# A Comparative Analysis Of The Different CNN-LSTM Model Caption Generation Of Medical Images

by

Mahzabin Yasmin Binte Amin
18101479
Weney Hasan Shammo
19101601
Jawad Bin Sayed
21341025
MD Junaied Hossain
20101204

Department of Computer Science and Engineering
Brac University
May 2023

# Declaration

It is hereby declared that:

1. The thesis submitted is my/our own original work while completing degree at Brac University.

2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.

3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.

4. We have acknowledged all main sources of help.

**Student's Full Name & Signature:**

[illegible]

Mahzabin Yasmin Binte Amin
18101479

[illegible]

Weney Hasan Shammo
19101601

[illegible]

Jawad Bin Sayed
21341025

[illegible]

MD Junaied Hossain
20101204

## Approval

The thesis titled "A Comparative Analysis Of The Different CNN-LSTM Models For Caption Generation Of Medical Images" submitted by

1. Mahzabin Yasmin Binte Amin (18101479)
2. Weney Hasan Shammo (19101601)
3. Jawad Bin Sayed (21341025)
4. MD Junaied Hossain (20101204)

Of Spring, 2023 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on May 29, 2023.

**Examining Committee:**

Supervisor:
(Member)

[illegible]

Md. Golam Rabiul Alam, PhD
Professor
Department of Computer Science and Engineering
Brac University

Co-Supervisor:
(Member)

Tanzim

Md. Tanzim Reza
Lecturer
Department of Computer Science and Engineering
Brac University

Thesis Coordinator:
(Member)

Md. Golam Rabiul Alam, PhD
Professor
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)

Sadia Hamid Kazi
Chairperson and Associate Professor
Department of Computer Science and Engineering
Brac University

# Dedication

To our loving parents, who constantly motivated us to achieve excellence in every aspect.

# Acknowledgement

Firstly, all praise be to Almighty Allah (SWT) for whom our thesis has been completed without any major interruption.

Secondly, to our thesis supervisor, Professor Dr. Md Golam Rabiul Alam Sir for his humble cooperation and guidance in our research. He has helped us enormously whenever the need arose.

Thirdly, to our Co-Supervisor, Mr. Tanzim Reza Sir for being cooperative and helping us troubleshoot throughout all the complexities that we have faced.

And finally, to our parents as without their thorough support this journey would not have been possible. With their kind support and prayers we are now on the verge of our graduation.

# Table of Contents

# List of Figures

# List of Tables

# Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

| | |
|---|---|
| $\sigma$ | Sigma |
| $BLEU$ | Bilingual Evaluation Understudy |
| $CNN$ | Convolutional Neural Network |
| $CT$ | Runs scored by Home team |
| $LSTM$ | Long Short Term Memory |
| $METEOR$ | Metric for Evaluation of Translation with Explicit Ordering |
| $MRI$ | Magnetic Resonance Imaging |
| $MT$ | Machine Translation |
| $RNN$ | Recurrent Neural Network |
| $SVM$ | Support Vector Machine |
| $VGG16$ | Visual Geometry Group with 16 layers |

# Abstract

The intent of this paper is to make the process of interpreting and understanding information within ultrasound pictures simpler and quicker by addressing the lack of techniques for automatically deciphering medical images. In order to do so, we propose a method of ultrasound image caption generation using AI that highlights the potential Machine Translation has in translating medical images to textual notations. The model needs to be trained on an ultrasound image dataset of the abdominal region including the uterus, myometrium, endometrium and cervix, a field of the medical sector that remains inadequately addressed. Two pre-trained CNN models, namely, VGG16 and Inception v3 have been used to extract features from the ultrasound images. Subsequently, the encoder-decoder model takes in two types of inputs, one for each of its layers. The two kinds of inputs are the text sequence and the image features. Both Vanilla LSTM and Bi-directional LSTM have been used to build the language generation model. The embedding layer along with the LSTM layer will process the text input. At last, the output from the two layers stated above will be merged.

**Keywords:** Ultrasound Image; Image Captioning; Medical Image Captioning; Convolutional Neural Network; LSTM

# Chapter 1

# Introduction

## 1.1 Motivation and Background

Traditionally, we see, a lot of contributions have been made to the field of medical image classification. However, the contributions made in the field of medical image caption, in the manner where a detailed diagnosis is portrayed, is very limited. Over that, diseases in ultrasound images can be very hard to identify given that the outlines of the shapes of organs are quite obscured. Moreover, it takes a large amount of time to prepare test reports for patients, increasing the waiting time required per patient to get a diagnosis. In a lot of cases it becomes an extremely tedious process to find skilled radiologists who can quickly provide an accurate diagnosis. In this paper we propose that Artificial Intelligence(AI) can save the day. Machine Translation (MT) is already being used in a wide range of general purposes. Within the field of MT, recurrent neural networks are used in the encoder-decoder architecture to solve sequence-to-sequence prediction issues. In an encoder-decoder model the encoder is a neural network that transforms input data into a different representation and extracts its important features. The encoder takes an input, for example, image or text and compresses it into a lower dimensional representation. After that, the decoder takes input from the output of the encoding layer and maps the compressed representation into its original format. With substantial changes made to it, the encoder-decoder architecture can bear great potential in translating medical images to captions. However, that it is trained on diagnosis based on medical images through the usage of Neural Network (NN) models such convoluted Neural Network (CNN) and Long Short Term Memory (LSTM) needs to be ensured.

## 1.2 Problem Statement

There exists a lack of techniques for automatically deciphering information from ultrasound images despite ultrasound imaging being an invaluable tool for studying complicated anatomical structures. This can be attributed to diseases in ultrasound images being very hard to identify given that the outlines of the shapes of organs are quite obscured. Moreover, it takes a large amount of time to prepare test reports for patients, increasing the waiting time required per patient to get a diagnosis.Most research found in the status quo in regards to image understanding is mostly concerned with natural images. Ultrasound image captioning is particularly more challenging than natural images since a description must capture not just the features within an image but also their relationships to one another and their characteristics.

An even more overlooked area of the already challenging field of research is female healthcare. A significant research deficit has long existed in the area of female healthcare, which has impeded our knowledge and development in this important area. Female-specific health issues and diseases have been left unaddressed or little understood. Wide-ranging effects of this disparity include misdiagnoses and inappropriate treatments for women. Focused study is necessary to address the specific biological and hormonal difficulties that women encounter throughout their lifetimes, such as reproductive health and menopause.

In this paper, we want to emphasize on the hybridization of the pre-trained models of CNN and LSTM in order to improve the current situation of caption or report generation in the medical imaging field.

## 1.3 Research Objective

We wish to present a method of ultrasound image caption generation through the usage of AI and develop the functionalities of caption generation such as time efficiency and the accuracy of existing caption generation models. In order to ensure equal improvements in medical knowledge and improved health outcomes for all women, it is critical that we rectify this historical neglect and actively engage in comprehensive and extensive research projects that target female healthcare. The paper will aim to focus on how CNN and LSTM, in tandem, will perform to generate medical image captions based on ultrasound images. Along with that, we want to showcase a comparative analysis of the certain variants of LSTM such as Vanila LSTM and Bi-directional LSTM.

The following is a list of the contributions of this research:

- We propose an encoder-decoder model where pre-trained CNN models have been initialised as the encoder while RNN, specifically variants of LSTM, has been used at the decoder end. The model is to process both image and text sequences in order to produce authentic annotations for ultrasound images.
- We provide a comparative analysis of the four different variants of the models used in the context of this research with the aid of relevant performance metrics. The combinations are Inception v3-vanilla LSTM, Inception v3-Bi-directional LSTM, VGG16-Vanilla LSTM and VGG16-Bi-directional LSTM.
- We extensively train our models on our own curated dataset consisting of ultrasound images and their corresponding reports providing diverse diagnoses pertaining to female health. This demonstrates the learning performance of our proposed architecture.

- We highlight the effectiveness of architectures such as CNN and LSTM in translating medical images and attaining salient information for given image input. This, in turn, can help patients and doctors understand obscure ultrasound images more clearly and make the whole process less time consuming.

## 1.4 Thesis Outline

The research report is organized as follows:

**Chapter 1** consists of the motivation behind our research, objectives of our research and problem statement.
**Chapter 2** details relevant theoretical concepts including medical caption generation, encoder-decoder model, CNN, and LSTM, and also includes previous works related to this research.
**Chapter 3** is the methodology section that is comprised of the workflow, description of the data and the model.
**Chapter 4** includes the result analysis and showcasing comparison of the performances of the implemented techniques.
**Chapter 5** concludes the report.

# Chapter 2

# Literature Review

## 2.1 Medical Image Caption Generation

A research area that has garnered rapid recognition in recent years is image captioning. It refers to the process of automatically generating captions from an image, a goal that is attained by capturing semantic information from images and thereafter expressing the extracted information in natural languages. This field necessitates the bridging of two major research fields, namely, computer vision and natural language processing.

A promising, even though challenging, sub-field of image captioning is medical image captioning. Medical caption generation in general, makes use of machine learning models, deep learning models in particular, to generate textual summaries or annotations of both medical images and videos [12]. Analyzing medical images and videos from various modalities ranging from X-ray, MRI, Ultrasound, CT, and more, effective advancements in this field has the potential to bring about substantial improvements in healthcare by providing better results and further down the line, at a lower cost[11]. Medical caption generation can play an integral role in improving the interpretability of medical images and assisting healthcare professionals in diagnosis, treatment, planning, and communication. It can contribute to improved patient care, increased efficiency, and enhanced medical education and research.

The non-negotiable requirement for acute precision and accuracy in the results obtained, makes it imperative that a sizable dataset is used to train the relevant deep learning models. In fact, the availability of large medical datasets is few and far between and that is exactly where the greatest challenge of medical caption generation resides. In order to extract meaningful features from the visual content to formulate accurate descriptive captions, techniques like convolutional neural networks (CNNs) for image analysis and recurrent neural networks (RNNs) or transformers for caption generation are used.

Metrics such as BLEU (Bilingual Evaluation Understudy) and METEOR (Metric for Evaluation of Translation with Explicit Ordering) are commonly used to evaluate the overall performance of the system by assessing the similarity between the generated captions and human-generated reference captions.

## 2.2 Encoder-Decoder Model

Encoder Decoder models are very popular for machine translation tasks[18]. In an encoder decoder model the encoder is a neural network that transforms input data into a different representation and extracts its important features. Most of the time the encoder converts the feature vectors into lower dimensional representations. The encoder takes an input that could either be image or textual data, and compresses it into a lower dimensional representation; this compressed representation contains the most important features of the input. After that, the decoder takes input from the output of the encoding layer. The decoder then maps the compressed representation into its original format. Encoder-Decoder models are used in many types of applications such as speech recognition, language translation, and data compression.

