Second Review Document

MEDICAL REPORT GENERATION ON CHEST X-RAY IMAGES

R.S. Rahul Sai
17BCE0136
9131038763
rs.rahulsai2017@vitstudent.ac.in

Prof. Sharmila Banu K (11989)
Associate Professor Grade 1
9942469321
sharmilabanu.k@vit.ac.in

B.Tech.

in

Computer Science and Engineering

School of Computer Science & Engineering

1. Introduction

1.1 Theoretical Background

Computer Aided Diagnosis (CAD) and medical imaging systems have evolved in the past decade to a point where they can partially mimic radiologists and doctors. These systems can learn and differentiate the features and abnormalities in medical images, and provide objective evidence with a higher diagnostic confidence and faster inference. In this project, we focus on generating detailed medical reports on chest X-ray images, which can be adapted later to work with other diagnostic tools such as ultrasounds and mammograms. The Indiana University dataset provides us with CXR images corresponding to various lung and heart ailments, along with well-defined reports and findings. The generation of medical reports mainly consists of two broad tasks. The first task is to treat the problem as a multi-label classification task to obtain the accurate tags for a particular image from the visual features. This is performed using Convolutional Neural Networks and a transfer learning framework called ChexNet, which is specialized for chest X-ray images. The second task is to generate the reports using these aforementioned tags, which requires the use of recurrent neural networks such as hierarchical LSTMs. To improve the quality of the sentences produced, a co-attention mechanism is also required, which makes use of the spatial information from the convolutional layers and the generated words in order to find localized regions from which the abnormalities are found.

1.2 Motivation

The use of deep learning in image captioning has been a popular use case over the past few years. It is a challenging problem as it requires the machine to generate textual description from the contents of an image, similar to how a human brain would describe an image. But consider the same scenario for medical images, with the example of Chest X-ray images. For a normal human eye, chest X-ray images are just images with the skeletal and muscular features of the lungs defined in black and white. But highly trained radiologists who have studied and diagnosed various respiratory and cardiovascular abnormalities, can mark multiple areas of the images and can write down clear reports for potential abnormalities. However, to read a chest X-ray image properly, it is important that the radiologist has a thorough knowledge of the human thorax, and how various respiratory diseases might affect them. This comes with multiple years of experience by analyzing chest x-ray images with a fixed pattern of evaluation, and evolves over time depending on the history of cases a particular radiologist may handle.

But, even for highly trained and experienced radiologists, writing reports is highly time-consuming especially in regions with higher population density. Looking at the other side of the spectrum, radiologists or pathologists in rural areas, with inferior imaging equipment face a similar issue. They either cannot get objective evidence to diagnose anything properly or lack the knowledge of the respiratory or cardiovascular abnormalities. Misdiagnosis of symptoms and medical errors are the third-leading cause of deaths across the world, leading to more than 3 million deaths. So, this project focuses on generating detailed medical reports using Chest X-ray images, which can facilitate the diagnosis of various respiratory and cardiovascular diseases. The dataset to be used in this project is the Indiana University Chest X-Ray (CXR) Image dataset **[1]** . It is a high resolution CXR dataset with multiple views i.e., frontal, side and posterior views. There are 7,470 images accompanying 3,955 well written reports encoded in XML. These XML reports have references to the CXR images, the findings, impressions and the indication from the CXR images. These CXR images are obtained from patients diagnosed with tuberculosis, pneumonia and various heart ailments.

1.3 Aim of the proposed work

To generate detailed medical reports with three heterogenous sections i.e., Findings, Impressions and Indications from IU Chest X-Ray images, highlighting the context of the particular disease i.e., location, severity and affected organs, using a combination of CNN, hierarchical LSTMs and co-attention mechanism.

1.4 Objective(s) of the proposed work

i. Preparing an NLP Pipeline for the original findings, impressions and indication
   a. Removing Contractions, Punctuations and Numbers
   b. Tokenization
   c. Representing the reports in word embeddings
ii. Obtain the image features using a Convolutional Neural Network and the Transfer Learning framework i.e., ChexNet [2] (acts as the encoder here).
iii. Generate the text using the labels/tags obtained from the encoder using Hierarchical LSTMs which act as the decoder in our Sequence-to-Sequence Model.
iv. Substituting Attention mechanisms [3] to improve the encoder-decoder approach and compare the results.

