Automated Radiology Generated System

Ankith Vijay #1
SID: 862394125
NetID: avija016

Shubham Sharma #2
SID: 862394567
NetID: sshar180

Mohit Asudani #3
SID: 862393012
NetID: masud001

ABSTRACT

Medical imaging, which includes X-rays, CT scans, and MRIs, plays a vital role in the diagnosis and treatment of diseases. However, processing these images can be time-consuming and prone to errors due to the specialized training and expertise required for accurate interpretation. To address this challenge, we are developing automated systems capable of generating radiology reports from medical images with minimal human intervention. This innovative approach aims to reduce the time and resources needed for report generation while improving the accuracy and consistency of interpreting medical images. Additionally, implementing automated systems would allow healthcare professionals to allocate more time to crucial tasks, such as patient care. During our endeavor, we encountered several obstacles. Firstly, each patient had multiple types of information, necessitating the identification of relevant features. Secondly, some patients lacked finding reports in the dataset. Thirdly, the number of images varied among patients, with some having four images while others had two. Furthermore, the report texts were generally lengthy, with 99 percentile of the data having a length of 79.0 words. Consequently, we processed the data accordingly.

ACM Reference Format:
Ankith Vijay #1, Shubham Sharma #2, and Mohit Asudani #3. 2023. Automated Radiology Generated System. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 8 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

The use of medical imaging is crucial to diagnosing and treating a variety of illnesses, such as pneumonia, pneumothorax, cancer, cardiovascular disorders, and musculoskeletal conditions. Images from radiology and pathology are indispensable tools used in healthcare facilities around the world. Professionals with extensive knowledge of their fields, such as radiologists and pathologists, require specialized training to interpret and report medical images accurately.

Nowadays, medical imaging is one of the most widely used diagnostic methods in healthcare. In radiology reports, radiologists point out abnormalities, the size of lesions, and the location of lesions observed in medical images. There is however a significant increase in workload due to the increasing patient population and a lack of experienced radiologists. Radiologists are often overwhelmed, having to conduct numerous imaging examinations and generate corresponding reports daily. This heavy workload not only poses a risk of misdiagnosis but also calls for promising approaches to automate report generation.

Writing medical-imaging reports can be particularly challenging for less-experienced radiologists and pathologists, especially those in rural areas with limited access to quality healthcare. Accurately interpreting chest x-ray images, for instance, requires a comprehensive understanding of thoracic anatomy, chest disease physiology, analytical skills, temporal evaluation, clinical history, and correlation with other diagnostic results.

Even experienced radiologists and pathologists face time-consuming and labor-intensive processes when generating imaging reports. In countries with large populations, like India, radiologists may face the daunting task of reviewing hundreds of images and composing corresponding reports. This workload increases stress and fatigue, ultimately impacting the efficiency and accuracy of the reporting process.

To address these challenges and enhance the efficiency of medical-imaging reporting, there is a growing interest in developing automated systems capable of generating radiology reports from medical images. These systems aim to alleviate the burden on medical professionals, improve accuracy, and streamline the reporting process. Leveraging advanced technologies, such as artificial intelligence and natural language processing, these automated systems have the potential to revolutionize the field of medical imaging and enhance patient care outcomes.

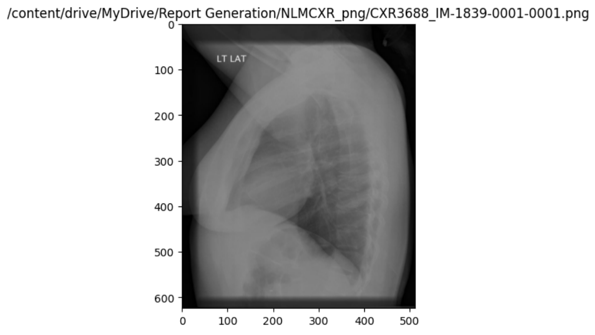

/content/drive/MyDrive/Report Generation/NLMCXR_png/CXR3688_IM-1839-0001-0001.png
LT LAT
0
100
200
300
400
500
600
0
100
200
300
400
500

Figure 1: X-Ray Image from Dataset

In order to create a comprehensive diagnostic report, it is necessary to incorporate various forms of heterogeneous information. The process of assembling a chest X-ray diagnostic report is depicted in Figure 1. It consists of a sentence that convey an impression, a paragraph detailing the findings, and tags describing relevant

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
Conference'17, July 2017, Washington, DC, USA
© 2023 Association for Computing Machinery.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM...$15.00
https://doi.org/XXXXXXX.XXXXXXX

keywords. Creating a unified framework capable of generating these different types of information poses significant technical challenges. To meet this challenge, we propose a multi-task framework that converts the prediction of tags into multi-label classifications and the production of detailed descriptions into text generation.