## 2.3 Convolutional Neural Network (CNN)

Convolutional Neural Networks(CNN) are a deep neural network which are mostly used for image processing [9]. These are hugely used for tasks like image classification and object recognition in images. In CNN, at the very onset, we take an input image and it is processed by the convolution layers. The convolution layers generate feature maps using filters that go through the pixels of the image. Each filter learns some patterns in the pixels of the images, patterns such as edges, corners and textures. The feature maps are activated to add non-linearity and enhance the model's capacity using activation functions to learn intricate patterns. Then pooling layers are used to downsize the feature maps and reduce spatial dimensionality. Afterwards, the output from convolution layers and pooling layers are passed to the hidden fully connected layers. These fully connected layers are then connected to an output layer that eventually predicts the final prediction. A loss function is used to calculate the weights and biases using the predicted outputs and ground truth.

This paper [11] shows the use of Convolutional Neural Networks (CNNs) for adaptive image processing. The discussion emphasizes CNN's capacity for a wide range of applications other than image processing, such as physiological records, financial time series analysis, and satellite image processing. It advises investigating CNN architectural enhancements such as the use of detachable filters and expansions to three-dimensional (video) processing. The paper also requests theoretical work that connects CNNs with finite impulse response filters, adaptive filters, and wavelet transformations. It also advises looking at approaches for customizing the CNN architecture using discriminant or entropy-based cost functions.

The research proves the applicability of CNNs for adaptive image processing while also identifying opportunities for future research and development in CNN structures and applications.

Author Andrew Gibiansky [17] explains convolutional neural networks (CNNs) and its architectural components, such as convolutional and max-pooling layers. The formula for determining the input to a unit in a convolutional layer in forward propagation is written as:

$$x_{\ell ij} = \sum_{a=0}^{m-1} \sum_{b=0}^{m-1} \omega_{ab} y_{\ell-1}(i+a)(j+b) \tag{2.1}$$

where $\omega$ is the filter, $y_{\ell-1}$ represents the previous layer, and $x_{\ell ij}$ is the pre-nonlinearity input.

The nonlinearity is applied using the formula :

$$y_{\ell ij} = \sigma(x_{\ell ij}) \tag{2.2}$$

where $\sigma$ denotes the activation function. Max-pooling layers Select the highest value inside each block to reduce dimensionality.

The gradient of the filter weights is determined using the formula for backward propagation:

$$\frac{\partial E}{\partial \omega_{ab}} = \sum_{i=0}^{N-m} \sum_{j=0}^{N-m} \frac{\partial E}{\partial x_{\ell ij}} y_{\ell-1}(i+a)(j+b) \tag{2.3}$$

The deltas $\left(\frac{\partial E}{\partial x_{\ell ij}}\right)$ at the current layer are computed as:

$$\frac{\partial E}{\partial x_{\ell ij}} = \frac{\partial E}{\partial y_{\ell ij}} \sigma'(x_{\ell ij}) \tag{2.4}$$

where $\sigma'$ represents the derivative of the activation function.

The errors $\left(\frac{\partial E}{\partial y_{\ell-1ij}}\right)$ at the previous layer are obtained through a convolution with the flipped filter weights:

$$\frac{\partial E}{\partial y_{\ell-1ij}} = \sum_{a=0}^{m-1} \sum_{b=0}^{m-1} \frac{\partial E}{\partial x_{\ell(i-a)(j-b)}} \omega_{ab} \tag{2.5}$$

CNNs use weight sharing to replicate the behavior of the human visual cortex, and they have shown to be extremely effective in applications like object identification.

The next paper [7] implemented multiple convolutional neural network models to create a system for identifying fruit flies using convolutional neural networks, deep learning architectures, and transfer learning. The researchers conducted two tests to assess the efficacy of various deep features and machine learning algorithms. They examined five in-depth features and nine machine-learning approaches (Inception, ResNet, VGG16, VGG19, and Xception). The results demonstrated that bottleneck characteristics retrieved using the VGG16 architecture outperformed support vector machines (SVM) using a linear kernel. The research also compared the top-performing tuples (in-depth feature + learning technique) to two state-of-the-art techniques from the literature. Statistical analysis found no significant difference between the deep learning algorithms and the two baselines despite the fact that VGG16+SVM attained the greatest mean accuracy of 95.68%. However, one of the baselines depended on color-based histogram characteristics, which may not be appropriate for real-time systems. The advantages of deep learning techniques, such as their ability to extract a mixture of visual features from data were underlined by the authors. Deep learning approaches, as opposed to the mid-level representation approach, eliminate the necessity of picking the appropriate mix of keypoint detectors and feature extractors. Without any extra image processing advancements, the suggested approach outperformed the baselines. Finally, the research showed the efficacy of deep learning architectures and transfer learning for fruit fly identification. The VGG16 architecture with SVM and a linear kernel produced the greatest results. This method has the potential to contribute to the development of real-time pest management systems in agriculture. Future work might include ensemble classifiers and the creation of a transportable device to help field specialists.

## 2.4 Long Short-Term Memory (LSTM)

According to the article[16], the neural networking algorithm, LSTM, is used to resolve the problem of exploding and vanishing gradients. The LSTM network is composed of one or multiple memory modules or cells, and each of them consists of units of memory that govern the flow of data throughout the system. LSTM takes on a function that is responsible for the memorization of time-series related data unlike RNNs, as these cells can consist of three different logic gates known as input, output and forget gate due to the sigmoid neural network layer. This layer is responsible for selectively passing information. In order to construct a new value, the input gate begins with summarising the cell unit status value, filtered value, and added value. The forget gate produces an output with a range of 0 to 1, which represents a value that can be disregarded or reserved in the system, respectively. The system employs a storage gate which in addition contains sigmoid and tanh layers. To choose new data to store in the cell, this gate is used. The sigmoid layer picks the value that has to be changed in this case, and the tanh layer's job is to create vectors of new candidate values before adding them to the cell unit state. The LSTM network architecture is described as a sequential model with two fundamental components: states and gates. The previously concealed layer's value is the hidden state, whereas the input state is a linear mixture of the present input data and the state that was previously concealed. Each unit of the LSTM cell network, which comprises three gates, uses an optimizer function to modify the weights connected with the network's units.

The value, "f(t)" is required to find which information from the previous state needs to be retained for further calculation. It is computed at the forget state using the following equation:

$$f(t) = \sigma(W(fx) \cdot x(t) + W(fs) \cdot s(t-1) + b(f)) \tag{2.6}$$

where, "$\sigma$" is the sigmoid activation function.

Subsequently, the intermediate parameters i(t) and C(t) are found using the input gate. This is done to figure out if the internal state values function as memory cells or not. The equations below are used for this purpose:

$$i(t) = \sigma(W(ix) \cdot x(t) + W(is) \cdot s(t-1) + b(i)) \tag{2.7}$$

$$c(t) = \tanh(W(cx) \cdot x(t) + W(cs) \cdot s(t-1) + b(c)) \tag{2.8}$$

Lastly, the information to be retained is derived by merging the outputs of the input and forget gates:

$$C(t) = f(t) \cdot C(t-1) + i(t) \cdot c(t) \tag{2.9}$$

The sigmoid and tanh layers are applied to calculate fresh data to be stored in cell state. The output layer then uses the o(t) equation to create an output that is used to anticipate the final output, s(t).

$$o(t) = \sigma(W(ox) \cdot x(t) + W(os) \cdot s(t-1) + b(o)) \tag{2.10}$$

$$\tilde{s}(t) = o(t) \cdot \tanh(C(t)) \tag{2.11}$$

The final functions are "W" and "b," which stand for the respective weights and biases applied at various layers, and "s(t)," which stands for the output of the LSTM network at time signal t.

The LSTM networks are classified as vanilla, stacked, bidirectional, and other flavours[16]. Vanilla LSTM has one LSTM unit in the hidden layer and one in the output layer. On the other hand, stacked LSTM architecture consists of numerous stacked LSTM layers that are transmitted to a dropout layer and output layer at the final output. Additionally, by using the information in both ways, bidirectional LSTM improves model performance for sequence classification challenges as the bidirectional LSTM has an additional LSTM layer that changes the direction of information flow.
The bidirectional mechanism research by J. Shah, R. Jain, V. Jolly, and A. Godbole demonstrates that the sigmoid layer specifies what information must be conserved

and erased from the bidirectional LSTM cell which holds for the more traditional and simple versions of LSTM as well.[14].

## 2.5 Related Works

According to Baltrusaitis et al.[6], the research based definition of multimodality lies within the different modalities of human perception, such as vision and auditory senses. The paper then goes into giving details of five adversities to the development of multimodal machine learning, which are representation, translation, alignment, fusion and co-learning. Data representation deals with formatting unprocessed data for any model to use and is essential for machine learning models. Unimodal data can be portrayed as multimodal data via joint representation and are used in neural networking, sequential representation and probabilistic graphical models, whereas coordinated representations are substitutes for it and are primarily used in similarity models. A huge chunk of multimodal learning is dependent on mapping or translation. Its job is to convert from one modality to another. Example-based translation can be retrieval-based where the translation is extracted with no changes, and combination based where translations are obtained from a complicated set of rules. Meanwhile, generative approaches to translation are concerned with grammar-based models, which make the task more comprehensible by limiting the target domain using a set of rules. Encoder-decoder models encode the input modality to a form of representation and then decode to produce output modality and continuous generation models which produce the output modality while input modality streams are constantly fed into the system. The alignment of multimodality is concerned with developing connections between sub-parts of multiple modalities and is of two types, implicit and explicit. Implicit modalities use graphical models or neural networks whereas explicit modalities work with supervised and unsupervised alignment techniques. The paper then talks about the process of fusion, that is, developing information from multimodal data for forecasting a result. Model agnostic fusion is not based on predefined ML architecture and can be divided into early, late and hybrid fusion. Model-based fusion prioritizes the concept of fusion and can be utilized in multiple kernel learning, and graphical models like RNN and LSTM. The last adversity in the hierarchy of multimodality is co-learning which helps develop a modality that lacks amenities by utilizing pre-learnt information from a different modality with sufficient amenities. Therefore it deals with the challenges of handling parallel and non-parallel data along with co-training and transfer learning.