2. Literature Survey

2.1 Survey of Existing Models/Work

In [4], the authors discuss the viability of a cascade model for medical image captioning where they cascade CNNs and RNNs over multiple steps. The first step is to train the CNN with the images and predicting single object labels, and the RNN to describe their context from the text. The second step carries over the weights from the previous step and introduces a mean pooling layer in order to derive the image/text context labels from the image/text contexts of the previous step. The type of RNNs used in this framework are often used sequence generation types i.e., LSTMs and GRUs, which use the input image's context vectors (in the form of CNN embeddings) in order to learn the annotation sequence.
The concept of using semantic attention is discussed in [5] where the authors first discuss the top-down paradigm (start from the high level features of the image and come up with words) and bottom-up paradigm which starts with words which describe various features of the image and combines them to form a coherent sentence using language models. Both paradigms suffer from their own weaknesses such as lack of attention to fine details in the top-down approach and the lack of end-to-end procedures from the individual features to sentences in the bottom-up approach. They suggest an idea that visual attention plays a major role of offering feedback which can help combine both the top-down and bottom-up information. Visual attention can be defined briefly as the mechanism in our visual cortex which tends to look at the low-level and semantic details of the image which it considers more important rather than the whole image. Their approach involves detection of semantic attributes using the bottom-up approach, which they call candidates, and then they employ a top-down approach in order to select which candidates should require more attention in order to yield better results. Their framework outperforms competing methods across various evaluation metrics such as BLEU and Meteor.

A similar concept of caption generation is discussed in [6][7] where the authors discuss the concept of scene understanding with the help of visual attention. They propose two techniques under a common framework, one being a soft and deterministic attention mechanism and another being a hard stochastic attention mechanism, which can be trained by maximizing a convergence function. A CNN extracts a 14x14 feature map, which is then processed by a RNN with visual attention over the image which provides a context vector. This vector is processed by a word LSTM which generates a word-by-word caption by utilizing a greedy search mechanism.

2.2 Summary/Gaps identified in the Survey

Most papers in the domain use word embeddings such as GloVe word embeddings [8] which has a certain disadvantage due to the lack of medical terms in the vocabulary. The GloVe word embeddings are trained on a generic text corpus, which is not. For resolving this, BioWordVec [9] can be used which are a type of new biomedical word embeddings.

The encoder-decoder or sequence-to-sequence model, due to the lack of localized visual context, couldn’t provide accurate predictions. Most of the predictions generated by the simple Encoder-Decoder model are correct when the case is related to cardiopulmonary abnormality, but other tags are predicted wrong. This can be solved by using a Global Attention Layer [3].

3. Overview of the Proposed System

3.1 Introduction and Related Concepts

A medical report can usually consist of various parts, which are usually heterogenous in nature. There may be images, abbreviations and complicated terminology in these reports. To avoid this issue, we will focus on three sections from a medical report i.e., findings – a large paragraph which contains keywords and parameters, indication – the doctor’s advice for the patient and impression – a sentence finalizing the results for a report. To achieve this, we propose a multi-task architecture which works by predicting the keywords/tags (findings) by treating it as a multi-label image classification task and generate longer descriptions by using a text generating mechanism such as hierarchical LSTMs. Hierarchical LSTMs [10] are specialized recurrent neural networks which are often used for text generation from images and video frames. It is built to consider both high-level language features from the training text and low-level visual features obtained from the processed image.

The dataset to be used in this project is the Indiana University Chest X-Ray (CXR) Image dataset. It is a high resolution CXR dataset with multiple views i.e., frontal, side and posterior views. There are 7,470 images accompanying 3,955 well written reports encoded in XML. These XML reports have references to the CXR images, the findings, impressions and the indication from the CXR images. These CXR images are obtained from patients diagnosed with tuberculosis, pneumonia and various heart ailments.