The second challenge lies in effectively localizing image regions and associating them with the appropriate descriptions, which can be quite demanding. These issues are addressed through the introduction of a co-attention mechanism that enables simultaneous attention to both images and predicted tags, effectively leveraging the synergistic effects of visual and semantic information. Another challenge we face is generating long reports. To address this, we use the way that reports are structured and create a hierarchical LSTM model. This approach allows us to generate long reports by first creating high-level topics and then creating detailed descriptions for those topics. By combining this hierarchical LSTM with a co-attention mechanism, we can effectively produce comprehensive and relevant reports.

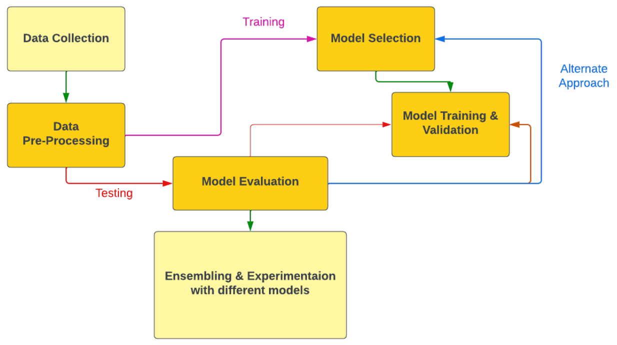


Figure 2: Workflow

2 RELATED WORK

In the field of automated radiology report generation from medical images, numerous pivotal studies have been conducted that have significantly shaped this research domain.

A significant study by Li and colleagues[2] introduced a method that automated the generation of radiology text reports for chest X-rays using a multi-task learning strategy. By learning from image-sentence pairs, the model could classify diseases and construct relevant reports from chest X-rays. This work underscored the potential of multi-task learning in the context of automated radiology report generation.

Another seminal work was undertaken by Jing et al. [1], who proposed an integrated model that harnessed the power of Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN) to extract visual details and create reports, respectively. This model demonstrated the viability of using CNN and RNN in concert for the purpose of automating report generation.

An alternative approach was explored by Wang et al. [4], where the Attention mechanism was incorporated into the automated generation of radiology reports. The use of CNN for visual feature extraction and RNN for report generation remained, but the novel Attention mechanism allowed the model to shift focus on different parts of the image during the report generation, enhancing the relevance of the output.

Zhang et al. [5] ventured down a different path, utilizing a pre-trained CNN model (CheXNet) for extracting visual features from chest X-ray images. The extracted features were processed through a sequence-to-sequence model to generate the final radiology reports. This research accentuated the value of using pre-trained models like CheXNet, which are specifically trained on X-ray images, to elevate the quality of feature extraction and report generation.

Addressing the challenge of generating long reports, Liu et al. [3] devised a Hierarchical LSTM model. This innovative approach generated comprehensive reports by first establishing high-level topics and then crafting detailed descriptions for each topic. The model's success underlined the potential of Hierarchical LSTMs for generating extensive reports.

Our study mirrors the research objectives of these seminal works, focusing on automating the generation of radiology reports from medical images. However, our approach aims to tackle the identified challenges differently, with a view to crafting a more detailed and accurate report generation system. We are exploring a novel amalgamation of Encoder-Decoder, Attention Mechanism, Hierarchical LSTM techniques, and a co-attention mechanism. This combination allows our system to pay simultaneous attention to both the images and the predicted tags.

3 DATASET : CHEST X-RAYS

Overview: The Chest X-rays dataset from Indiana University is a collection of chest radiographs used for various medical imaging research purposes. It consists of a large number of chest X-ray images along with associated findings and clinical reports. The dataset is widely used in the field of medical imaging and healthcare research to develop and evaluate machine learning algorithms for chest X-ray analysis, disease detection, and medical diagnosis.

Dataset Composition: The dataset comprises a diverse range of chest X-ray images captured from different patients with various conditions and diseases affecting the thoracic region. The images are labeled with corresponding findings and clinical reports, providing valuable information about the observed abnormalities or indications present in the X-rays. The dataset covers a wide range of thoracic conditions, including but not limited to pneumonia, lung nodules, pulmonary edema, pleural effusion, and pneumothorax.

Size and Resolution: The Chest X-rays dataset is large in scale, consisting of thousands of chest X-ray images. The exact size may vary depending on the specific version or subset of the dataset being utilized. The images have varying resolutions, but they are typically standardized to a common size for consistency and ease of analysis. Common resolutions include 1024x1024 pixels or 512x512 pixels, depending on the source and preprocessing techniques applied.

Data Acquisition and Preprocessing: The data pre-processing steps were performed to prepare the input data for training and ensure its compatibility with the chosen model architecture. The following pre-processing steps were applied to the data:

1. Image Processing: The images were processed to standardize their size and enhance their quality. They were resized to a fixed resolution of 224x224 pixels using the torchvision.transforms.Resize() function. This resizing step ensures that all images have a consistent size, which is required by the pre-trained DenseNet-121 model used for feature extraction. Additionally, the images were converted to the RGB color space using the convert('RGB') method from the PIL library to ensure uniform representation across all images.