Zhang et al.[10], discusses the representation of multimodalities, and its fusion and applications. The paper details the different types of representations of unimodal intelligence such as linguistic, visual, word vector, image vector, speaker, etc. The paper also discusses the representation of multimodal intelligence such as unsupervised training techniques. Examples include joint embeddings and supervised learning with either multimodal biased components or intra-modality producing components. Other representations of multimodal intelligence can be done through zero-shot learning techniques and techniques that use transformer models.

Shah et al.[13], present the challenges imposed by the typical encoder-decoder method of image captioning. Conventionally, CNN is used as an encoder whereas RNN, a kind of sequence generator, is employed as the decoder. While this approach has been proven to be efficient in most cases, the need for sequences to be processed in order can be of disadvantage. To address the inability to parallel the generation of captions, considerable changes have been made to image captioning in the English language in recent times. Most notable is the use of a Transformer model. The paper essentially attempts to replicate the efficiency derived from the Transformer model in respect of Bengali language using three Bengali datasets including Flicker8k, BanglaLekha and their own curated dataset, Bornon.

Park et al [12] presents how medical image caption generation is a promising but challenging sub-field of image captioning. Medical caption generation in general, makes use of machine learning models, deep learning models in particular, to generate textual summaries or annotations of both medical images and videos. Analyzing medical images and videos from various modalities ranging from X-ray, MRI, Ultrasound, CT, and more, effective advancements in this field has the potential to bring about substantial improvements in healthcare by providing better results at a lower cost. Medical caption generation can play an integral role in improving the interpretability of medical images and assisting healthcare professionals in diagnosis, treatment, planning, and communication. It can contribute to improved patient care, increased efficiency, and enhanced medical education and research. The non-negotiable requirement for acute precision and accuracy in the results obtained, makes it imperative that a sizable dataset is used to train the relevant deep learning models. In fact, the availability of large medical datasets is few and far between and this is where the greatest challenge of medical caption generation resides. In order to extract meaningful features from the visual content to formulate accurate descriptive captions, techniques like convolutional neural networks (CNNs) for image analysis and recurrent neural networks (RNNs) or transformers for caption generation are used.

Zeng et al. [8], discusses the challenges associated with understanding ultrasound images and proposes a coarse-to-fine medical image captioning ensemble model for generating descriptive captions for ultrasound images. They initialized their model for extensive training of the ultrasound pictures using transfer learning, which transfers parameters from the pre-trained VGG16 model. They have suggested a strategy to train the ultrasonic image encoding model from coarse to fine in order to lessen the mutual interference between various illnesses of various organs. The ultrasound pictures used in this study were taken between 2014 and 2015 at a tertiary hospital in China. The approach highlights the necessity of studying both the natural language necessary for sentence formation as well as the visual comprehension of ultrasound pictures. The authors suggest integrating image captioning generation, a method used to comprehend content information in natural images, into the field of ultrasonic image understanding in order to address the lack of techniques for automatically deciphering information pertaining to diseases within ultrasound pictures. With the use of this technique, ultrasound pictures may be converted directly into annotation text, providing a thorough grasp of the information contained within. The ultrasound picture captioning generating method provides doctors and patients with a more practical and effective way to immediately understand ultrasound images. This concept aims to improve the efficiency and practicality of content interpretation and understanding inside ultrasound pictures. The model has three primary steps for operation. As the initial step to identify the organs in the ultrasound images, a coarse classification model is used. Second, based on the determined organ labels, a fine-grained classification model encodes the ultrasound pictures. As a final step, the encoded vectors are sent into a language generation model, which creates annotation text for the ultrasound images that automatically describes the diseases they contain. The ultrasound image's encoding vector is expected to include as much information about illnesses as feasible. CNN is used as the ultrasonic image encoder throughout the encoding step. To initialize the encoder model, they employed the pre-trained CNN model that was trained on the ImageNet dataset. The LSTM model is an excellent answer to the issue of long-term sequence dependency because it features a multiplicative gate structure that effectively handles gradient explosion and vanishing. Hence, they have chosen to employ LSTM to build the language generation model because of the considerable advancements achieved by the LSTM model in the existing techniques for image captioning. The liver, gallbladder, and kidney are the three categories into which they have categorized the ultrasound pictures after the first phase of training using a coarse classification model they named "organ." The matching language generation model with organ labels receives the encoding vector as input. The annotation text, which is composed of n-grams and is utilized by BeamSearch to describe the content of the ultrasound picture, is then obtained. Four assessment metrics—BLEU, ROUGE-L, METEOR, and CIDEr—have been utilized to assess the performance of the ensemble model for ultrasound picture captioning.

# Chapter 3

# Methodology

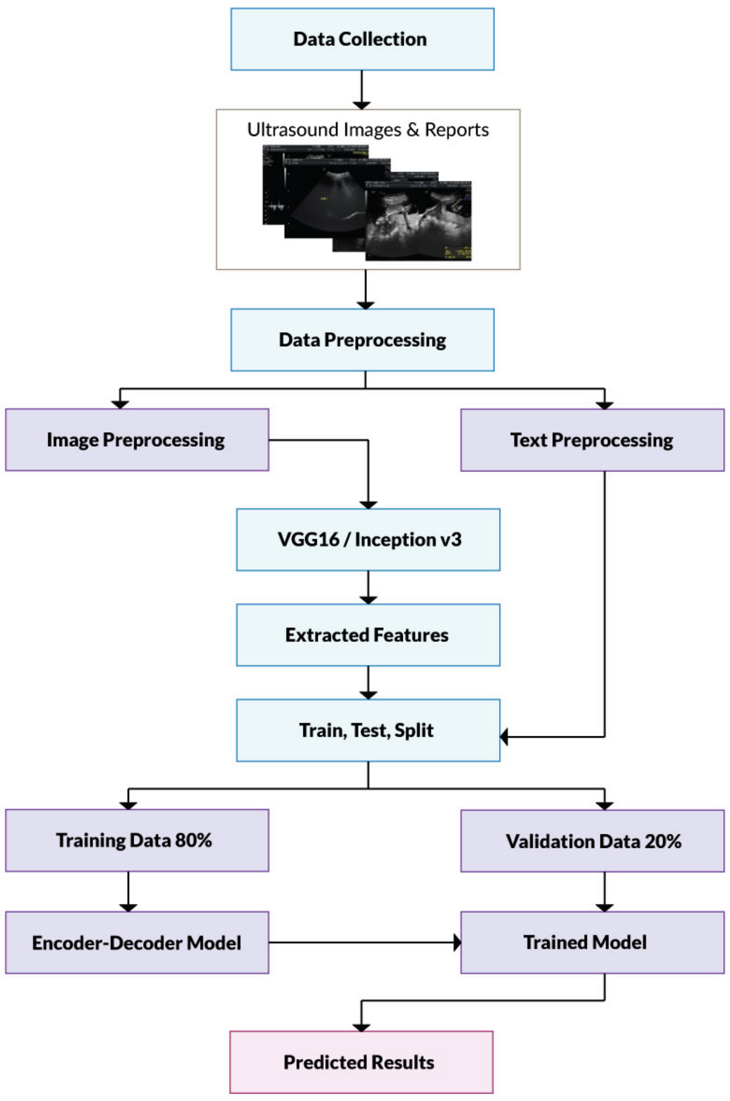


Figure 3.1: Top level overview of the proposed method.

We began by collecting the data, consisting of the ultrasound images and their corresponding reports. The collected data was filtered to suit our needs, that is, the scans for the required organs were selected and kept. The reports, too, were filtered to gain the necessary portions of texts. Thus, going forward, the images were prepared and visualized for usage. Afterwards, the text pre-processing was done. Patient's personal information was removed from both the respective scans and reports to maintain confidentiality. Following the pre-processing of data, it was split into the train and test datasets. Each set was pre-processed using relevant pre-processing techniques. From here, we move on to the feature extraction step that utilizes pre-trained CNN algorithms and the structure's ability to automatically learn characteristics at different levels from data. Baseline CNN-LSTM models were built following that. We used our models to compare the predicted and actual captions. Based on the performance of the models and the percentage accuracy derived from the results, the best model was determined.

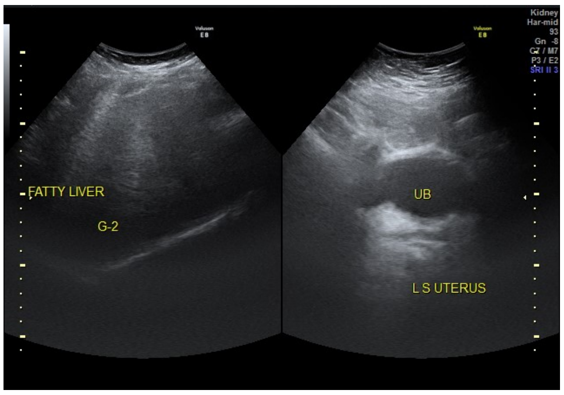
## 3.1 Description of the Data

### 3.1.1 Data Acquisition

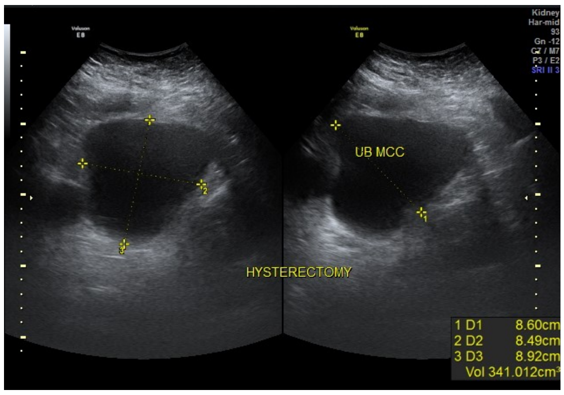
We have constructed an ultrasound image dataset including the uterus, myometrium, endometrium and cervix. The dataset includes images for an array of women of different ages and in different cycles of their lives ranging from pregnant women, elderly women and women who have surgically removed their uterus. Needless to say, there are distinctions within those categories as well.

Our ultrasound images are from an esteemed hospital in Dhaka, Bangladesh, and have been collected over the span of a month, that is, from late April 2023 to May 2023. The shape of the same organ may alter in different ultrasound pictures because doctors examine patients' bodies from a variety of angles and the ultrasound intensity is not consistent. Ultrasound images from a particular radiologist or sonologist can belong to patients of both sexes, and also be of different regions of the body. Therefore, not all ultrasound images can be used to construct the available dataset. From the data collected in the given period, reports and images belonging to male patients were discarded. Afterwards, reports containing no comments on the uterine region, such as reports on the breast tissues, are not used in our dataset either.