3.2 Architecture for the Proposed System

The architecture of the proposed medical report generation system can be divided into three distinct parts:

NLP Pipeline – The findings, impressions and the indications obtained from the reports have to be properly cleaned to be used in the model. This involves the following steps:

a) Converting all characters into lowercase
b) Removing contractions from the text e.g., won't – will not, can't – cannot.
c) Removing punctuation from text with the exception of full stop, as the findings from the reports may contain more than one sentences.
d) Removing all numbers and redacted data from the text.
e) Removing smaller words and adverbs with the exception of "no" as it adds significant value. e.g., adverbs such as "there", "then".
f) Tokenization and addition of identifier tokens such as "_start" and "_end" tokens which are necessary in the text generation process.

Examples –

Findings before the cleaning process:

No pleural effusion or pneumothorax. No acute bone abnormality.

Findings after the cleaning process:

_start no pleural effusion no pneumothorax. no acute bone abnormality. ***_end***

Convolutional Neural Networks – In order to extract the visual features of the image, we make use of convolutional neural networks. The CNN acts as the encoder in our model, and provides us a feature vector with the visual features of the image, which can be used to predict the tags for that particular X-ray. This is done by considering the problem as a multi-label classification task where the output layer provides you with the probability of a set number of tags. These predicted tags help us massively in text generation which is discussed in the next section. However, the size of our dataset (7,470 images) is not enough to train a CNN properly. To remove this bottleneck, we have to use a transfer learning framework. Most transfer learning frameworks such as VGG16 or InceptionV3 are trained over generic image datasets which doesn't serve our purpose.

Fortunately, ChexNet is a convolutional neural network especially trained on Chest X-ray images. It was trained over 1,12,120 images and contains 121 layers where the input is a chest X-ray image, and the output is the probability of 14 different diseases along with a localized heatmap which highlights the visual features of the chest x-ray image. However, we do not need to classify the image into one of those 14 categories, so we can remove the final classification layer. From a image of dimensions (224,224,3), we get a feature vector with a length of 1,024. We have two images associated with a report, so we concatenate the two feature vectors to get a feature vector with length 2,048. This final feature vector will be passed along with the report to the decoder which is discussed in the next section.

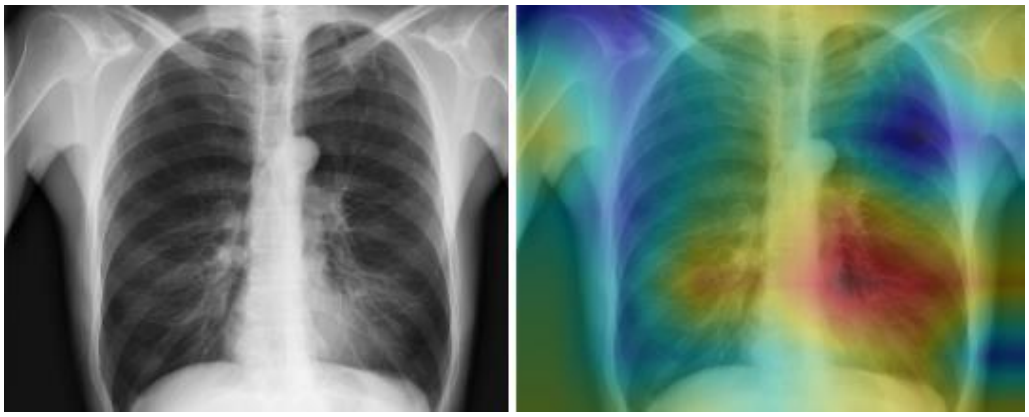


Figure 1: Chest X-ray image from the dataset before and after passing through the CNN

Hierarchical LSTMs and Attention Mechanism – Hierarchical LSTMs [10] are specialized recurrent neural networks which are often used for text generation from images and video frames. It is built to consider both high-level language features from the training text and low-level visual features obtained from the processed image. Note that the keywords/tags obtained from the image are generated by the fully connected layer, which results in a loss of spatial information. To improve these results, an additional mechanism known as Co-Attention is added. Co-Attention mechanism uses the spatial information from the visual features of the convolutional layers and the semantic features obtained from the tags of the specific image (which are generated by the fully connected layer).