2. Text Processing: The textual reports associated with the images underwent a series of pre-processing steps to clean and normalize the text data. The following text pre-processing steps were applied:

- Lowercasing: All text was converted to lowercase using the lower() method.
- Decontractions: Common contractions in the text were expanded to their full forms using a predefined decontractions dictionary.
- Punctuation Removal: Punctuation marks were removed from the text using regular expressions.
- Number Removal: Numbers were removed from the text using regular expressions.
- Word Filtering: Words shorter than 3 characters were filtered out from the text, except for the special cases of "no" and "ct" which were retained.
- Multiple Full Stops: Multiple consecutive full stops were replaced with a single full stop using regular expressions.
- Full Stop Separation: Full stops followed by a space were separated by an additional space.
- Multiple Spaces: Multiple consecutive spaces were replaced with a single space using regular expressions.
- Separating Starting Words: Starting words with a full stop were separated from the preceding word by an additional space.
- Apostrophe Removal: Apostrophes were removed from the text using regular expressions.

3. Tokenization: The pre-processed textual reports were tokenized using the BERT tokenizer from the Hugging Face transformers library. Tokenization breaks down the text into individual tokens, which are the basic units for language modeling. The BERT tokenizer splits the text into subwords using WordPiece tokenization, which helps in capturing morphological variations and improving model performance.

4. Sequence Timeline: The tokenized reports were further processed by adding special tokens to denote the start and end of the sequence. The "startseq" and "endseq" tokens.

4 PROPOSED METHOD

4.1 Encode-Decoder

Image analysis plays a crucial role in Automated Radiology Report Generation Systems by extracting meaningful information from images and translating it into descriptive text. Convolutional neural networks (ConvNets) have emerged as a powerful machine learning technique for image analysis, finding applications in hyperspectral image classification, channel reduction, and textual image analysis. ConvNets consist of interconnected layers of neurons that enable the analysis and interpretation of images.

By leveraging ConvNets, we can analyze images and extract valuable insights that can be harnessed by a natural language generator. This generator is responsible for producing human-readable descriptions that accurately depict the contents of the images. This process falls under the domain of image captioning, which aims to generate textual descriptions that effectively represent the visual information captured within the images.

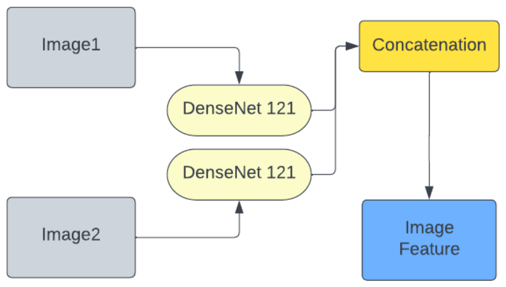


Figure 3: Image Feature Extraction

To comprehend the input images, we employ a strategy of converting each image into fixed-sized vectors. This conversion process can be achieved through transfer learning using pre-trained models. However, it is important to note that the aforementioned pre-trained models are trained on extensive datasets that differ significantly from X-ray images. Consequently, employing transfer learning from these models may yield suboptimal results in the case of X-ray images. To address this limitation, we require a pre-trained model specifically trained on X-ray images. Fortunately, we have access to the CheXNet model, which is a 121-layer convolutional neural network trained on the ChestX-ray14 dataset. This dataset comprises over 100,000 frontal-view X-ray images encompassing 14 different diseases. In order to obtain image features using the CheXNet model, we will remove the final classification layer and extract the resulting output.

The image features are processed through a multi-label classification network, which consists of word-embedded vectors representing each tag in the tag vocabulary. This allows the network to predict the relevant tags for the image. Afterwards, the visual and semantic features of the image are fed into a co-attention model. This model generates a context vector that captures both the visual and semantic information in a unified manner. At this stage, the encoding process is finished.

Our objective now at this stage is to transform the output of the encoder into text by utilizing LSTM networks. LSTM, known as a sequence-to-sequence model, is particularly suitable for handling textual data and generating coherent sequences of words. We feed the encoder output and the embedding vector of the previous word (at time step t-1) into the LSTM layer as inputs. The LSTM layer then predicts a vector representation for the word at time step t. To convert this vector into a meaningful word, we apply a softmax layer and utilize the argmax function to select the word from our vocabulary. This iterative process allows us to generate a sequence of words that form the desired text output.