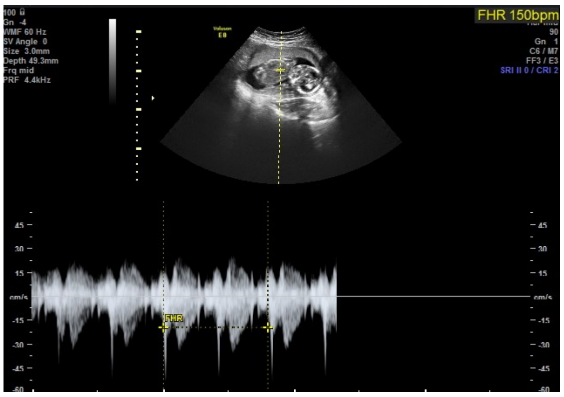
The ultrasound images are partitioned according to the keywords "uterus", "myometrium", "endometrium" and "cervix" in the text extracted from the original diagnostic reports with the help of a seasoned radiologist or operator. The ultrasound images and the corresponding text data are both saved after all personal patient information from the images and reports are removed to maintain confidentiality.



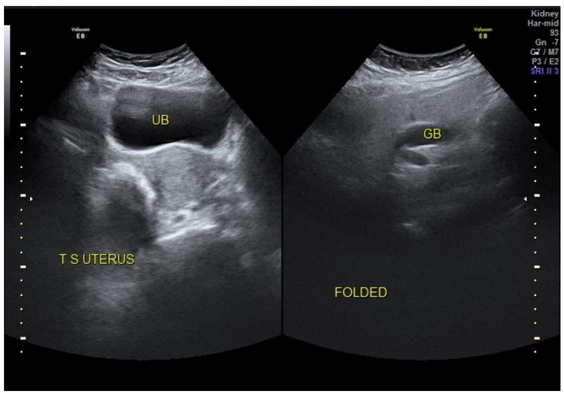
Uterus is atrophied



Uterus is surgically removed



Uterus is gravid containing single foetus



The uterus is bulky in size

Figure 3.2: Partial samples of dataset

### 3.1.2 Text Data Preparation

Each of the captions was placed after their respective image file names in a single text file. The caption was extracted per image and redundancies such as punctuation, spaces, newlines were removed. All letters were made lower case to maintain consistency as well. The text was also put in order as such, similar information was placed in the same order to further maintain this consistency. Each of the captions was tokenized, that is, divided into words. Each word was assigned a unique index number. Each unique word is considered a new vocabulary. The total amount of vocabulary was then calculated. Maximum length of the sequences is calculated. Smaller lengths of sequences were padded in comparison to the maximum length so that all of the sequences are padded to the maximum length.

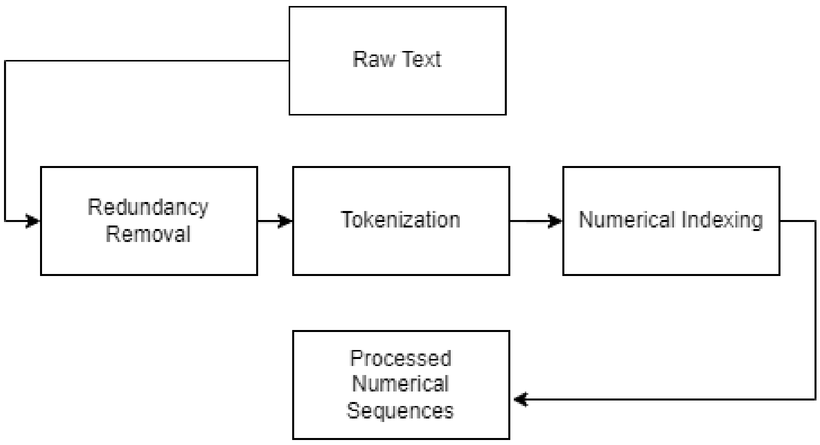


Figure 3.3: Steps for text preprocessing.

### 3.1.3 Image Data Preparation

To process the images, at first, the color space was changed. For the Inception v3 model, RGB is required therefore, if any of the images contain VGG color space it is changed. For the VGG16 model, VGR is required, therefore if any RGB image exists it is converted to VGR. The images are then resized. For VGG16 all the images are resized to 224 x 224 resolution whereas for Inception v3, all the images are resized to a resolution of 299 x 299. Finally, the pixel values have been modified and were normalized as such that the value ranges from -1 to +1.

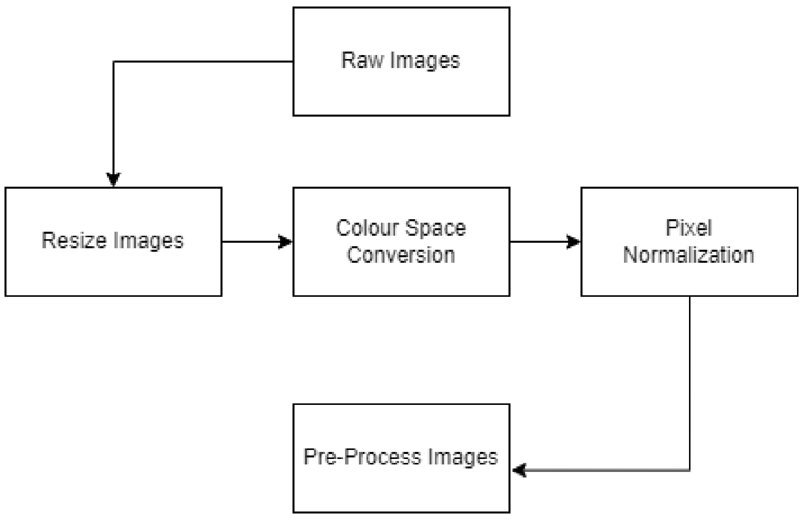


Figure 3.4: Steps for image preprocessing.

## 3.2 Model Specification

### 3.2.1 Inception v3

The input size for Inception v3, typically, is 299x299 pixels [5]. The first component of the architecture is the stem block where input image is processed, and low-level features are extracted. Convolutional, max pooling, and dimensionality reduction layers are frequently used here in conjunction. The stem block helps to recognize fundamental visual patterns and gets the data ready for further processing.

Inception-A is a building block of the architecture where features are captured and processed at multiple scales. Parallel pathways with various convolutional procedures, including 1x1 convolutions, 3x3 convolutions, and double 3x3 convolutions, are included in this. These routes enable a variety of feature representations and information acquisition at various receptive fields.

The Reduction-A block is reached after a series of Inception-A blocks. Its goal is to increase the number of channels while downsampling the feature maps' spatial dimensions. It frequently combines max pooling with convolutional layers with greater strides. This block aids in introducing some spatial compression. It does so by acquiring higher-level features and lowering computational complexity.

The Inception-B captures features at multiple scales, much like the Inception-A block. Parallel routes with various convolutions, such as 1x1 convolutions, 7x7 convolutions, and double 7x7 convolutions, are included. A string of Inception-B blocks is followed by the Reduction-B block. The spatial dimensions of the feature maps are further reduced. Downsampling the feature maps using the reduction blocks allows for the capture of higher-level, more abstract features.

The Inception-C block continues to utilize parallel pathways with various convolutions, including 1x1, 3x3, and double 3x3 convolutions to obtain multi-scale features. The block makes it possible to integrate the features at multiple scales by improving the expressive power of the network.

Auxiliary classifiers are extra branches that link to intermediate layers. These are included to help with the training process' vanishing gradient issue. They add more gradients and loss functions, facilitating learning and enhancing gradient flow.

The average pooling layer comes after the Inception-C block. The network can handle input pictures of any size due to this layer's reduction of the spatial dimensions of the feature maps to a fixed size. By averaging each spatial region, the features are summarized. Figure 3.5 shows the detailed architecture of the model.

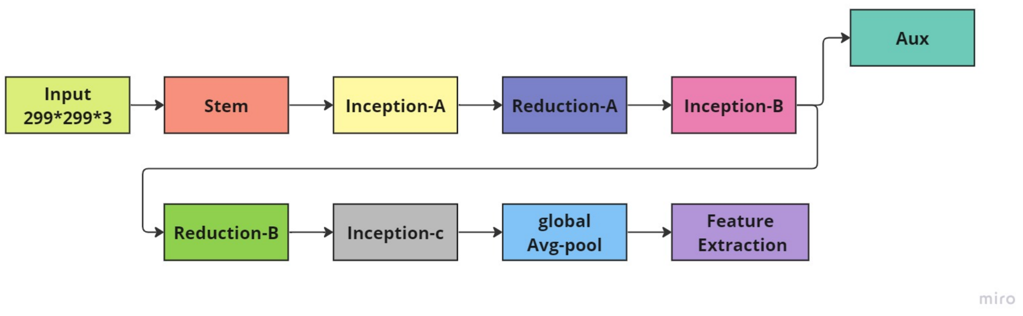


Figure 3.5: Model architecture of Inception v3.

### 3.2.2 VGG16

The VGG16 network receives a fixed-size RGB image as its input, generally measuring 224x224 pixels. There are 16 layers altogether in the VGG16 architecture, comprising 13 convolutional layers and 3 fully linked layers [4]. Small (3x3) receptive fields are used in the convolutional layers to better capture local features as well as patterns. From 64 to 512 filters are added progressively. This enables the network to learn increasingly complex and abstract representations as it progresses.

The max pooling layers with a 2x2 window downsample the feature maps while the convolutional layers collect features from the input picture. The pooling layers downscale the feature maps' spatial dimensions over time, collecting just the most important characteristics. Each of the three fully connected layers following the convolution and pooling layers, consists of 4096 neurons. Using the softmax activation function, The final layer of VGG16, the fully connected layers at the network's end categorize the features into one of the 1000 distinct classes in the ImageNet dataset.

A ReLU activation function follows each convolutional layer besides the last layer. The network can learn complicated representations thanks to the ReLU activation, which aids in the creating non-linearity. The application of dropout regularization after every fully connected layer is done to avoid overfitting.

The sequential structure of VGG16 is useful in allowing the network to progressively collect features, both local and global, from the input images. This leads to a representation that is discriminative and rich and hence, results in a more accurate image classification. Figure 3.6 shows the detailed architecture of the model.

