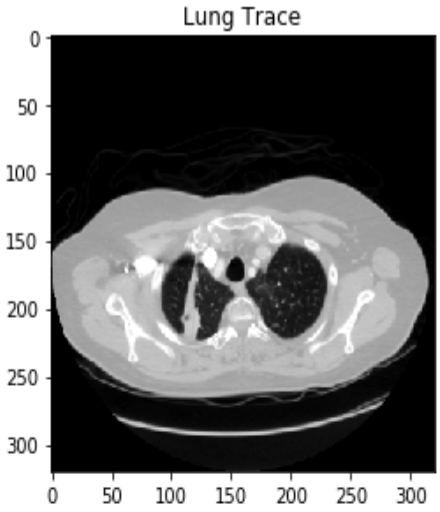
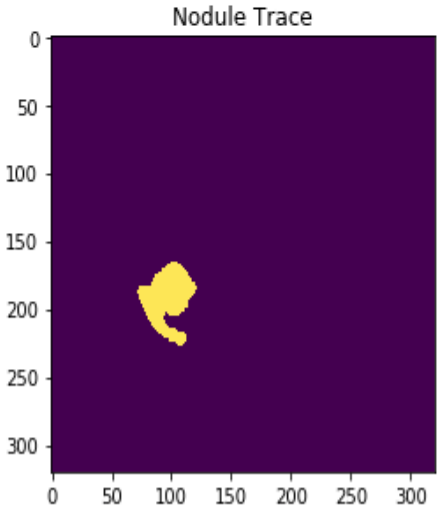


Fig.21 AUC/ROC

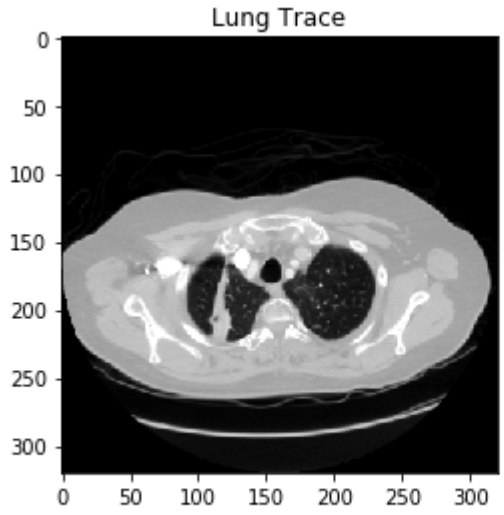
ROC curve enables the performance of a binary classification system to be evaluated by analyzing the relationship between real-positive (TPR) and false-positive (FPR) rates. The classification threshold changes are measured by TPR and FPR in detail. A ROC curve is then generated for each pair of (TPR, FPR). In particular, high TPR and low FPR have a strong classifier; therefore its ROC curve moves to the top-link corner. Figure 21shows the ROC curve of our using the 0 to 1 identification limit in a phase of 0.01, using both focused failure and cross-entropical loss. The figure shows that the ROC curves are very close. The AUC stands for separation which indicates the difference between the positive and the negative classes of the classifying system. The AUC values with cross-entropy losses and focal loss are summarized. We are a high-grade classification with a 98.2 percent AUC value.

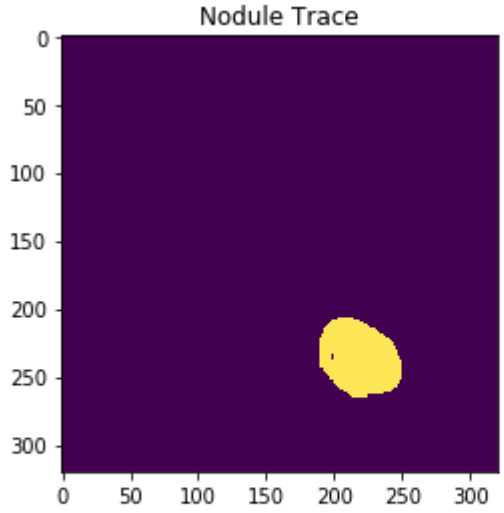




	coordX	coordY	coordZ	diameter_mm
0	-5.325108	-33.87652	-217.342132	9.233994

Fig. 22 Predicted Image





	coordX	coordY	coordZ	diameter_mm
0	54.866501	83.421341	-200.520805	15.99762

Fig. 23 Predicted Image

CHAPTER 5
CONCLUSION

In this thesis, we have tried many characteristic approaches in order to achieve high accuracy and robust result for the project of image summarization technique. But after many hit and trials we found the combination of CNN, LSTM-RNN model with the game theory concept of award generating procedure has provided best performance in our case.