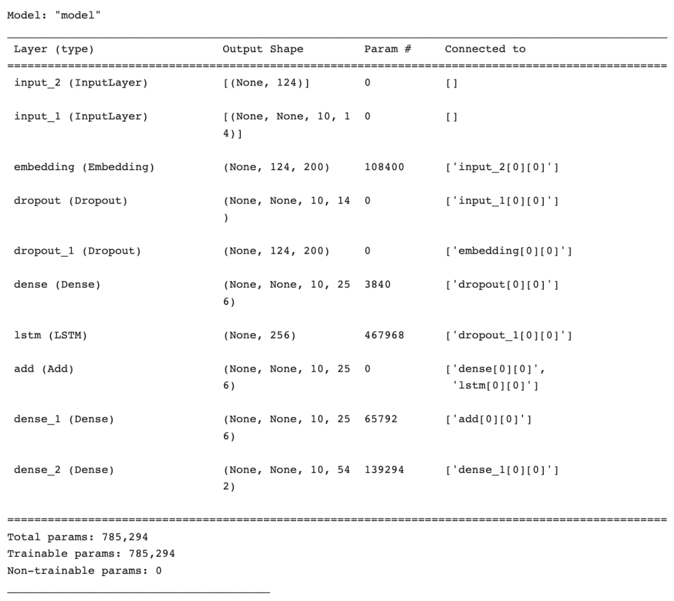
Model: "model"
Summary: The model architecture consists of an input layer for image features and an input layer for text captions. It utilizes an embedding layer, two dropout layers, a dense layer, an LSTM layer, an addition layer, and two additional dense layers. The model takes image features as input and generates text captions based on the given images.

Layers and Output Shapes:

- Input Layer (Image Features): Input shape (None, None, 10, 14)
- Input Layer (Text Captions): Input shape (None, 124)
- Embedding Layer: Output shape (None, 124, 200)
- Dropout Layer: Output shape (None, None, 10, 146)
- Dropout Layer: Output shape (None, 124, 200)
- Dense Layer: Output shape (None, None, 10, 256)
- LSTM Layer: Output shape (None, 256)
- Addition Layer: Output shape (None, None, 10, 256)
- Dense Layer: Output shape (None, None, 10, 256)
- Dense Layer: Output shape (None, None, 10, 542)

Parameters:

- Total Parameters: 785,294
- Trainable Parameters: 785,294
- Non-trainable Parameters: 0



```
Model: "model"
__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
==================================================================================================
 input_2 (InputLayer)           [(None, 124)]        0           []

 input_1 (InputLayer)           [(None, None, 10, 1  0           []
                                4)]

 embedding (Embedding)          (None, 124, 200)     108400      ['input_2[0][0]']

 dropout (Dropout)              (None, None, 10, 14  0           ['input_1[0][0]']
                                )

 dropout_1 (Dropout)            (None, 124, 200)     0           ['embedding[0][0]']

 dense (Dense)                  (None, None, 10, 25  3840        ['dropout[0][0]']
                                6)

 lstm (LSTM)                    (None, 256)          467968      ['dropout_1[0][0]']

 add (Add)                      (None, None, 10, 25  0           ['dense[0][0]',
                                6)                                'lstm[0][0]']

 dense_1 (Dense)                (None, None, 10, 25  65792       ['add[0][0]']
                                6)

 dense_2 (Dense)                (None, None, 10, 54  139294      ['dense_1[0][0]']
                                2)

==================================================================================================
Total params: 785,294
Trainable params: 785,294
Non-trainable params: 0
__________________________________
```

Figure 4: Model and its Parameters

Explanation: The model takes image features as input, which have a shape of (None, None, 10, 14). It also takes text captions as input, which have a shape of (None, 124). The image features are processed through an embedding layer, which converts them into a dense representation with a shape of (None, 124, 200). The text captions undergo a dropout layer with a shape of (None, None, 10, 146). The embedded captions then pass through another dropout layer, resulting in a shape of (None, 124, 200).

The dense layer takes the output of the second dropout layer, and the LSTM layer takes the output of the dropout layer for further processing. The LSTM layer reduces the sequential information to a fixed-size representation with a shape of (None, 256). The addition layer combines the output of the dense layer and the LSTM layer to yield an output of (None, None, 10, 256).

Two dense layers follow, with the first one producing an output shape of (None, None, 10, 256), and the second one generating the final output with a shape of (None, None, 10, 542). These layers contribute to the final output of the model, which represents the predicted text captions based on the given image features.

The model contains a total of 785,294 parameters, all of which are trainable. It serves as a tool for generating captions based on image features, allowing for various applications in image understanding and captioning tasks.