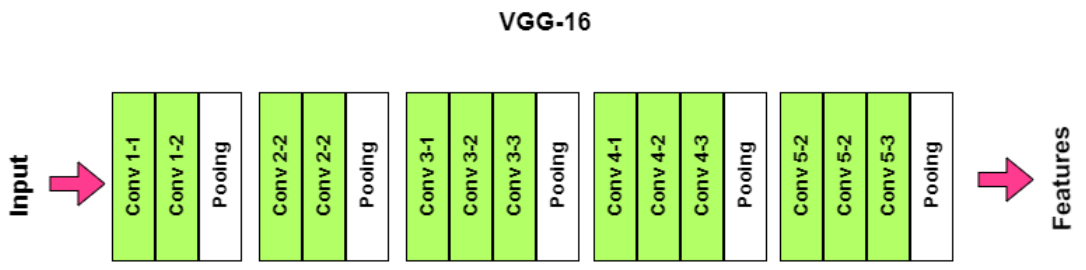


Figure 3.6: Model architecture of VGG16.

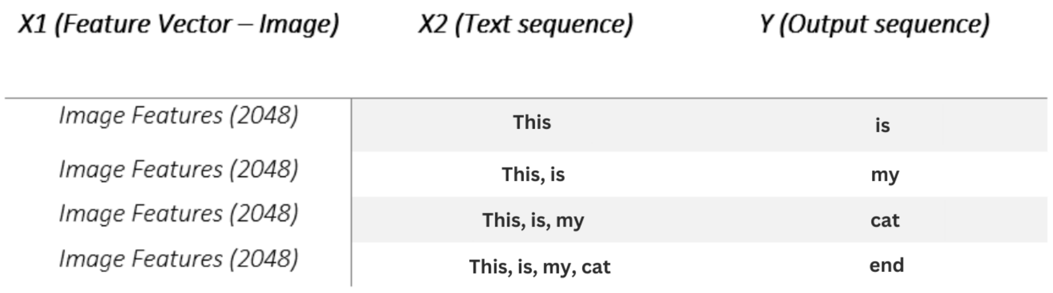
### 3.2.3 Feature Extraction

Two pre-trained CNN models, that is, the VGG16 and Inception v3 were used to extract the features from the ultrasound images. Pretrained models are advantageous as they do not require to be trained from scratch with large amounts of data. The models were trained using the ImageNet dataset. The pretrained weights were loaded for both the models. For both these models the final layer, that is, the dense layer is removed. The immediately preceding layer of the dense layer, that is, the global pooling layer, is the layer from which the features were extracted. The output that was generated from this layer is the actual feature vector.

### 3.2.4 Input Data Preparation

After extracting the required features using both the VGG16 and the Inception v3 model, it was time for preparing the input data for the actual caption generation model which is the encoder-decoder model consisting of two layers. The layers take in two types of inputs, that is, the text sequence and the image features.

The caption generation model works in the manner such that it takes one image feature and a specific text sequence depending on the specific word and therefore, predicts and then generates the immediate next word as shown in Figure 3.7.



| *X1 (Feature Vector – Image)* | *X2 (Text sequence)* | *Y (Output sequence)* |
|---|---|---|
| *Image Features (2048)* | **This** | **is** |
| *Image Features (2048)* | **This, is** | **my** |
| *Image Features (2048)* | **This, is, my** | **cat** |
| *Image Features (2048)* | **This, is, my, cat** | **end** |

Figure 3.7: Sample input and output data.

### 3.2.5 Vanilla LSTM

The basic recurrent neural network (RNN) model is extended with the vanilla LSTM (Long Short-Term Memory) architecture in order to overcome the vanishing gradient problem [1]. This also captures the long-term relationships in sequential data. It adds new gating systems that control how information moves across the network. The input gate, forget gate, and output gate are the three primary parts of a standard LSTM. At various levels of processing, each of these gates is in charge of managing the flow of information.

The input gate chooses which components from the prior hidden state and the current input should be included in the current memory cell. It applies a sigmoid activation function on the hidden state and the current input. The numbers between 0 and 1 produced by this activation function represent the relative importance of each component. A potential activation vector is element-wise multiplied by the outcome of the sigmoid operation. Information from the current input that is considered to be pertinent is included in this vector. The value derived from the multiplication of the sigmoid result and the candidate activation vector is equivalent to the output of the input gate. This output stands for the new information meant to be added to the memory cell.

Which components of the preceding memory cell should be destroyed or forgotten is decided by the forget gate. It uses a sigmoid activation function to transmit the current input and the prior hidden state through, much like an input gate. The prior memory cell state is multiplied element-wise by the sigmoid output. This procedure manages the quantity of information from the preceding memory cell that is to be preserved. The components of the preceding memory cell that will be retained are represented by the resultant vector.

The input gate and forget gate outputs are combined in the memory cell update step to update the memory cell's current state. The output of the forget gate is elementally multiplied with the preceding memory cell state. The network can now get rid of unnecessary data from the preceding memory cell thanks to this procedure. The output of the input gate, which represents the new information to be assimilated, is added element by element as a result. The resultant vector shows the altered state of the memory cell.

The output gate chooses the hidden state that will be transferred to the following time step and utilized to make predictions. A sigmoid activation function is used to process the current input and the prior hidden state. A hyperbolic tangent (tanh) activation function, which condenses the values between -1 and 1, is applied to the state of the present memory cell to enable non-linear changes. The sigmoid output and the tanh output are multiplied element-by-element as this gate's output. This procedure regulates the information transfer from the memory cell to the hidden state. The hidden state is represented by the resultant vector. The vanilla LSTM design enables the network to selectively retain or forget information over lengthy sequences by adding these gating mechanisms.

### 3.2.6 Bi-Directional LSTM

A variation on the basic LSTM architecture, the bidirectional LSTM architecture uses data from both the past and the future to produce predictions [2]. It consists of two LSTM layers, one of which processes the input sequence forward and the other of which processes it backward.

The input sequence is processed by the forward LSTM in the orginal order, from beginning to end. It creates an updated hidden state at each time step using the previous hidden state and the current input. Additionally, the forward LSTM keeps track of the state of the memory cells that store pertinent data from earlier time steps. The hidden state is represented by the forward LSTM's output at each time step, which may be used as an input by layers subsequent to it. The input sequence is processed by the backward LSTM in reverse, that is, from the end to the beginning. It uses the prior hidden state and the current input, much like the forward LSTM, to create an updated hidden state. It also preserves the state of a memory cell. Each time step's backward LSTM output reflects the hidden state and extracts data from the context of the future.

The outputs of both LSTMs are concatenated once both the forward and backward LSTMs have effectively processed the complete input sequence. Information from both the past and future contexts are combined using concatenation. Hence, dependencies in both directions are captured.

### 3.2.7 Defining The Encoder-Decoder Model

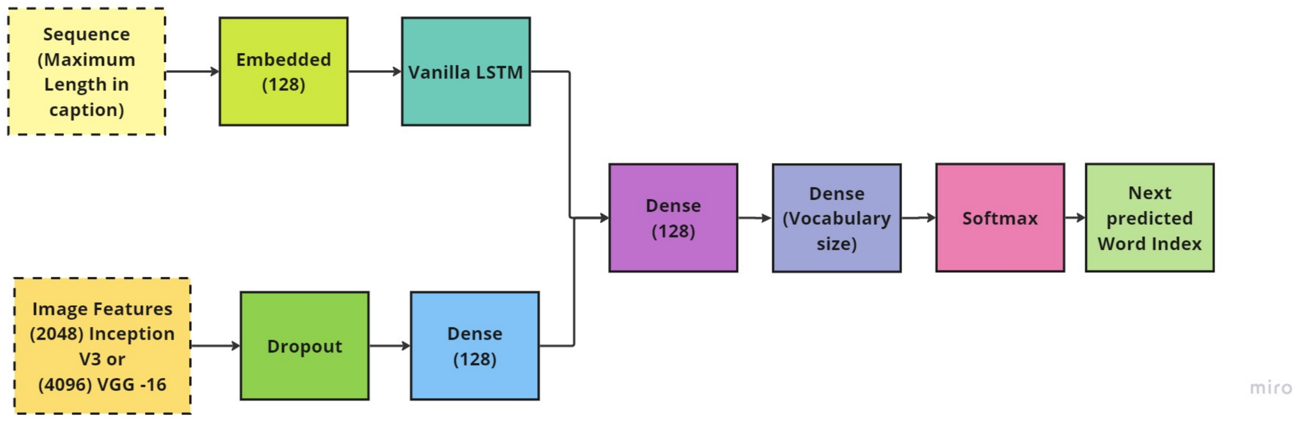
There are two types of inputs, firstly, text, which is a length of numerical sequence and secondly, the image features for the encoder decoder model. For the image features the channel dimension is 2048 for Inception v3 and 4096 for VGG16. The numerical sequence lengths are taken to be equal to that of the maximum length. For the captions that do not have maximum length, we pad the remaining length.

The input image features are then passed onto the dropout layer for regularization or generalization to prevent overfitting. From there the features are then passed onto the dense layer where the dimension is reduced because the dimension of the output of LSTM needs to match with the dimension of the extracted features.

The numerical sequence of the text data will be passed through the embedded layer consisting of 128 units. Output of the embedded layer will be propagated through an LSTM layer which is either a vanilla LSTM or a bi-directional LSTM. In case of vanilla LSTM there are 128 units in the layer and in case of bi-directional LSTM there are 256 units in the layer.

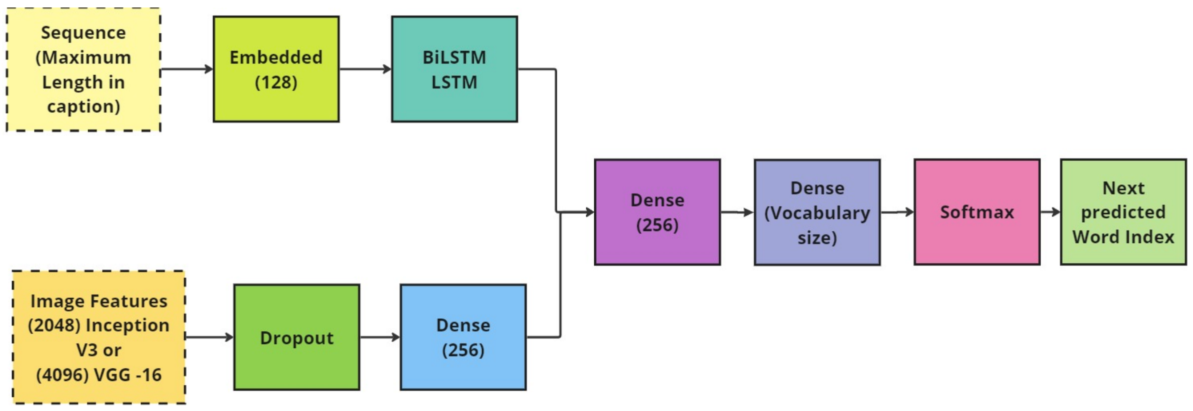
Then we will merge the outputs of the LSTM and dense layer of the two input layers to another dense layer. If the LSTM in the previous layer is vanilla, the dense layer is of 128 units. Otherwise, if the LSTM is bi-directional, the dense layer is of 256 units. The dense layer is processed to generate the ultimate predicted caption.

