

# **GENG5512 MPE Engineering Research Project Part 2**

## **Automatic Report Generation for Medical Images**

**SHU-CHI LIU**

Student Number: 23831023

School of Engineering, University of Western Australia

**Supervisor: Dr. Naeha Sharif**

School of Physics, Maths and Computing, Computer Science and Software  
Engineering, University of Western Australia

**Co-Supervisor: Dr. Mubashar Hassan**

School of Physics, Maths and Computing, Computer Science and Software  
Engineering, University of Western Australia

*Word count: 7282*

**School of Engineering  
University of Western Australia**

Submitted: 14 October 2024

## DECLARATION

In accordance with University Policy, I certify that:

*The attached work submitted for assessment is my own work and that all material drawn from other sources has been fully acknowledged and referenced.*

### Use of AI tools

Students are permitted to use AI tools for general research purposes and to check and improve the quality of written English in their report. Students are not permitted to use AI tools to generate content.

I have used AI tools in the preparation of this report: Yes

If yes, then provide details:

I use AI tools for general research purposes and to check and improve the quality of written English in reports, but I do not use AI to generate content.

Signed Shu-Chi Liu

Date October 5, 2024

## Project Summary

This project aims to explore the latest research developments in the field of automated medical imaging report generation and to design and develop an automated medical image report generation model to address challenges and issues encountered in the traditional reporting process. Firstly, the conventional process of medical imaging report generation is introduced, detailing each step from data acquisition to report generation, and highlighting issues such as staffing shortage and high-pressure working environments. Subsequently, through a literature review, the current status of research in automated medical report generation is analyzed, focusing on the application of model architectures and technologies. Based on this analysis, the objectives of the project are clarified, namely to develop an automated medical imaging report generation system that surpasses previous models. To achieve this goal, the IU X-ray dataset was selected for training. In terms of model design, Convnext V2 was chosen as the visual component, while LLaMA 3 served as the language component, accompanied by detailed descriptions of their technical features and performance advantages. The model design process is elaborated upon, covering data preparation, setup and training of both the language and visual models, as well as the construction and training of the joint model. After testing and evaluating the joint model, it was found that the model could indeed generate medical reports; however, the text quality metrics scores did not meet the established targets, even though the scores on the METEOR metric were very close to the goal. The low precision of the quantized model and the inappropriate choice of pooling operations could be contributing factors to the suboptimal performance in text quality metrics. Although this project did not fully achieve its intended goals, it demonstrated that Convnext V2 is a viable tool for medical feature extraction, and LLaMA 3 is effective for medical report generation.

## **Acknowledgements**

I would like to express my gratitude to Dr. Naeha Sharif and Dr. Mubashar Hassan for their invaluable assistance and guidance throughout the research process. They were always patient and provided me with many valuable suggestions; without their support, I would not have been able to complete this project. I also appreciate my classmate Bill for providing me with technical support and helping me troubleshoot strange bugs that I encountered.

## Table of Contents

Signed Declaration .....	i
Project Summary.....	ii
Acknowledgements.....	iii
Table of Contents .....	iv
List of Figures .....	v
List of Tables .....	v
1. Introduction.....	1
1.1 The Conventional Process of Medical Imaging Report Generation.....	1
1.2 The Need for Automated Medical Report Generation .....	1
2. Literature Review.....	3
2.1 Automated Medical Imaging Report Generation Models Architecture .....	3
2.2 Review of the Techniques Used in Past Papers for Automated Medical Report Generation.....	3
3. Project Objectives .....	5
4. Model Architecture: Visual and Language Components.....	6
4.1 Visual Component.....	6
4.2 Language Component .....	6
5. Model Design.....	8
5.1 Hardware .....	8
5.2 Language Model Setup.....	8
5.3 Dataset Preparation and Preprocessing .....	9
5.4 Language Model Finetuning.....	10
5.5 Evaluation of Fine-Tuning Performance in Language Generation .....	10
5.6 Visual Model Setup .....	12
5.7 Constructing the Joint Model .....	12
5.8 Joint Model Training .....	15
6. Model Testing and Evaluation .....	17
6.1 Testing .....	17
6.2 Evaluation.....	17
7. Research Reflection .....	19
7.1 Low precision of the quantized model .....	19
7.2 Choice of Pooling Operations .....	19
7.3 Objective Setting Issue: Random Train-Test Split with Different Ratios .....	20
8. Conclusions.....	21
References.....	22