4.2 Encoder-Decoder with Attention Mechanism

In the Encoder-Decoder method, there was a limitation in how the model generated captions for images. The model only considered the overall representation of the image when it is predicting the

```
Data: Image features $image\_features$, Text input
      $text\_input$
Result: Output tensor $out$
---
/* Initial transformations */
$dense1 \leftarrow$ Relu (Linear ($image\_features$))
$dense1 \leftarrow$ Repeat (Unsqueeze ($dense1$, 1), 1,
  $text\_input$.size(1), 1)
$emb \leftarrow$ Embedding_from_pretrained($text\_input$)
$emb \leftarrow$ Cat ($dense1, emb$, dim=-1)
$lstm1\_output, \_ \leftarrow$ LSTM ($emb$)
$hidden \leftarrow lstm1\_output[:, -1, :]$
$a \leftarrow$ Attention ($hidden, lstm1\_output$)
$a \leftarrow$ Unsqueeze ($a$, 1)
/* Apply attention and final transformations */
$weighted \leftarrow$ Bmm ($a, lstm1\_output$)
$lstm2\_output, \_ \leftarrow$ LSTM ($weighted$)
$lstm2\_output \leftarrow$ Squeeze ($lstm2\_output$, 1)
$out \leftarrow$ Linear (Dropout ($lstm2\_output$))
---
return $out$
```

Algorithm 1: Forward pass in EncoderDecoder with attention mechanism model

```
Input: Hidden state $hidden$, Encoder outputs
       $encoder_outputs$
Output: Attention weights $attention$
---
/* Calculate attention scores */
$src\_len \leftarrow$ Length($encoder\_outputs$)
$hidden \leftarrow$ Repeat($hidden$, 1, $src\_len$, 1)
$energy \leftarrow$ Tanh (Linear (Concatenate
  ($hidden, encoder_outputs$)))
/* Apply transformation and softmax */
$attention \leftarrow$ Linear ($energy$)
$attention \leftarrow$ Squeeze ($attention$, 2)
$attention \leftarrow$ Softmax ($attention, dim = 1$)
---
return $attention$
```

Algorithm 2: Attention Weight Calculation

next word in the caption. This approach had a drawback because it might not capture the specific details that are being described by that word. To address this limitation, an attention mechanism is introduced. The attention mechanism allows the model to selectively focus on specific parts or regions of the image while generating each word in the caption. This means that instead of relying solely on the global representation of the entire image, the model can dynamically allocate its attention to relevant areas that are most important for generating the current word.

Let's consider a specific word, the nth word, in the target output sequence. This word is intended to describe a particular portion of the X-ray image rather than the entire image itself. If the model only uses the global image representation for generating this nth word, it may not accurately capture the specific details or nuances required for an accurate description. By incorporating the attention mechanism, the model can adaptively focus on relevant regions or parts of the image during the generation of each word. This allows the model to align the generated words more effectively with the corresponding areas of interest in the X-ray image. As a result, the captions produced by the model become more accurate and meaningful, as they are better able to describe the specific details and features of the X-ray image.

The attention mechanism improves the caption generation process by allowing the model to selectively attend to different regions of the image, ensuring that the generated words are closely aligned with the corresponding areas of interest in the X-ray image. This mechanism enhances the accuracy and relevance of the generated captions, as it captures the specific details and nuances required for a comprehensive description of the image.

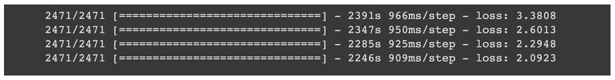
5 TRAINING & EVALUATION

5.1 Training

The training process begins by setting the total number of epochs for which the model will be trained. An epoch represents a complete pass through the entire training dataset, followed by testing of the verification set. Initially, the model is set to training mode, and a variable for accumulating the total loss is initialized to zero.
During each epoch, the following steps are executed:

(1) The model is set to training mode.
(2) A running total of the loss for each batch is initialized to zero.
(3) For each batch of images and reports in the training dataset, the following operations are performed:
  (a) The image and report data are transferred to the appropriate computational device (usually a GPU, if available).
  (b) The report data is one-hot encoded to match the size of the vocabulary.
  (c) The model's forward method is called with the image and report data as inputs, and the model's output is captured.
  (d) The one-hot encoded report data is converted to a float tensor. Then, the loss is calculated by comparing the model's output with the actual reports. The 'view' function is used to reshape the tensors to match the shape expected by the loss function.
  (e) The calculated loss is backpropagated through the network, and the optimizer updates the model's parameters based on this calculated loss.
  (f) The total loss is updated by adding the average loss for the current batch.
(4) After processing all the batches, the average loss for the epoch is calculated by dividing the total loss by the number of batches. This average loss is then printed for inspection.