When the visual features and the tags arrive at the decoder, the high-level spatial information provided by the localized heatmap help us to focus on the tags which are visually highlighted more, and yield better results. This new context vector with the embeddings of the selected tags is passed on to a sentence LSTM. A Sentence LSTM will generate multiple sentences as suggestions to the provided words, using a technique known as beam search which predicts the probability distribution across the given vocabulary and returns the words which have the maximum probability to be subsequent words in the sentence. Beam search selects multiple alternatives for an input sequence, based on conditional probability and a parameter known as beam width. When the sentence vector is successfully produced, it is passed on as a context vector to the Word LSTM. Word LSTM employs a greedy search mechanism, which selects a single candidate which is suitable for the input sequence in a time step. This improves the quality of the final sentence generated.

3.3 Proposed System Model

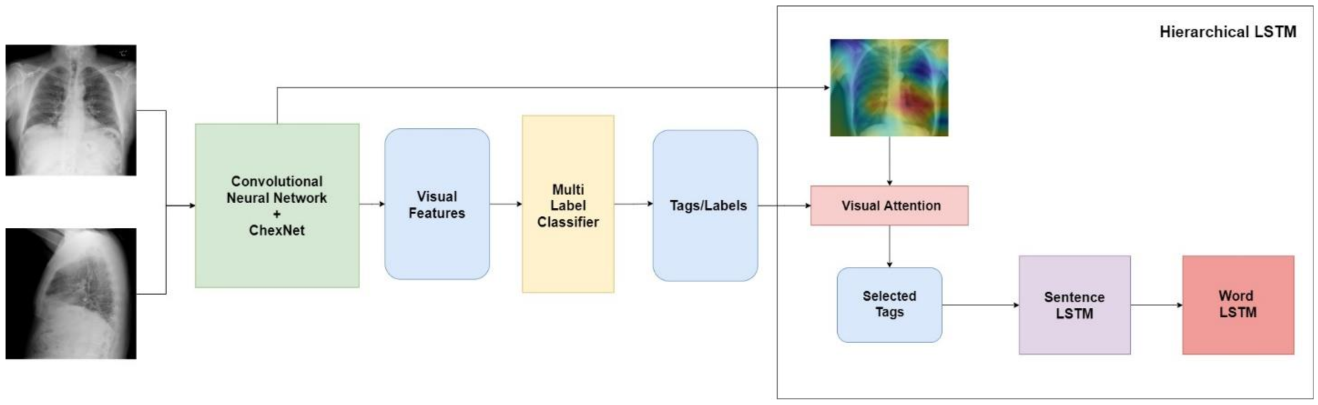


Figure 2: Proposed Architecture of the medical report generation system

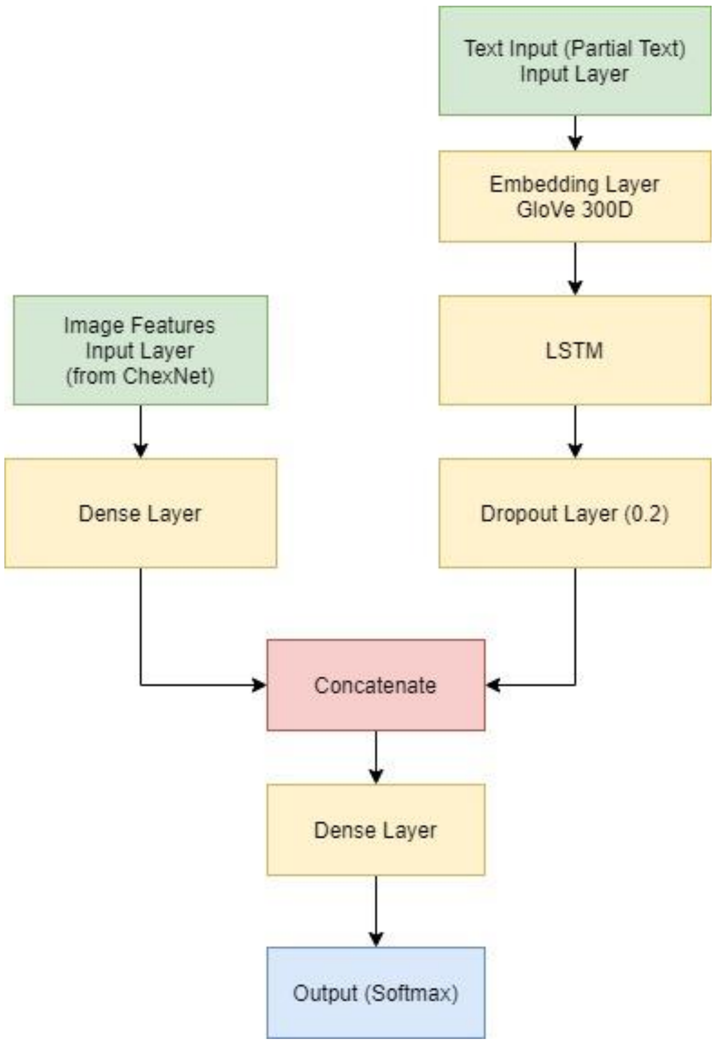
a) During the training phase, the preprocessed reports from the NLP Pipeline along with the two associated images form the inputs to the neural network. The input dataframes are obtained from pickled data using the python library Pandas.

b) The images are then augmented in order to avoid overfitting and increasing the amount of data available to the CNN. This is done using the Image Augmentation utilities which is available in the TensorFlow library. The text is converted into their embedding form using a GloVe embedding (300 dim).

c) The images are then passed through the ChexNet CNN along with a few custom CNN layers which allows us to get a visual feature context vector. The visual features of both the images are concatenated and sent to the visual attention layer. This is the encoded context vector in our sequence-to-sequence model.

d) The visual features are also sent to a multi-label classifier which predicts the tags/labels from the given images which are sent to the input layer as partial reports.

e) Once the inputs are encoded, they are passed to the hierarchical LSTMs which are specialized RNNs used for image captioning. There are multiple methods for injecting an image in an RNN as mentioned in [11], but the merge architecture suits our use-case the best. Using the merge architecture ensures that our RNN is not exposed to the visual features until the partial reports are prefixed. Due to this late binding technique, the RNN does not modify the image representation with every time step.