## List of Figures

Figure 2.1: Automated medical imaging report generation models architecture .....	3
Figure 4.1: Performance comparisons of Llama3 with other large language models [18]. .....	7
Figure 5.1: Comparison of ground truth report and generated reports from pretrained and finetuned Llama3.....	11
Figure 5.2: Joint Model's block diagram.....	14
Figure 5.3: Vector Dimension Change Block Diagram .....	14
Figure 6.1: Test results example .....	17
Figure 6.2: Text quality metrics evaluation boxplot.....	18

## List of Tables

Table 3.1: Objectives of Text Quality Metrics.....	5
Table 5.1: Text quality evaluation of generated reports comparing the pretrained Llama 3 model with the finetuned Llama 3 model. ....	11
Table 6.1: Text quality metrics evaluation.....	18

## 1. Introduction

### 1.1 The Conventional Process of Medical Imaging Report Generation

The conventional process of medical imaging report generation begins with the acquisition of patient imaging data through radiological techniques and medical imaging equipment. Common medical imaging modalities include X-rays, CT scans, and MRI scans. For modalities such as CT scans and MRI scans, the imaging data may consist of a series of image slices, whereas for X-rays, the data may be a single image. Subsequently, radiologists or radiological scientists examine and evaluate the images. They scrutinize the images to detect any abnormalities and thereby determine the underlying pathology of the patient, such as fractures, tumors, organ diseases, etc. Finally, based on the assessment of the images, physicians generate reports detailing their findings and diagnostic conclusions.

These reports generally adhere to a standardized format, beginning with essential elements including patient demographics, examination date, image type, findings, and impression [1].

Findings constitute the largest section of the report, providing an objective description by the physician of observed or unobserved abnormalities, normal phenomena, and diagnoses, sometimes structured with tables for enhanced clarity. Impressions, alternatively referred to as conclusions in some medical imaging reports, succinctly summarize the content of the findings section in brief language. Optional elements may include Technique, Indication, and Comparison. Technique denotes the technical aspects employed in image acquisition, encompassing any pre-processing and preparatory actions. Indication pertains to patient-reported symptoms, encompassing descriptions of discomfort, pain localization, onset, and duration. The Comparison section involves the analysis of a series of image slices comprising a case, comparing differences between images taken at different times or from different angles. Physicians utilize templates structured around these elements to craft reports that articulate the conditions depicted in the images clearly, specifically, and employing professional terminology.

### 1.2 The Need for Automated Medical Report Generation

The generation of traditional medical imaging reports necessitates substantial support from automated technological systems, owing to both external environmental factors and inherent internal reasons. Externally, the scarcity of nursing personnel contributes to the deterioration of working conditions. In Taiwan, for instance, according to the "Assessment Study on the Supply and Demand of Nursing and Obstetric Workforce for the Next Decade [2]" proposed by the Ministry of Health and Welfare, it is projected that there will be a shortage of 16,000 to 24,000 nursing personnel in Taiwan by 2024. Radiology, in particular, faces severe personnel shortages. The Directors of the Taiwan Medical Union stated that, on average, there is only one radiographer per 20 hospital beds. However, over the past six years, while there has been a growth of over 100% in the demand for advanced diagnostic imaging and advanced radiation therapy, the number of radiographers has only increased by 26% [3]. The training of radiologists is also time-consuming, with a minimum training period of four years required to qualify as a radiologist [4]. This contributes to the shortage of personnel in radiology. Internally, the generation of medical imaging reports is characterized by high pressure, time consumption, and requisite experience. Hospitals operate within complex organizational environments, necessitating adaptation to various tasks and scenarios. Prolonged exposure to such

high-pressure environments may adversely affect the physical and mental well-being of physicians and could potentially compromise the quality of patient care, posing risks to patients [5]. This highly demanding nature of the work can lead to significant fatigue, with research indicating a turnover rate of 52.10% among radiographers with tendencies to resign [6]. In conclusion, the exploration of efficient and precise automated models to assist radiologists in medical imaging report generation warrants significant research attention.