This loop is repeated for the specified number of epochs. During each epoch, the model learns to adjust its parameters in order to minimize the calculated loss. By the end of the final epoch, the model will have been sufficiently trained on the dataset.
The effectiveness of the trained model often depends not only on the number of epochs but also on factors such as the complexity of the model architecture, the size and quality of the training data, and the choice of optimizer and loss function.



```
2471/2471 [==============================] - 2391s 966ms/step - loss: 3.3808
2471/2471 [==============================] - 2347s 950ms/step - loss: 2.6013
2471/2471 [==============================] - 2285s 925ms/step - loss: 2.2948
2471/2471 [==============================] - 2246s 909ms/step - loss: 2.0923
```

Figure 5: Loss without attention mechanism

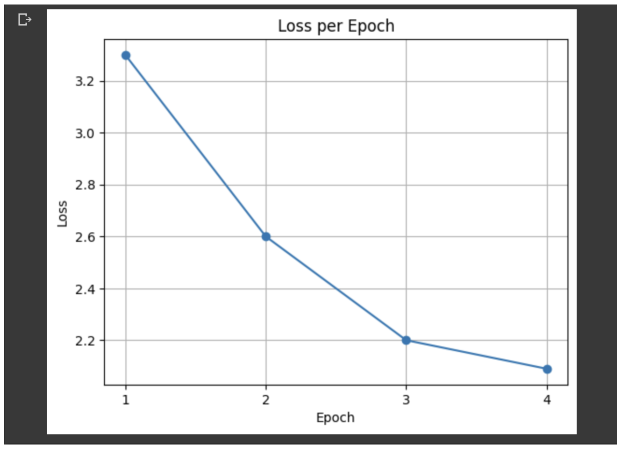


Figure 6: Training Loss(without attention)

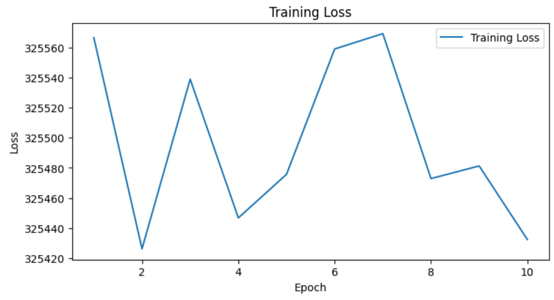


Figure 7: Fine Tuning(Losses not Normalized)

5.2 Validation

Following the training process, the model is validated to assess its generalization capability on unseen data. This involves testing the model with the validation data set and computing the validation loss, which provides an indication of how well the model is expected to perform on new data.
In the validation process:

```
Data: Number of epochs $num\_epochs$, Model $model$,
      Training data loader $train\_dataloader$, Criterion
      $criterion$, Optimizer $optimizer$

for $epoch \leftarrow 1$ to $num\_epochs$ do
    $model$.train()
    $total\_loss \leftarrow 0$
    for each batch $(images, reports)$ in $train\_dataloader$ do
        Move $images$ and $reports$ to device
        $reports\_onehot \leftarrow$ OneHot
         $(reports, num\_classes = vocab\_size)$
        $out \leftarrow model(images, reports)$
        Convert $reports$ to float tensor:
         $reportf \leftarrow reports\_onehot$.Float ()
        Calculate loss: $avg\_loss \leftarrow criterion($View
         $(out, -1, vocab\_size)$, View(reportf, -1, vocab_size))
        $avg\_loss$.Backward ()
        $optimizer$.Step ()
        $total\_loss \leftarrow total\_loss + avg\_loss$.item()
    end
    $avg\_loss \leftarrow total\_loss / len(train\_dataloader)$
    Print: $Epoch[epoch/num\_epochs], Loss : avg\_loss$
end
```

Algorithm 3: Training loop for the EncoderDecoder model

(1) The model is set to evaluation mode. This is crucial as certain layers, such as dropout and batch normalization, behave differently during training and evaluation.
(2) A running total of the validation loss is initialized to zero.
(3) The gradients are not computed during validation to save memory and computation, which is achieved by wrapping the validation loop inside the `t.no_grad()` context.
(4) For each batch of images and reports in the validation dataset, the following steps are performed:
    (a) The image and report data are transferred to the appropriate computational device.
    (b) The report data is one-hot encoded to match the size of the vocabulary.
    (c) The model's forward method is invoked with the image and report data as inputs, and the model's output is captured.
    (d) The one-hot encoded report data is converted to a float tensor.
    (e) The validation loss is calculated by comparing the model's output with the actual reports using the predefined loss function.
    (f) The validation loss for the batch is added to the running total of the validation loss.
(5) After going through all the batches, the average validation loss is calculated by dividing the total validation loss by the number of batches in the validation dataset. This average validation loss is then printed out for review.