The proposed process is somewhat similar to all previous traditional methods and concepts. In term of performance we look out that the previous methods performed quite well with high efficiency but with the introduction of award generating process the priority level has been introduced to the model. This helps the model to judge the captions according to ranking of the sentences with the image information.

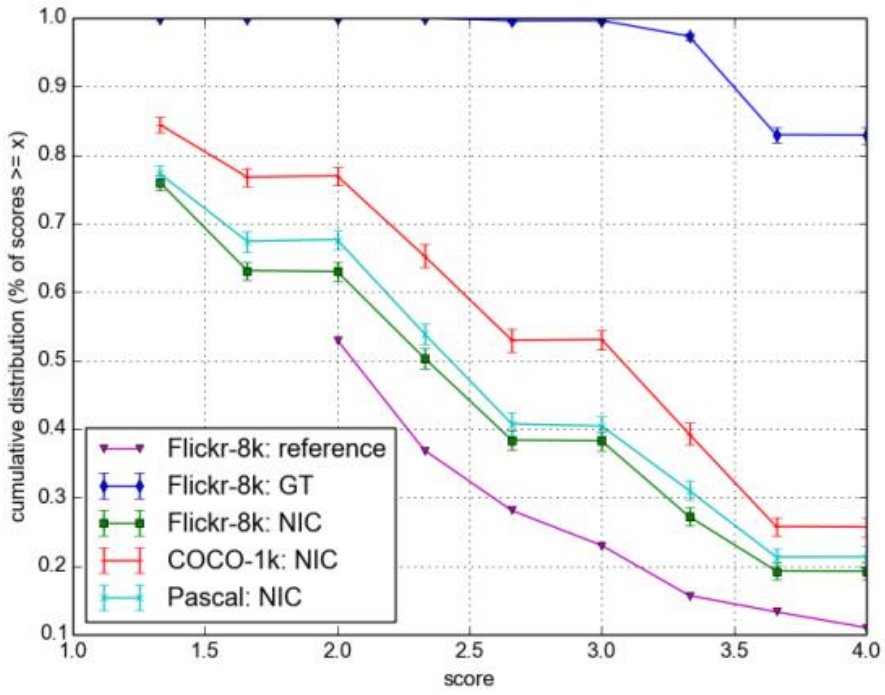


Fig. 24 Dataset Accuracy

Here we also tried to compare the accuracy of the trained model on various available public dataset. Here we noticed that the COCO dataset performed better in comparison to other datasets. It is due to the diversity of the image provided by the COCO dataset and the text annotation defined for every image is more précised.

From the beginning of the project we have seen the trend of epochs and the priority

score on the output results of the projects. On increasing the epochs, the accuracy of the model increasing. This is due to the learning more salient features of the image by LSTM model. Also taking the large dataset helps the model to get trained over high permutation of the feature variable. Also, we have seen on increasing number of parameter the model become more complex and can learn large number of feature variables which establish the relationship between the image feature vector. This helps the model to trained itself to learn mapping pixel with the language corpse. And this sequence of mapping with image and floating-vectors of token vectorizer to generate the sequence of token to generate the captions.

REFERENCES

1. X. Chen and C.L. Zitnick, Mind's eye: "A recurrent visual representation for image caption generation." *In Proceedings of the IEEE conference on computer vision and pattern recognition.* Page. 2422–2431, 2015.

2. R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation." *In Proceedings of the IEEE conference on computer vision and pattern.* Page. 580–587, 2014.

3. E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax." *In International Conference on Learning Representations (ICLR),*Page. 630-640, 2016

4. K. He, X. Zhang, S. Ren, and J. Sun. "Deep residual learning for image recognition." *In Proceedings of the IEEE conference on computer vision and pattern recognition*. Page. 770–778, 2016.

5. S. Bai and S. An. "A Survey on Automatic Image Caption Generation. Neurocomputing", Page. 820-823, 2016.

6. A. Aker and R. Gaizauskas. "Generating image descriptions using dependency relational patterns*." In Proceedings of the 48th annual meeting of the association for computational linguistics. Association for Computational Linguistics*, Page. 1250–1258, 2010.

7. X. Chen and C.L. Zitnick, Mind's eye: "A recurrent visual representation for image caption generation." *In Proceedings of the IEEE conference on computer vision and pattern recognition.* Page. 2422–2431, 2015

8. Available online Data Science Bowl dataset Lung cancer data and Nodule annotation."https://wiki.cancerimagingarchive.net/display/Public/Data+Science+Bowl+2017"

9. Available [online] COCO dataset of captioning dataset, "http://images.cocodataset.org/zips/train2017.zip"

10. Available [online] https://www.analyticsvidhya.com/blog/2019/09/demystifying-bert-groundbreaking-nlp-framework/

11. Available [online] "https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148"

12. Available [online] "https://medium.com/mlreview/understanding-lstm"