Since the output of the dense layer is one-hot encoded, we will require the same number of output nodes. The probability distribution obtained is then passed to a softmax function where the index with the highest probability is chosen. This, in turn, becomes the newly generated output in each timestep. The final layer will consist of an equal number of nodes to the size of the set of the vocabulary. The Figures 3.8 and 3.9 show the different architectures of the encoder-decoder models.



Sequence (Maximum Length in caption)
Embedded (128)
Vanilla LSTM
Image Features (2048) Inception V3 or (4096) VGG -16
Dropout
Dense (128)
Dense (128)
Dense (Vocabulary size)
Softmax
Next predicted Word Index
miro

Figure 3.8: Architecture of the encoder-decoder model with Vanilla LSTM as the decoder.



Sequence (Maximum Length in caption)
Embedded (128)
BiLSTM LSTM
Image Features (2048) Inception V3 or (4096) VGG -16
Dropout
Dense (256)
Dense (256)
Dense (Vocabulary size)
Softmax
Next predicted Word Index

Figure 3.9: Architecture of the encoder-decoder model with Bi-Directional LSTM as the decoder.

### 3.2.8 Predicting Captions Using The Model

For caption prediction, features are extracted from the set of images for testing. The same model of CNN used for the training set of images for feature extraction is also used for the feature extraction of the set of test images. If the training model was VGG16, then VGG16 was also used for the test set as well and if the training CNN was Inception v3, then Inception v3 was also used for the test set as well. For predicting the caption token by token, a start token along with specific image features was given as input. The model hence predicted the next word. The newly predicted token was merged with the existing sequence and both the sequence and the same feature vectors were fed to the caption generation model once again. Until the end token was predicted the process repeated itself. Next, another feature vector along with the start token was fed to the caption generation model to predict the next caption following the same process mentioned earlier.

### 3.2.9 Holistic View Of The Proposed Model For Image Caption Generation

In most cases, the encoder-decoder architecture employs LSTM to serve both as an encoder and decoder. In our version of the model, CNN has been utilised to initialise the encoder while RNN, more specifically LSTM, has been used at the decoder end. Two pre-trained models of CNN, namely VGG16 and Inception v3 have been used to extract important features from the medical images. The extracted feature vectors along with text sequences have been merged and served as the input to the generation model. The embedding layer along with the LSTM layer will process the text input. At last, the output from the two layers were merged.

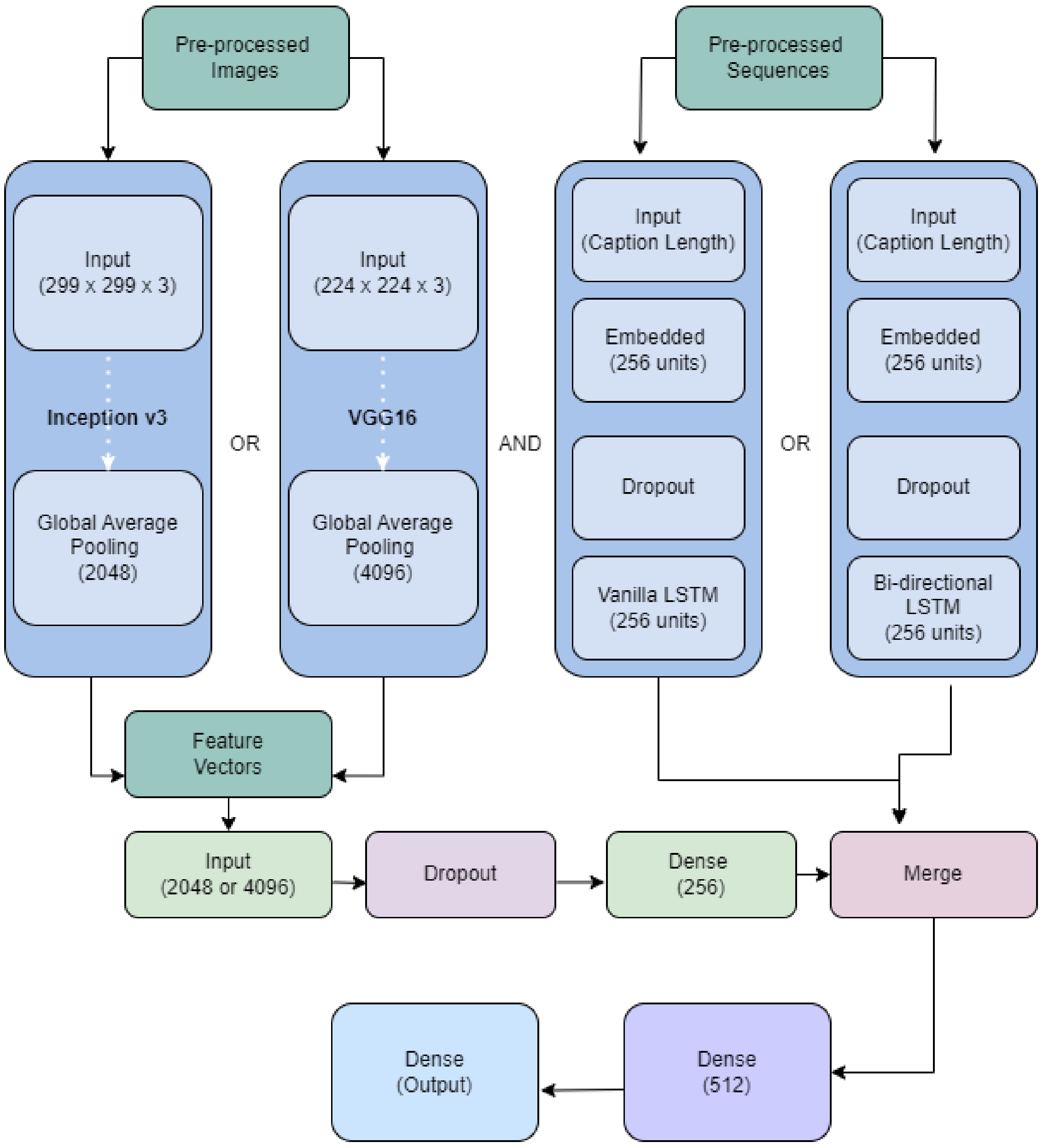


Figure 3.10: Detailed architecture of the hybridized CNN-LSTM model.

# Chapter 4

# Results and Discussion

## 4.1 Performance Metrics

### 4.1.1 Loss Function

If we assume that for a classification based problem, Softmax probabilities (S) and the labels (T), then in terms of equations, categorical cross-entropy can be defined as:

$$L_{CE} = -\sum_{i=1}^{n} T_i log(S_i) \tag{4.1}$$

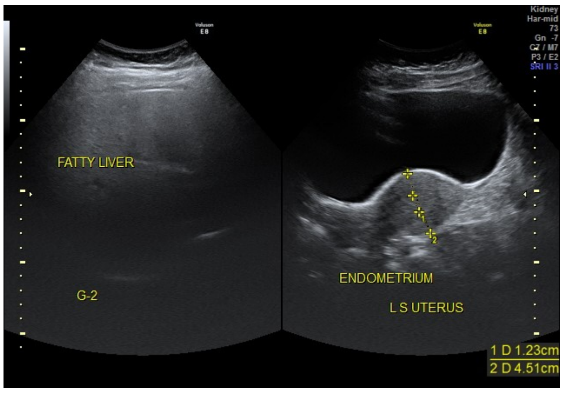
Softmax is continuously differentiable function and therefore, it enables the derivative of the loss function to be calculated for each weight in the neural network[15]. Due to this quality, the model is able to modify the weights in a way that minimizes the loss function and produces results that are near to the real values.

If the current loss is less than previous loss, it implies that the model is learning. Up until the conclusion of training, the optimization process (changing weights to bring the output close to true values) continues.

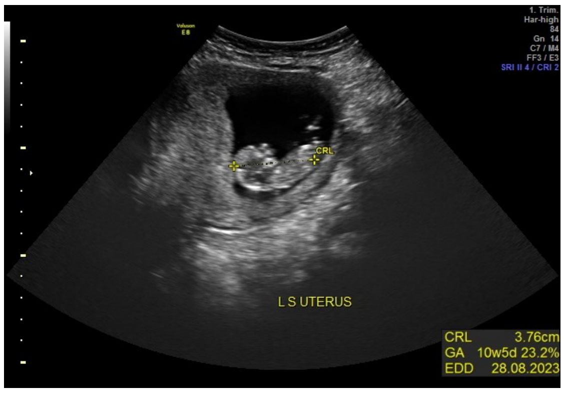
When true labels are one-hot encoded, categorical cross-entropy is utilized. For instance, the true values for the three-class classification issue are [1,0,0], [0,1,0], and [0,0,1].

### 4.1.2 Cosine Similarity

The cosine similarity index [3] calculates how similar two vectors in an inner product space are to one another. It establishes if two vectors are generally pointing in the same direction by calculating the cosine of the angle between them. In text analysis, it is frequently used to gauge document similarity.



Ultrasound image for caption 1.



Ultrasound image of caption 2

Figure 4.1: Ultrasound images for the captions in Table 4.1.