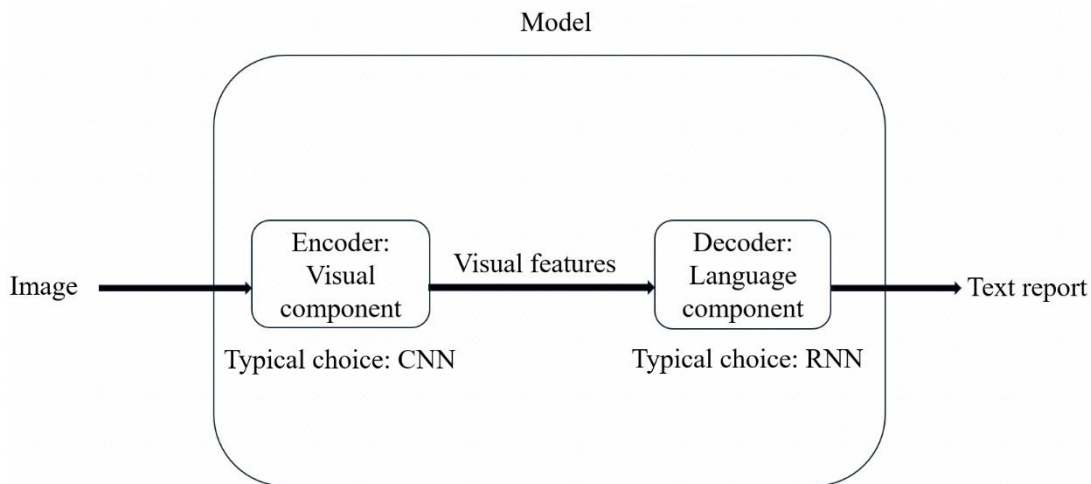


## 2. Literature Review

### 2.1 Automated Medical Imaging Report Generation Models Architecture

In recent years, with the advancements in the field of machine learning, the utilization of this technology to assist research across various domains has rapidly expanded, including numerous studies focusing on the development of automated medical imaging report generation models. The majority of these studies employ deep learning models composed of visual components and language components to design such models, commonly utilizing the typical encoder-decoder architecture. This is a deep learning model architecture that includes the encoder and the decoder as its two main components, where the encoder converts input data into an intermediate representation, and the decoder subsequently transforms this intermediate representation into the desired output data. Model architecture as shown in the Figure 2.1.

In automated medical imaging report generation models, the visual component serves as the encoder, typically employing Convolutional Neural Networks (CNNs) to extract visual features. Convolutional Neural Networks (CNNs) are a class of deep neural networks primarily used in computer vision tasks, such as image classification, object detection, and image segmentation. Subsequently, the language component, acting as the decoder, will transform these visual features into textual reports. The language component is usually designed using recurrent neural networks (RNNs), which are powerful tools in Natural Language Processing (NLP), adept at handling sequential data such as sentences or texts and analyzing the dependencies within them.



**Figure 2.1:** Automated medical imaging report generation models architecture

### 2.2 Review of the Techniques Used in Past Papers for Automated Medical Report Generation.

In Messina et al.'s paper [7], a survey of 40 past research papers focusing on the development of automated medical imaging report generation models was conducted to inventory and compare the technologies employed in these models. The paper highlights Convolutional Neural Networks (CNNs) as the most effective models in the computer vision domain, with nearly all papers utilizing them to design visual components. The DenseNet, ResNet, and VGG architectures are the top three most

widely used techniques in CNN architectures. Furthermore, this project also identified the use of methods based on the Swin Transformer, a neural network architecture based on the Transformer architecture rather than convolutional neural networks, for medical image segmentation tasks. Regarding the design of language components, Transformer models, as well as recurrent neural networks (RNNs) such as Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRUs), have been employed.

Additionally, Messina et al.'s paper [7] raises an important challenge, namely the lack of alternative accuracy evaluation methods besides manual physician review. The only automated accuracy evaluation metric for radiological report generation is the Medical Image Report Quality Index (MIRQI) designed by Zhang et al. [8] However, this evaluation metric has not yet been endorsed by medical experts. Therefore, in the majority of papers, only text quality metrics are used to assess system outputs.