Text Input (Partial Text) Input Layer
Embedding Layer GloVe 300D
LSTM
Dropout Layer (0.2)
Image Features Input Layer (from ChexNet)
Dense Layer
Concatenate
Dense Layer
Output (Softmax)

Figure 3: Depicting Merge Architecture for caption generation

f) The RNN Outputs are regularized using a dropout layer, and the "merged" with the normalized image input vector, which is then passed through the dense layer. The output layer has a SoftMax function which then generates the probability distribution across the words present in the vocabulary.

g) The final report is generated using a greedy search mechanism by the sentence and the word LSTM.

4. Proposed System Analysis and Design

4.1 Introduction and Related Concepts

A medical report can usually consist of various parts, which are usually heterogenous in nature. There may be images, abbreviations and complicated terminology in these reports. To avoid this issue, we will focus on three sections from a medical report i.e., findings – a large paragraph which contains keywords and parameters, indication – the doctor's advice for the patient and impression – a sentence finalizing the results for a report. To achieve this, a multi-task architecture is proposed, which works by predicting the keywords/tags (findings) by treating it as a multi-label image classification task and generate longer descriptions by using a text generating mechanism such as hierarchical LSTMs. Hierarchical LSTMs are specialized recurrent neural networks which are often used for text generation from images and video frames. It is built to consider both high-level language features from the training text and low-level visual features obtained from the processed image.

4.2 Requirement Analysis

4.2.1 Functional Requirements

4.2.1.1 Product Perspective

A medical report can usually consist of various parts, which are usually heterogenous in nature. There may be images, abbreviations and complicated terminology in these reports. To avoid this issue, we will focus on three sections from a medical report i.e., findings – a large paragraph which contains keywords and parameters, indication – the doctor's advice for the patient and impression – a sentence finalizing the results for a report. To achieve this, this application uses a multi-task architecture which works by predicting the keywords/tags (findings) by treating it as a multi-label image classification task and generate longer descriptions by using a text generating mechanism such as hierarchical LSTMs.