A document may include hundreds of characteristics, each of which tracks the frequency of a certain word or phrase (such as a keyword) inside the text. As a result, every document is an object that may be represented by a term-frequency vector. For instance, Table 4.1 and Table 4.2 show that the term "is" appears three times in actual caption, but the word "uterus" only appears one time for caption 1. The term "containing" is not present anywhere in the whole text, as shown by a count value of 0. These statistics may be quite asymmetrical.

| **No.** | **Actual** | **Predicted** |
|---|---|---|
| 1. | the uterus is bulky in size it is anteverted in position two hypoechoic lesions are seen in fundal region and anterior wall of uterus near to cervix endometrial echo is central and thickness is within normal limit endometrial cavity is empty | the uterus is normal in size it is anteverted in position myometrium shows uniform echotexture endometrial echo is central and thickness is within normal limit no focal lesion is seen endometrial cavity is empty cervix is normal |
| 2. | uterus is gravid containing single gestational sac foetal pole is seen mild fluid collection is seen around perisac area intrauterine pregnancy | uterus is gravid containing single gestational sac foetal cardiac pulsation is present placenta anterior in location amniotic fluid adequate intrauterine pregnancy |

Table 4.1: Comparison of Actual and Predicted Uterus Evaluation

| **Word(caption 1)** | **Actual Frequency** | **Predicted Frequency** |
|---|---|---|
| uterus | 1 | 1 |
| is | 3 | 2 |
| gravid | 1 | 1 |
| and | 1 | 0 |
| retroverted | 1 | 0 |
| in | 1 | 2 |
| position | 1 | 0 |
| a | 1 | 0 |
| single | 1 | 1 |
| gestational | 1 | 1 |
| sac | 1 | 1 |
| seen | 1 | 0 |
| foetal | 1 | 1 |
| cardiac | 1 | 1 |
| pulsation | 1 | 1 |
| present | 1 | 1 |
| intrauterine | 1 | 1 |
| pregnancy | 1 | 1 |
| containing | 0 | 1 |
| placenta | 0 | 1 |
| anterior | 0 | 1 |
| location | 0 | 1 |
| amniotic | 0 | 1 |
| fluid | 0 | 1 |
| adequate | 0 | 1 |

Table 4.2: Actual and predicted word count in caption 1 of Table 4.1.

| **Word(caption 2)** | **Actual Frequency** | **Predicted Frequency** |
|---|---|---|
| uterus | 1 | 1 |
| is | 3 | 2 |
| gravid | 1 | 1 |
| containing | 1 | 1 |
| single | 1 | 1 |
| gestational | 1 | 1 |
| sac | 1 | 1 |
| foetal | 1 | 1 |
| pole | 1 | 0 |
| seen | 1 | 0 |
| mild | 1 | 0 |
| fluid | 1 | 1 |
| collection | 1 | 0 |
| around | 1 | 0 |
| perisac | 1 | 0 |
| area | 1 | 0 |
| intrauterine | 1 | 1 |
| pregnancy | 1 | 1 |
| cardiac | 0 | 1 |
| pulsation | 0 | 1 |
| present | 0 | 1 |
| placenta | 0 | 1 |
| anterior | 0 | 1 |
| in | 0 | 2 |
| location | 0 | 1 |
| amniotic | 0 | 1 |
| adequate | 0 | 1 |

Table 4.3: Actual and predicted word count in caption 2 of Table 4.1.

Term-frequency vectors frequently feature a lot of 0 values and are quite lengthy. Information retrieval, text document grouping, biological taxonomy, and gene feature mapping are examples of applications that use these structures. For such sparse numerical data, the conventional distance metrics do not perform well. For instance, two term-frequency vectors may have a large number of 0 values, indicating that the associated texts do not include many terms in common, but this does not imply that the two papers are comparable. We require a metric that concentrates on the terms that are shared by the two papers as well as their frequency of occurrence. In other words, we need a numeric data metric that does not take into account zero matches.

Cosine similarity can be used to rank documents in relation to a given vector of search terms or to compare documents. Assuming x and y be two vectors that are to be compared, utilizing the cosine metric as a similarity function, we get:

$$sim(x,y) = \frac{x.y}{||x||y||'} \tag{4.2}$$

where $||x||$ is the Euclidean norm of vector $x = (x_1, x_2, ..., x_p)$, defined as $\sqrt{x_1^2 + x_2^2 + ... + x_p^2}$
.
It is the theoretical length of the vector. Similar to that, $||y||$ is vector y's Euclidean norm. It calculates the cosine of the angle formed by the vectors x and y. When the cosine value is 0, it signifies that the two vectors are orthogonal and do not match. The lower the angle and the better the fit between the vectors, the closer the cosine value is to 1. It should be noted that the cosine similarity measure is regarded as a nonmetric measure.

For the actual and predicted captions in Table 4.1 we can assume that the actual and predicted captions are two different documents. Therefore, we can calculate the cosine similarity based on the frequency values in Table 4.2 and Table 4.3.

Here, for caption 1, $x = (1, 3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0)$ and $y = (1, 2, 1, 0, 0, 2, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)$.

By calculating the cosine similarities, we get: $sim(x,y) = \frac{x.y}{||x||y||'} = 0.73$

Again, for caption 2, $x = (1, 3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0)$ and $y = (1, 2, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1)$.

By calculating the cosine similarities, we get: $sim(x,y) = \frac{x.y}{||x||y||'} = 0.62$

The closer the value of cosine similarity is to 1, the more similar are the actual and predicted captions.

### 4.1.3 Naive Approach

We have used two measures for measuring the naive similarity of the actual and predicted words. The first one is the number of correct words present in the predicted caption that are also present in the actual caption taken from the report made by the radiologist and is defined by the following equation:

$$\% \, of \; correct \; words = \frac{\sum_i^n (\sum_j^W \; w_j) : w_j \epsilon \;\; a_i, p_i}{n} * 100\% \tag{4.3}$$

The second measure gives us the percentage of words in the predicted caption that are either redundant or incorrect compared to the actual caption and can be defined by the equation below:

$$\% \, of \; not \; matching \; words = \frac{\sum_i^n (\sum_j^W \; w_j) : w_j \epsilon \;\; p_i \cap w_j \not\epsilon \;\; a_i}{n} * 100\% \tag{4.4}$$

In each of these naive metric equations, $n$ represents number of total sentences in each corpus(actual and predicted), $W$ represents number of total vocabularies and $w_i$ represents the $i^{th}$ word. The $i^{th}$ word in actual caption is represented by $a_i$ and the $i^{th}$ word in predicted caption is represented by $p_i$.
Therefore, if we want to calculate the naive accuracy for caption 1 in Table 4.1 using the values in Table 4.2, firstly, we get:

$\% \, of \; correct \; words = \frac{\sum_i^n (\sum_j^W \; w_j) : w_j \epsilon \;\; a_i, p_i}{n} * 100\% = 72.22\%$
secondly, we get:

$\% \, of \; not \; matching \; words = \frac{\sum_i^n (\sum_j^W \; w_j) : w_j \epsilon \;\; p_i \cap w_j \not\epsilon \;\; a_i}{n} * 100\% = 48\%$

Again, if we want to calculate the naive accuracy for caption 2 in Table 4.1 using the values in Table 4.3, firstly, we get:

$\% \, of \; correct \; words = \frac{\sum_i^n (\sum_j^W \; w_j) : w_j \epsilon \;\; a_i, p_i}{n} * 100\% = 61.11\%$
secondly, we get:

$\% \, of \; not \; matching \; words = \frac{\sum_i^n (\sum_j^W \; w_j) : w_j \epsilon \;\; p_i \cap w_j \not\epsilon \;\; a_i}{n} * 100\% = 59.25\%$

## 4.2 Result Analysis

### 4.2.1 Training Loss Of The Models

We have calculated the training loss by importing categorical cross-entropy from Keras and then compared the training loss of all our four models. The Figures 4.2, 4.3, 4.4 and 4.5 shows the visual representations of the training loss of all the 4 models. Here we can see that all the models converge after 10 epochs. But they do not converge entirely. For better convergence, training the models for 20 to 25 epochs is recommended.

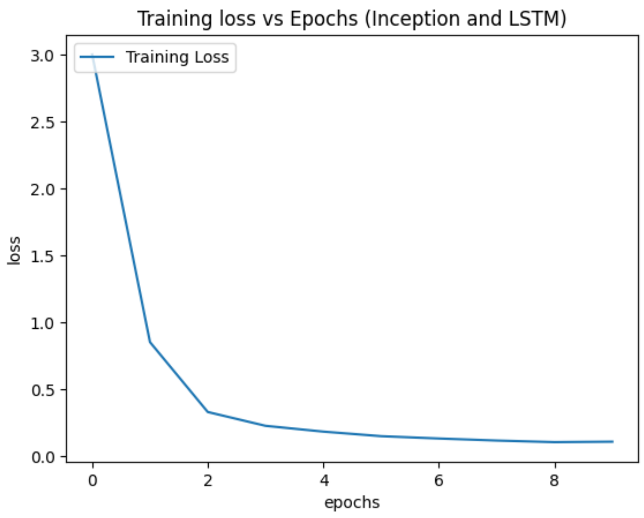


Figure 4.2: Training loss of Inception v3 and Vanila LSTM.

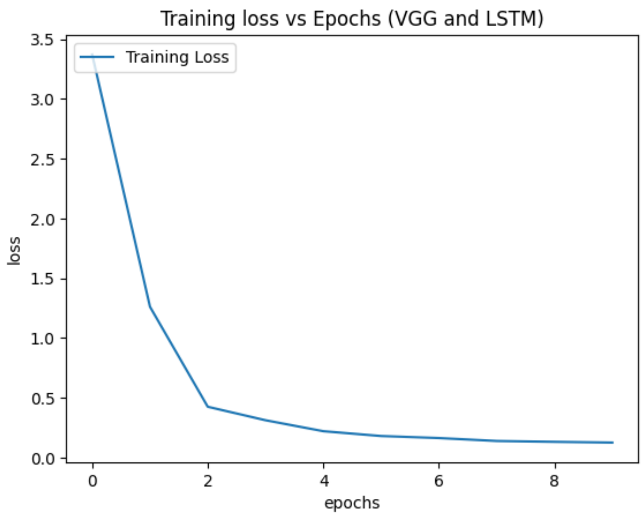


Figure 4.3: Training loss of VGG16 and Vanila LSTM.

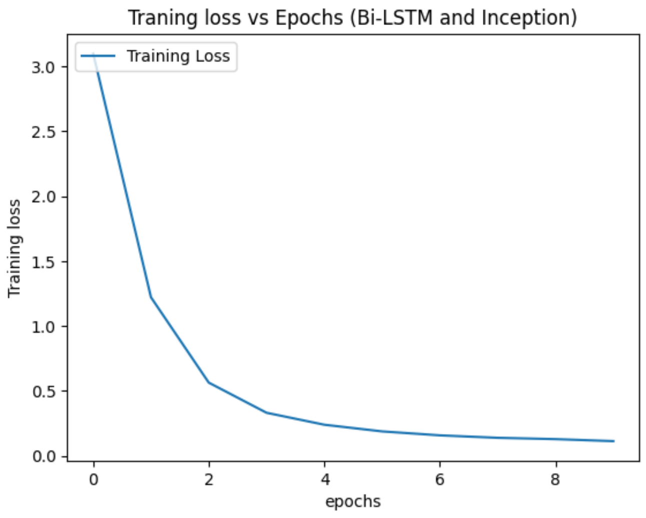


Figure 4.4: Training loss of Inception v3 and Bi-directional LSTM.