The validation process gives us a measure of how well our model is performing on unseen data, which can be particularly helpful in identifying issues such as overfitting during the training process.

```
Data: Model $model$, Validation data loader $val\_dataloader$,
      Criterion $criterion$

$model$.eval()
$total\_val\_loss \leftarrow 0$

/* Loop over validation data */
$t$.no_grad() for each batch $(images, reports)$ in
$val\_dataloader$ do
    Move $images$ and $reports$ to device
    $reports\_onehot \leftarrow$ OneHot
      $(reports, num\_classes = vocab\_size)$
    $outputs \leftarrow model(images, reports)$
    Convert $reports$ to float tensor:
      $reportf \leftarrow reports\_onehot$.Float ()
    Calculate loss: $avg\_val\_loss \leftarrow criterion($View
      $(outputs, -1, vocab\_size)$, View(reportf, -1,
      vocab_size))
    $total\_val\_loss \leftarrow total\_val\_loss + avg\_val\_loss$.item()
end

/* Calculate and print average validation loss */
$avg\_val\_loss \leftarrow total\_val\_loss/len(val\_dataloader)$
Print:  $ValidationLoss : avg\_val\_loss$
```

Algorithm 4: Validation loop for the EncoderDecoder model

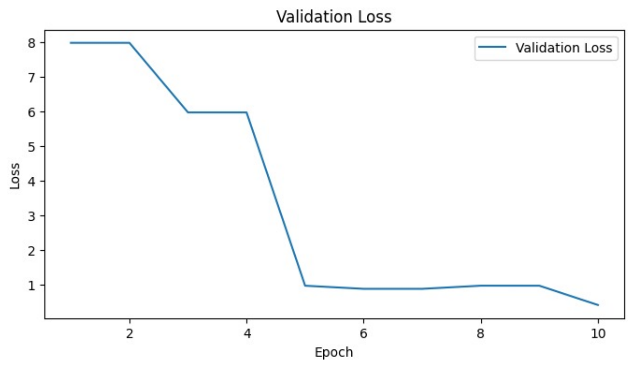


Figure 8: Validation Loss

6 RESULTS

In this section, we present the results of our radiology report generation model. We evaluate the model's performance based on several metrics, including BLEU score.

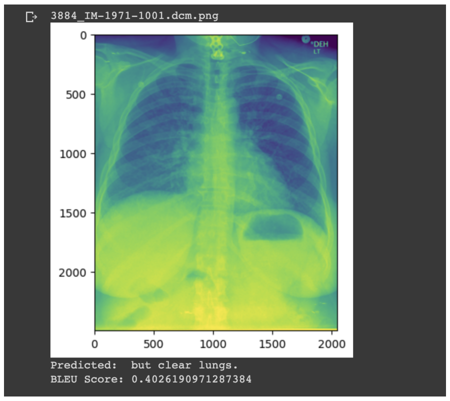
6.1 Evaluation Metrics-BLEU Score

The BLEU (Bilingual Evaluation Understudy) score is commonly used to evaluate the quality of machine-generated text. It measures the similarity between the generated reports and the ground truth reports based on n-gram overlap.

Based on the evaluation metrics, our radiology report generation model achieved highest of 1.6 BLEU score. We also observed a good responsive report generation which is below in the attached images.

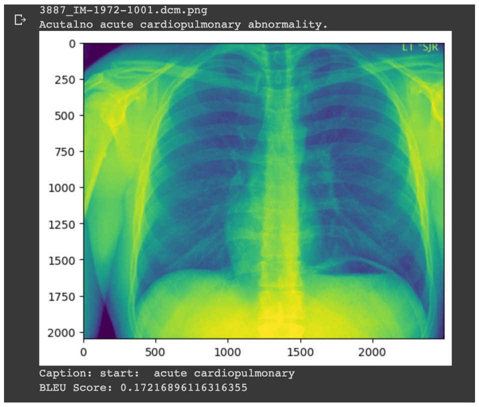
However, we also identified several areas for improvement. Our model predicts nearly correctly for small sequence lengths but it performs worse when the sequence length increases as expected by a simple LSTM network. For future work, we plan to remove such limitations.

Overall, the results demonstrate the potential of our model for automated radiology report generation and its contribution to enhancing efficiency and standardization in the healthcare domain.



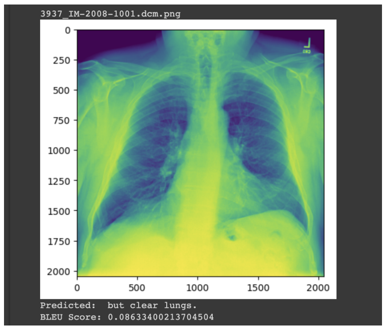
```
3884_IM-1971-1001.dcm.png
Predicted:  but clear lungs.
BLEU Score: 0.4026190971287384
```

Figure 9: without attention)



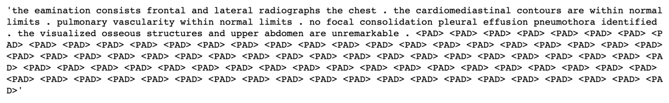
```
3887_IM-1972-1001.dcm.png
Acutalno acute cardiopulmonary abnormality.
Caption: start:  acute cardiopulmonary
BLEU Score: 0.17216896116316355
```

Figure 10: Training Loss(without attention)