This project compares the performance of various visual component technologies that have been previously utilized, providing a performance ranking. Both the VGG and ResNet architectures participated in the ImageNet Large Scale Visual Recognition Challenge, winning in 2014 and 2015, respectively. According to the competition results provided on the ImageNet website [9][10], ResNet achieved a localization error of 0.09 and a classification error of 0.04, outperforming VGG's errors of 0.25 and 0.07. In the paper by Huang et al. [11], DenseNet was proposed. This architecture surpassed ResNet in classification results on ImageNet, earning the paper the Best Paper Award at CVPR 2017 [12]. The Swin Transformer, introduced by Liu et al. in 2021 [13], demonstrated superior performance compared to ResNet according to the paper, although it did not directly compare with DenseNet. However, based on the performance results provided in their respective papers, the Swin Transformer achieved an 87.3 top-1 accuracy on the ImageNet-1K database, surpassing DenseNet's 79.2 accuracy. In summary, the performance ranking of the visual components from highest to lowest is: Swin Transformer, DenseNet, ResNet, and VGG.

### 3. Project Objectives

The objective of this study is to develop a medical image automatic reporting system that surpasses previous models in terms of Text Quality Metrics. To ensure fair comparison with previous models using a common benchmark, the IU X-ray dataset is utilized for training purposes. The rationale behind selecting this dataset stems from Messina et al.'s paper [7], where it is noted that 24 out of 40 papers surveyed, accounting for 60% of the research, opted for the IU X-ray dataset as their training dataset. Choosing this dataset provides ample sample size for setting the objectives of this project based on the findings of other studies. With this premise established, specific Text Quality Metrics standards can be derived.

In this project, the best scores achieved by previous models in each Text Quality Metric are selected as benchmarks. For instance, the KERP model designed by Li et al. [14] achieved a BLEU-1 score of 0.482 and a BLEU-2 score of 0.325. Leveraging the summary of Evaluation Results of Papers That Use the IU X-ray Dataset table provided by Messina et al. [7], extracting the best scores for each Text Quality Metric from the 24 papers is straightforward. The goals to be achieved for each Text Quality Metric by the project are outlined in the table below.

**Table 3.1:** Objectives of Text Quality Metrics

Section	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE-L	METEOR
Findings	0.482	0.325	0.226	0.199	0.380	0.223

## 4. Model Architecture: Visual and Language Components

The project adopts the encoder-decoder architecture mentioned in the Literature Review, which combines a visual component with a language component in model design.

### 4.1 Visual Component

For the visual component, the project chooses to utilize the Convnext V2 design. Convnext V2 is a novel convolutional neural network model proposed by Woo et al. [15] that is optimized based on Convnext and incorporates two innovations: a fully convolutional masked autoencoder new framework and a new global response normalization layer. This new self-supervised learning technique and architecture endow Convnext V2 with performance surpassing that of previous CNN architectures and the Swin Transformer, demonstrating outstanding performance across multiple image databases.

### 4.2 Language Component

Previous research predominantly employed RNNs to design the language component. However, with the emergence of large language models like ChatGPT, employing them as the language component is prudent. Large language models are massive deep learning models pre-trained on vast amounts of data, mostly based on transformer architecture. Transformers utilize self-attention mechanisms for feature extraction, circumventing the issues of gradient vanishing and exploding encountered in RNNs, allowing for deeper feature extraction. This enables better handling of long-range dependencies and more effective capture of contextual relationships in text compared to RNNs [16]. Transformers do not require a large number of memory units during training and process the entire input sequence in parallel, significantly reducing training time, thereby making it feasible to train models with billions of parameters. Large language models trained on extensive data possess stronger generalization capabilities than RNNs. They exhibit a general language understanding capability to comprehend text and grammar, and their flexibility enables them to learn new knowledge and perform diverse tasks. Fine-tuning, which involves further training pre-trained universal large language models on smaller, specific datasets, refines the model's abilities and enhances its performance on specific tasks or domains [17]. In summary, large language models offer excellent capabilities in handling long-range dependencies and language processing, along with flexibility, multi-functionality, and tunability. These reasons underlie the project's decision to choose them over RNNs for designing the language component.