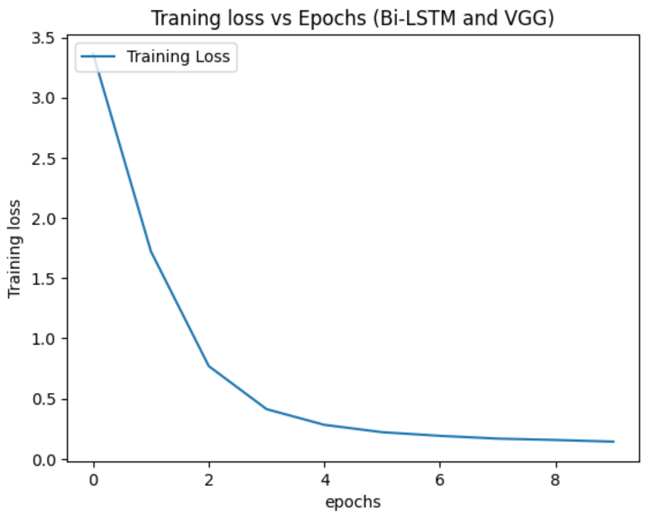


Figure 4.5: Training loss of VGG16 and Bi-Directional LSTM.

### 4.2.2 Training Time Of The Models

Table 4.4 shows the total training time for each of the four model combinations used for this task. The Inception V3 with the vanilla LSTM network took 26 minutes and 10 seconds to train whereas the Bidirectional LSTM model having the same CNN backbone needed 19 minutes and 20 seconds to train itself. On the other hand, the combination of VGG16 and traditional LSTM network required 27 minutes and 21 seconds to train compared to the 22 minutes and 12 seconds training time of the VGG16 and Bidirectional LSTM-based architecture.

| **Result** | **Training Time** |
|---|---|
| Inception v3 + BiLSTM | 19 minutes 20 seconds |
| Inception v3 + LSTM | 26 minutes 10 seconds |
| VGG16 + LSTM | 27 minutes 21 seconds |
| VGG16 + BiLSTM | 22 minutes 12 seconds |

Table 4.4: Model Training Times

### 4.2.3 Cosine Similarity Of The Models

We have also calculated the cosine similarity as an accuracy metric for all the four models. We calculated the cosine similarity for each of the captions and then we have taken the avaerage cosine similarity for all captions combined and converted it into a percentage. Table 4.5 shows the average cosine similarities of all the four models. Here we can see that the inception and Bidirectional LSTM gives the best result with 62.28% cosine similarity. And the VGG16 and Bidirectional LSTM gives the lowest accuracy.

| **Model** | **Cosine Similarity** |
|---|---|
| Inception v3 + BiLSTM | 62.28% |
| Inception v3 + LSTM | 60.39% |
| VGG16 + LSTM | 54.61% |
| VGG16 + BiLSTM | 52.60% |

Table 4.5: Model Cosine Similarity

### 4.2.4 Naive Accuracy Of The Models

A naive accuracy check for the predicted and actual texts has been carried out. We calculated the naive accuracy for each of the captions and then we have taken the avaerage naive accuracy for all captions combined and converted it into a percentage. In this metric, the number of words in actual captions that are also present in the predicted captions has been calculated. The number of words that are not present in the predicted captions but are present in the actual captions have also been calculated as shown in Table 4.6.

| **Model** | **Number of words in predicted caption present in the actual caption (%)** | **Number of words in predicted caption that are not matching with the actual caption (%)** |
|---|---|---|
| Inception v3 + BiLSTM | 76.83% | 45.68% |
| Inception v3 + LSTM | 70.55% | 37.05% |
| VGG16 + LSTM | 59.54% | 53.25% |
| VGG16 + BiLSTM | 69.77% | 50.25% |

Table 4.6: Naive Accuracy Metrics

For the VGG16-LSTM model, 69.77% of the words that are present in the actual captions are also present in the predicted captions and 50.25% of words are not in either predicted captions or the actual captions. The VGG16-Bi-Directional LSTM model produced results where 59.54% of the words that are present in the actual captions are also present in the predicted captions. 53.25% of words are not in either predicted captions or the actual captions. Afterwards, results for the model Inception V3 and LSTM has been derived. The results showcased that 70.55% of the words that are present in the actual captions are also present in the predicted captions and 37.05% of words are not in either predicted captions or the actual captions. Finally, the results for the Inception V3 and Bi-Directional LSTM model show that 76.83% of the words that are present in the actual captions are also present in the predicted captions and 45.68% of words are not in either predicted captions or the actual captions.

## 4.3 Discussion

In results section we have seen which model takes how much time to train fully. The pre-trained CNN models we used, VGG16 and Inception v3 take some time to extract features from images. In comparison to VGG16 the Inception v3 model takes more time to extract the features. Inception v3 takes about 28 seconds to extract all features from the image and the VGG16 model takes 54 seconds, which is more than double the time. In Table 4.1 we can see the total times each model has taken to train on our dataset. From the table we can say that the Inception and Bidirectional LSTM takes the minimum time out of the four models. So the fastest model is Inception and bi directional LSTM which takes 19 minutes and 20 seconds to train while on the other hand the slowest model is VGG16 and vanilla LSTM which takes 27 minutes and 21 seconds.

# Chapter 5

# Conclusion

In our paper, we have curated a dataset of ultrasound images of women and their corresponding textual diagnoses. The dataset includes images mostly from the abdominal region to include the uterus, myometrium, endometrium and cervix in order to obtain an array of impressions related to female health. The lack of techniques for automatic medical caption generation is addressed with the usage of Machine Translation. Two pre-trained CNN models, namely, VGG16 and Inception v3 were used to extract features from the ultrasound images. The feature vectors generated and the text sequences serve as inputs for the encoder-decoder model. Vanilla LSTM and Bi-directional LSTM are the generation models that formulate the annotations that explain the diagnoses from the image content. Four different combinations using VGG16, Inception v3, Vanilla LSTM and Bi-directional LSTM will be used to derive results of varying levels of accuracy. Based on the performance of the models and the percentage accuracy derived from the results, the best model was determined. Our model not only makes medical image interpretation convenient for both patients and medical staff, but also lessen the void that exists in works related to automatic ultrasound image captioning. With further exploration, this model can also be used for similar medical image datasets of different modalities given corresponding textual representations are present. A further scope of research may include, in the stead of discreet values, utilizing continuous data from reports in regards to medical diagnosis.

# Bibliography

[1] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, pp. 1735–80, Dec. 1997. DOI: 10.1162/neco.1997.9.8.1735.

[2] M. Schuster and K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997. DOI: 10.1109/78.650093.

[3] J. Han, M. Kamber, and J. Pei, "2 - getting to know your data," in *Data Mining (Third Edition)*, ser. The Morgan Kaufmann Series in Data Management Systems, J. Han, M. Kamber, and J. Pei, Eds., Third Edition, Boston: Morgan Kaufmann, 2012, pp. 39–82, ISBN: 978-0-12-381479-1. DOI: https://doi.org/10.1016/B978-0-12-381479-1.00002-2. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780123814791000022.

[4] K. Simony and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv 1409.1556*, Sep. 2014.

[5] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2818–2826. DOI: 10.1109/CVPR.2016.308.

[6] T. Baltrusaitis, C. Ahuja, and L. Morency, "Multimodal machine learning: A survey and taxonomy," *CoRR*, vol. abs/1705.09406, 2017. arXiv: 1705.09406. [Online]. Available: http://arxiv.org/abs/1705.09406.

[7] M. M. Leonardo, T. J. Carvalho, E. Rezende, R. Zucchi, and F. A. Faria, "Deep feature-based classifiers for fruit fly identification (diptera: Tephritidae)," in *2018 31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*, 2018, pp. 41–47. DOI: 10.1109/SIBGRAPI.2018.00012.

[8] X.-H. Zeng, B.-G. Liu, and M. Zhou, "Understanding and generating ultrasound image description," *Journal of Computer Science and Technology*, vol. 33, pp. 1086–1100, 2018.

[9] K. Smagulova and A. P. James, "A survey on LSTM memristive neural network architectures and applications," *European Physical Journal Special Topics*, vol. 228, no. 10, pp. 2313–2324, Oct. 2019. DOI: 10.1140/epjst/e2019-900046-x.

[10] C. Zhang, Z. Yang, X. He, and L. Deng, "Multimodal intelligence: Representation learning, information fusion, and applications," *CoRR*, vol. abs/1911.03977, 2019. arXiv: 1911.03977. [Online]. Available: http://arxiv.org/abs/1911.03977.

[11] E. C. Constantinescu, A.-L. Udriștoiu, Ș. C. Udriștoiu, *et al.*, "Transfer learning with pre-trained deep convolutional neural networks for the automatic assessment of liver steatosis in ultrasound images," en, *Med. Ultrason.*, vol. 23, no. 2, pp. 135–139, May 2021.

[12] H. Park, K. Kim, S. Park, and J. Choi, "Medical image captioning model to convey more details: Methodological comparison of feature difference generation," *IEEE Access*, vol. 9, pp. 150 560–150 568, 2021. DOI: 10.1109/ACCESS.2021.3124564.

[13] F. M. Shah, M. Humaira, M. A. R. K. Jim, A. S. Ami, and S. Paul, "Bornon: Bengali image captioning with transformer-based deep learning approach," *CoRR*, vol. abs/2109.05218, 2021. arXiv: 2109.05218. [Online]. Available: https://arxiv.org/abs/2109.05218.

[14] J. Shah, R. Jain, V. Jolly, and A. Godbole, "Stock market prediction using bi-directional lstm," in *2021 International Conference on Communication information and Computing Technology (ICCICT)*, 2021, pp. 1–5. DOI: 10.1109/ICCICT50803.2021.9510147.

[15] K. E. Koech, *Cross-entropy loss function*, Jul. 2022. [Online]. Available: https://towardsdatascience.com/cross-entropy-loss-function-f38c4ec8643e.

[16] M. Samin-Al-Wasee, P. S. Kundu, I. Mahzabeen, T. Tamim, and G. R. Alam, "Time-series forecasting of ethereum price using long short-term memory (lstm) networks," in *2022 International Conference on Engineering and Emerging Technologies (ICEET)*, 2022, pp. 1–6. DOI: 10.1109/ICEET56468.2022.10007377.

[17] *Convolutional neural networks - andrew gibiansky*, https://andrew.gibiansky.com/blog/machine-learning/convolutional-neural-networks/, (Accessed on 06/03/2023).

[18] S. Srivastava, *Machine Translation(Encoder-Decoder Model) !! — medium.com*, https://medium.com/analytics-vidhya/machine-translation-encoder-decoder-model-7e4867377161, [Accessed 29-May-2023].