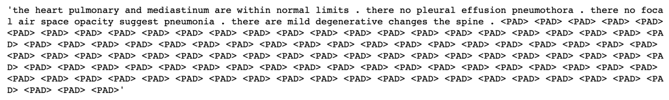
```
3937_IM-2008-1001.dcm.png
Predicted:  but clear lungs.
BLEU Score: 0.08633400213704504
```

Figure 11: Training Loss(without attention)



```
'the eamination consists frontal and lateral radiographs the chest . the cardiomediastinal contours are within normal
limits . pulmonary vascularity within normal limits . no focal consolidation pleural effusion pneumothora identified
. the visualized osseous structures and upper abdomen are unremarkable . <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <P
AD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD>
<PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PA
D> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD>
<PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PA
D>'
```

Figure 12: Report Generation



```
'the heart pulmonary and mediastinum are within normal limits . there no pleural effusion pneumothora . there no foca
l air space opacity suggest pneumonia . there are mild degenerative changes the spine . <PAD> <PAD> <PAD> <PAD> <PAD>
<PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PA
D> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD>
<PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PA
D> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD>
<PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PAD> <PA
D> <PAD> <PAD> <PAD>'
```

Figure 13: Report Generation

7 CONCLUSIONS

We usually get X-Rays for different medical examination like insurance but we don't know the meaning behind the X-Ray as we don't want to go to a doctor and paying a fee. This model will generate approximated radiology reports to let you know of any conditions that you or your loved one might be facing even before meeting the doctor.

In this study, we embarked on an intriguing journey to explore the potential of deep learning in revolutionizing radiology reporting. With a dual arsenal of image features and textual data at our disposal, we endeavored to devise a system capable of automatically weaving together accurate and contextually appropriate radiology reports from a given set of medical images.

Our deep learning model incorporated an encoder-decoder design complemented by an attention mechanism, forming the backbone of our system. We harnessed the power of a pre-trained Convolutional Neural Network (CNN) to distill salient image features from the intricate maze of visual data encapsulated in our input images. Simultaneously, we used pre-trained word embeddings to transform our textual data, infusing a layer of semantic richness into our generated reports.

The encoder segment of our system, a Long Short-Term Memory (LSTM) layer, was tasked with processing the meld of image features and word embeddings. The goal was to encapsulate the essence of the inputs and produce a dense representation. With the attention mechanism acting as a discerning guide, we were able to spotlight crucial parts of the input during the encoding phase. As for the decoder, another LSTM layer took the baton, using the encoded information, sprinkled with attention weights, to churn out sequential outputs, thus crafting the tokens that would form our radiology reports.

Guided by the North Star of a cross-entropy loss function during the training phase, we continually honed our model. The iterative dance of backpropagation and stochastic gradient descent helped whittle down the loss, pushing the boundaries of prediction accuracy.

8 DISCUSSION

Our project, though successful in its own right, has served to illuminate several intriguing avenues for potential improvement and future exploration in the realm of automated radiology reporting.

To start with, the scope of our dataset was somewhat modest, both in terms of size and diversity. As is often the case with machine learning endeavors, having access to a broader and more varied collection of radiology images and corresponding reports could very well augment the model's proficiency in dealing with a more eclectic range of medical cases.

Another promising line of thought is to further adapt our pre-trained models, such as the CNN used for image feature extraction and the word embeddings, to the specific idiosyncrasies of the radiology domain. By fine-tuning these elements, we may be able to tease out more intricate information and catch subtleties that are unique to the realm of medical imaging.

In terms of attention mechanisms, there's certainly room to experiment. Alternatives such as self-attention or multi-head attention could provide our model with a more refined lens, allowing it to zero in on the most crucial areas of the images, ultimately leading to even more accurate reports. One particularly exciting prospect is to enrich our model with external resources, such as medical ontologies or clinical databases. By doing so, we could provide an extra layer of context and a richer domain-specific knowledge base for our model, leading to more precise and context-aware reports. Lastly, but certainly not least, there's the matter of validating the output of our model. Using appropriate metrics to evaluate the generated reports is an obvious necessity, but we should also consider engaging medical professionals for qualitative assessment and feedback. Their expertise would be invaluable in assessing the model's clinical utility and ensuring it could stand up to the demands of real-world scenarios.

REFERENCES

[1] ... Jing. An integrated cnn and rnn model for automated radiology reporting in chest x-ray images. In *Medical Image Computing and Computer Assisted Intervention (MICCAI)*, 2018.

[2] ... Li. Utilizing a multi-task learning methodology for automatic radiology text report generation in chest x-rays. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[3] ... Liu. Using hierarchical lstm for automated radiology report generation. In *6th International Conference on Biomedical Informatics and Technology*, 2021.

[4] ... Wang. Incorporating the attention mechanism in radiology reporting for chest x-rays. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[5] ... Zhang. Leveraging a pre-trained cnn model for automated radiology report generation in chest x-ray images. In *5th International Conference on Machine Learning and Computing*, 2020.