Among numerous large language models, Llama3 is chosen for this project based on three reasons. Firstly, it is an open-source model. The technology industry has repeatedly demonstrated that open source leads to better, safer, and more secure products, faster innovation, and a healthier market. Additionally, the security of AI is a major concern, and open source is the best way to ensure that no single large company can control AI. The project chooses this model to support this belief. The second reason is the extensive community support. Llama3 provides comprehensive community support, including fine-tuning and code-related documentation, facilitating development. Lastly, it is because of its outstanding performance. As a model released in 2024, Llama3 exhibits state-of-the-art performance. Comparative results of models released with Llama3 show that it is currently the best-

performing open-source model among models of similar size. Performance comparisons of Llama3 with other large language models are illustrated in Figure 4.1 [18].

Meta Llama 3 Instruct model performance				
	Meta Llama 3 8B	Gemma 7B - It Measured	Mistral 7B Instruct Measured	
MMLU 5-shot	68.4	53.3	58.4	
GPQA 0-shot	34.2	21.4	26.3	
HumanEval 0-shot	62.2	30.5	36.6	
GSM-8K 8-shot, CoT	79.6	30.6	39.9	
MATH 4-shot, CoT	30.0	12.2	11.0	

	Meta Llama 3 70B	Gemini Pro 1.5 Published	Claude 3 Sonnet Published	
MMLU 5-shot	82.0	81.9	79.0	
GPQA 0-shot	39.5	41.5 CoT	38.5 CoT	
HumanEval 0-shot	81.7	71.9	73.0	
GSM-8K 8-shot, CoT	93.0	91.7 11-shot	92.3 0-shot	
MATH 4-shot, CoT	50.4	58.5 Minerva prompt	40.5	

**Figure 4.1:** Performance comparisons of Llama3 with other large language models [18].

## 5. Model Design

### 5.1 Hardware

This project requests access to the University of Western Australia's High Performance Computational Research platform, Kaya. Kaya provides robust computational resources, including advanced CPU and GPU configurations, well-suited for computation-intensive tasks. The available hardware resources are as follows:

CPU Resources:

- CPU Name: Intel(R) Xeon(R) Gold 6254 CPU @ 3.10GHz
- Physical Cores: 36 cores
- Total Threads: 36 threads
- Threads per Core: 1 thread

The Intel Xeon Gold 6254 processor is a high-performance server CPU with a significant number of cores and high single-core frequency, making it ideal for large-scale parallel computation and data-intensive tasks. In this project, Kaya's 36-core CPU will provide ample computational power, ensuring efficient training and inference of complex models.

GPU Resources:

- GPU Name: Tesla V100-PCIE-32GB
- Memory: 32 GB

The Tesla V100 is an NVIDIA GPU designed specifically for high-performance computing, deep learning, and data analytics. Each GPU has 32 GB of memory, which is ideal for handling large-scale deep learning model training and inference tasks.

By leveraging Kaya's high-performance CPU and GPU resources, this project will significantly enhance data processing efficiency, accelerate deep learning model training, and optimize the overall workflow, ultimately improving experimental outcomes and reducing processing time.

### 5.2 Language Model Setup

Considering the available hardware resources, the model used in this project is the Variations LLaMA 3 8B, which indicates that this pre-trained model consists of 8 billion parameters. LLaMA 3 model The original LLaMA 3 model and its parameter weights are available on Meta's official GitHub repository. The model is also accessible via Hugging Face's platform. The original checkpoint can be converted using a provided conversion script [19]. After conversion, the model and tokenizer can be loaded and utilized through Hugging Face's Transformers library. However, during practical experiments, a "CUDA out of memory" error occurred when loading the original LLaMA 3 parameters, which was not unexpected. The GPU used in this project, a V100 with 32GB of memory, was insufficient because the LLaMA 3 model uses 32-bit floating point precision, with each parameter occupying 4 bytes. Upon calculation, the model would require 32GB of memory, and when accounting for additional memory allocated by other functions, the required space

exceeds the capacity of the V100. Therefore, this project switched to using the 4-bit pre-quantized LLaMA 3 8B models provided by Unsloth [20]. The 4-bit quantization reduces the model's parameters from high precision to 4-bit integer representation, significantly reducing storage requirements and inference computational resources.