4.2.1.2 Product Features

The application provides us a comprehensive medical report with three heterogenous sections i.e., Findings, Impressions and Indications from Chest X-Ray images, highlighting the context of the particular disease i.e., location, severity and affected organs, using a combination of CNN, hierarchical LSTMs and co-attention mechanism.

4.2.1.3 User Characteristics

The intended audience for this application consists mostly of medical professionals and radiologists. The users of the application should be aware of the all the different functions of the application and the proper procedure to use the application.

4.2.1.4 Assumptions & Dependencies

It is assumed that the dataset and the associated reports being used to implement the application is publicly available with all the required legal permissions. The images and reports have been thoroughly checked in order to redact any personal patient data in order to maintain their privacy. The application's accuracy partially depends on the provided data (both training and testing) as well as the data provided by the intended users.

4.2.1.5 Domain Requirements

Due to the critical nature of medical expert systems, the domain requirements mostly consist of adherence to patient-privacy laws, and patient-disclosure laws, along with regulations to check the diagnosis and the accuracy of the reports generated. Although the objective of the project is to reduce the human involvement in generating the medical reports, during the initial release of the system, it would require heavy involvement from radiologists and medical professionals in order to reduce the amount of misdiagnosis or critical mistakes.

4.2.1.6 User Requirements

The intended audience of the application would have some essential requirements, such as proper user management, high visibility and contrast on the user interface suitable for all environments and complete transparency of the data and the predictions while maintaining the privacy of the patients involved. The application would require a well-prepared documentation and tutorials for the intended users.

4.2.2 Non-Functional Requirements

4.2.2.1 Product Requirements

Efficiency

The application should be computationally efficient and the cost of working on larger datasets should be proportional. The efficiency can be greatly improved by parallelizing the training and testing of the deep learning models as they are the ones which require the most amount of computational time and resources.

Reliability

Due to the sensitive and critical nature of medical expert systems (especially involving diagnostics), it is necessary that the application is highly reliable and accurate. The complete application can be deployed as a web application on highly available servers (like AWS or GCP) to be accessed from anywhere. An alternative could also include packaging this as a standalone application for offline support which might be beneficial in rural areas where connectivity might be an issue.

Portability

Machine Learning models can be easily deployed online using micro-web frameworks such as Flask and FastAPI which are intended to be used on Python. TensorFlow along with Keras allows us to save the model weights in order to quickly model predictions on the given data. TensorFlow also allows us to convert these models in order to achieve compatibility with Android/iOS mobile applications and JavaScript. So, it’s entirely possible to deploy this application as a web application making it extremely portable, with the ability to run on multiple platforms.

Usability

With a well-prepared documentation, FAQs and tutorials, the application will be usable and convenient for every intended user. Future revisions can include multilingual support in order to improve usability among a wide variety of users. However, the multilingual support for the medical reports would require a highly accurate language translator or an accurate dataset with the reports in the language to be supported. This is due to the fact that medical reports are highly critical and sensitive in nature.

4.2.2.2 Organizational Requirements

Implementation Requirements (in terms of deployment)

- The training for the model is handled on faster GPU-enabled Colab instances (complete detail in H/W requirements).
- The saved model can be converted into various formats using TensorFlow in order to be deployed in various environments.