### 5.3 Dataset Preparation and Preprocessing

#### *5.3.1 Preparation*

IU X-ray, introduced by Demner-Fushman et al. in 2016 [21], comprises a database of chest X-ray images paired with their corresponding diagnostic reports. This dataset encompasses 7,470 image-report pairs. The original dataset is available from the National Library of Medicine website [22], provided in the raw DICOM standard. DICOM (Digital Imaging and Communications in Medicine) is a format specifically designed for storing medical imaging data along with associated information. It not only contains image data but also includes metadata related to the patient, scanning parameters, and other relevant details. On the Kaggle platform, a version of the database [23] is available, which has undergone several post-processing steps, including resizing, cropping, linear scaling, and conversion to PNG format. Each image in this processed dataset has been manually classified into either frontal or lateral views. This dataset has a total size of 15 gigabytes and comprises three components: medical images, projection tables, and textual reports. The medical images are stored in .dcm.png format, reflecting the conversion of DICOM images to PNG, while retaining the original file names to indicate their source as DICOM. The textual reports are stored in a CSV file containing seven attributes: uid, MeSH, Problems, image, indication, comparison, findings, and impression.

#### *5.3.2 Preprocessing*

Before fine-tuning the model, it is essential to preprocess the dataset. The preprocessing involves several key steps:

- **Filter:** In this project, only the "findings" column from the textual reports is used for training. The content of this column is extracted, and any missing or empty values are removed to ensure the dataset is clean.
- **Split:** The dataset is partitioned into training and testing sets using the `train_test_split` function. This split is based on unique uid values, with 90% allocated for training and 10% for testing.
- **Tokenization:** Tokenization refers to the process of converting text into a series of tokens. These tokens can be individual characters, parts of words, entire words, or fragments of sentences. The purpose of tokenization is to break the input text into manageable and meaningful units, converting it into a format that is easier for computational models to analyze.
- **Encoding:** Encoding is the process of converting the tokenized units into numerical IDs. Once the text has been tokenized, each token is assigned a unique integer ID, representing its position in the model's internal vocabulary. The process of tokenization and encoding is completed using the tokenizer provided by LLaMA 3. This tokenizer tokenizes the text and constructs a vocabulary while mapping the text to a sequence of integers, facilitating its use for training and processing by machine learning models.

- **Truncation and Padding:** The encoded sequences are truncated and padded to ensure that all sequences have a uniform length. The maximum length of the sequences is set to 128 tokens. Sequences shorter than this length are padded with a special padding token, while those longer than this limit are truncated to fit within the defined maximum length. Padding is managed using the `pad_sequence` method, which pads sequences to ensure that all sequences in a batch are of equal length.
- **EOS Token:** Each sequence is appended with the end-of-sequence (EOS) token, which signals the end of input to the model. The EOS token is added at the end of each encoded sequence to mark its conclusion, ensuring that the model can properly process the entire input during training.

## 5.4 Language Model Finetuning

The fine-tuning of this model employs the traditional fine-tuning process, with the number of fine-tuning epochs set to three. The following are the steps involved in the fine-tuning:

- **Optimizer Setup:** The optimizer is initialized using the AdamW algorithm, which is a variant of the Adam optimizer that includes weight decay. This is set up to optimize the parameters of the model with a learning rate of  $5 \times 10^{-5}$ .
- **Zeroing the Gradients:** At the beginning of each iteration, the gradients of the model parameters are reset to zero. This step is crucial to prevent the accumulation of gradients from previous iterations, which would otherwise lead to incorrect updates.
- **Forward Pass:** The input batch is passed through the model to perform a forward pass. This process involves computing the model's output based on the input data.
- **Loss Calculation:** After obtaining the outputs from the forward pass, the loss is calculated. This loss quantifies how well the model's predictions align with the actual target values.
- **Backward Pass:** The loss is then backpropagated through the model. This step computes the gradients of the loss with respect to the model parameters, allowing for updates to be made during optimization.
- **Optimizer Step:** Finally, the optimizer updates the model parameters based on the calculated gradients. This step applies the weight updates to minimize the loss.