4.2.2.3 Operational Requirements

- **Economic –** The application should be economically viable in terms of modelling predictions and should be cheap and efficient when the application needs to be scaled.
- **Social –** The application should be socially responsible in terms of the data and the predictions generated from it. The application and the necessary data should be publicly accessible.
- **Political –** The application should be unbiased and free of any political propaganda.
- **Ethical –** The application should respect the personal privacy of the users and provide unbiased results irrespective of race or gender.
- **Health and Safety –** Due to the application being an important part in the medical system, it is important that health and safety regulations are strictly followed, which may require validation from regulatory authorities. The regulatory checks will be to check the viability and the accuracy of the predicted results due to the sensitive and critical nature of the task at hand.
- **Sustainability –** The application should be easy to update to the latest versions in order to provide the intended features without any interruption to the users of the applications.
- **Legality –** The datasets used to train the model should be open-source or publicly available along with the guarantee that the provided information in the images and reports are correct and validated by proper authorities, given that medical reports are highly critical. Any personal data in the datasets as well as the application should be redacted or encrypted as per requirements.
- **Inspectability –** Proper measures should be taken in order to facilitate any debugging in case of any errors, along with a flexible and modular design architecture which can be inspected and validated by independent as well as government authorities.

4.2.3 System Requirements

4.2.3.1 H/W Requirements

The model was trained over a Colab Pro instance, which has the following specifications. The processor is a server-grade Intel Xeon processor with support for multithreading which helps with the initial preprocessing and streaming of the data during the training phase. The GPU is one of the most essential components when training deep learning models as it allows us to distribute the massive amount of mathematical computation amongst many cores. The Nvidia Tesla P100 is the world's most advanced GPU for HPC and Deep Learning workloads. It has a 16GB of HBM2 memory along with 3584 CUDA cores. The Colab instance also has 28 GB of usable RAM and disk storage of 150GB which provides us plenty of memory in order to retain the model data, weights and various other information.

However, for using the deployed web application or to use the desktop application, the hardware requirements can be reduced i.e.

Processor	Intel Core i3 family / AMD Ryzen 3
RAM	4 GB DDR3
Hard Disk	50 GB
GPU	Integrated Intel Graphics / AMD Radeon

Table 1: Hardware Requirements

4.2.3.2 S/W Requirements

The software requirements including the programming languages, libraries/frameworks and dependencies are listed below.

Operating System	Windows, Linux, MacOS
Programming Language	Python
Data Libraries	NumPy, Pandas
Visualization Libraries	Matplotlib, Seaborn
Neural Network Libraries/Frameworks	TensorFlow, Keras

Table 2: Software Requirements

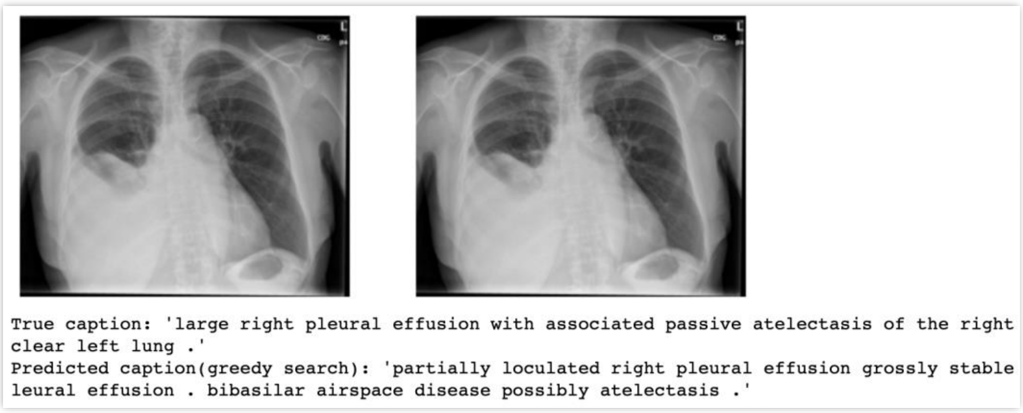
5. Results and Discussion

The automated generation of medical reports on chest X-ray images is highly viable, using a multi-step approach which employs the best techniques offered by deep learning and NLP. Chest X-rays are one of the most popular diagnostic tools and this project can improve the speed and quality of diagnosis. The CNN predicts the tags from the visual features and retains the spatial information in order to provide better context. The hierarchical LSTMs can decode the vector provided by the encoding layer in order to generate legible medical reports, which can be compared and evaluated using metrics such as BLEU Scores.

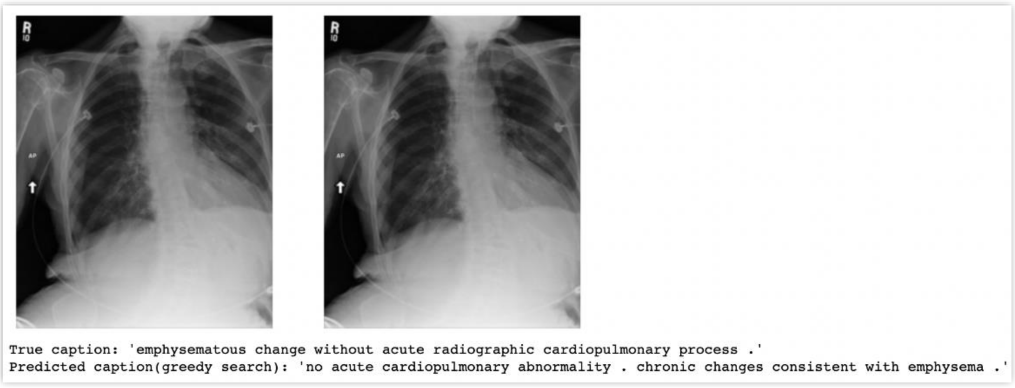
Model	**BLEU1**	**BLEU2**	**BLEU3**	**BLEU4**
Cascade RNN Model [4]	0.399	0.251	0.168	0.118
Co-attention model [7]	0.517	0.386	0.306	0.247
Simple Encoder	0.198	0.227	0.286	0.314
With Visual Attention	0.213	0.258	0.325	0.381

Table 3: Comparing Results using BLEU scores

Example Predictions with True and Predicted Captions



```
True caption: 'large right pleural effusion with associated passive atelectasis of the right
clear left lung .'
Predicted caption(greedy search): 'partially loculated right pleural effusion grossly stable
leural effusion . bibasilar airspace disease possibly atelectasis .'
```



```
True caption: 'emphysematous change without acute radiographic cardiopulmonary process .'
Predicted caption(greedy search): 'no acute cardiopulmonary abnormality . chronic changes consistent with emphysema .'
```

6. References

[1] D. Demner-Fushman *et al.*, “Preparing a collection of radiology examinations for distribution and retrieval,” *J. Am. Med. Informatics Assoc.*, vol. 23, no. 2, pp. 304–310, 2016.

[2] P. Rajpurkar *et al.*, “CheXNet: Radiologist-level pneumonia detection on chest X-rays with deep learning,” *arXiv*. 2017.

[3] A. Vaswani *et al.*, “Attention is all you need,” *Adv. Neural Inf. Process. Syst.*, vol. 30, pp. 5998–6008, 2017.

[4] H.-C. Shin, K. Roberts, L. Lu, D. Demner-Fushman, J. Yao, and R. M. Summers, “Learning to Read Chest X-Rays: Recurrent Neural Cascade Model for Automated Image Annotation,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2497–2506.

[5] Q. You, H. Jin, Z. Wang, C. Fang, and J. Luo, “Image Captioning with Semantic Attention,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 4651–4659.

[6] K. Xu *et al.*, “Show, attend and tell: neural image caption generation with visual attention,” in *Proceedings of the 32nd International Conference on International Conference on Machine Learning-Volume 37*, 2015, pp. 2048–2057.

[7] B. Jing, P. Xie, and E. Xing, “On the Automatic Generation of Medical Imaging Reports,” in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018, pp. 2577–2586.

[8] J. Pennington, R. Socher, and C. D. Manning, “GloVe: Global Vectors for Word Representation,” in *Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543.

[9] Y. Zhang, Q. Chen, Z. Yang, H. Lin, and Z. Lu, “BioWordVec, improving biomedical word embeddings with subword information and MeSH,” *Sci. Data*, 2019.

[10] L. Gao, X. Li, J. Song, and H. T. Shen, “Hierarchical LSTMs with adaptive attention for visual captioning,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 5, pp. 1112–1131, 2019.

[11] M. Tanti, A. Gatt, and K. P. Camilleri, “Where to put the image in an image caption generator,” *arXiv Prepr. arXiv1703.09137*, 2017.