## 5.5 Evaluation of Fine-Tuning Performance in Language Generation

To validate the effectiveness of the fine-tuning, the assessment focuses on whether the fine-tuned language model demonstrates improved understanding of domain-specific language, as well as enhanced accuracy and consistency in text generation. The `model.generate` function is employed to test the fine-tuning results. This function utilizes the first five words of a sample report as a prompt for the model, subsequently generating text based on that prompt. This evaluation is referenced from Deepanshu Jindal's blog [24]. An example of the ground truth report, Pretrained Llama3 Generated Report, and Finetuned Llama3 Generated Report is shown in Figure 5.1.



- **Ground truth report:**

Low lung volumes are present. The heart size and pulmonary vascularity appear within normal limits. The lungs are free of focal airspace disease. No pleural effusion or pneumothorax is seen. There has been interval development of multiple healed left rib fractures. Degenerative changes are present in the spine.

- **Pretrained Llama3 Generated report:**

Low lung volumes are present. However, in the absence of a normal flow-volume loop, it is difficult to quantify them. A low vital capacity indicates airflow limitation, but this may be caused by other conditions, such as neuromuscular disease. The low forced vital capacities (FVCs) observed in patients with airflow obstruction are the result of the combined effects of airflow limitations and expiratory muscle weakness. Forced expirograms are not always reproducible, and sometimes patients cannot perform them.

- **Finetuned Llama3 Generated report:**

Low lung volumes are present. The heart size and pulmonary vascularity appear within normal limits. No focal air space opacities are seen. Patchy air filled lungs are noted.

**Figure 5.1:** Comparison of ground truth report and generated reports from pretrained and finetuned Llama3.

As observed, since the first five words of a sample report serve as a prompt, the initial five words of all generated reports are identical. Another notable feature is the length of the reports; it is evident that the pretrained report is longer, whereas the fine-tuned report is shorter. This difference in length may be attributed to the model's ability to generate more concise reports after training, thereby producing text that is more specific and focused on medical topics while avoiding unnecessary content. To verify this hypothesis, this report employs the text quality metrics Bleu-1, Bleu-2, and Bleu-3 to evaluate all generated reports. Bleu-1, Bleu-2, and Bleu-3 are variations of the BLEU (Bilingual Evaluation Understudy) score, which is a metric used to evaluate the quality of text generated by machine learning models, especially in tasks like machine translation or text generation. Each variation measures the overlap between the generated text and reference text (ground truth) at different levels of precision. Bleu-1 measures the precision of unigrams (single words) in the generated text relative to the reference text, evaluating how many single words in the generated report match those in the reference. This metric focuses on capturing the basic content and relevance of individual words. Bleu-2 builds upon this by assessing the precision of bigrams (pairs of consecutive words), thereby considering both individual word accuracy and the relationships between adjacent words. This allows for capturing some of the structure and flow within the text. Similarly, Bleu-3 extends the analysis to trigrams (three consecutive words), providing a more in-depth evaluation of the text's fluency and coherence. The higher the Bleu-1, Bleu-2, and Bleu-3 scores, the more closely the generated text aligns with the reference, reflecting a higher degree of similarity in both content and linguistic quality [25]. The average text quality metrics Bleu-1, Bleu-2, and Bleu-3 of the reports generated by the Pretrained Llama3 model and the Finetuned Llama3 model are shown in Table 5.1.

**Table 5.1:** Text quality evaluation of generated reports comparing the pretrained Llama 3 model with the finetuned Llama 3 model.

Text Quality Metrics	Pretrained Llama3 model	Finetuned Llama3 model
Bleu-1	0.18	0.38
Bleu-2	0.13	0.33
Bleu-3	0.11	0.30

The results presented in the table support the previous hypothesis, as the text quality evaluation scores for all three metrics in the fine-tuned model's generated reports are higher than those of the pretrained model's reports. This indicates that the fine-tuned model is better equipped than the pretrained model to generate text that closely resembles the reference reports.

## 5.6 Visual Model Setup

The ConvnextV2 visual model can be accessed from the official GitHub repository [26]. Available models include the original model and a pretrained version, which has been trained on the ImageNet-1K dataset. Prior to training, images must be arranged into the specified folder hierarchies. However, during practical implementation, the official installation instructions for ConvnextV2 were found to be lacking in comprehensiveness, especially concerning environmental requirements.

Notably, the documentation did not specify the necessary operating system kernel version, nor did it indicate that the model is incompatible with the Windows environment. For instance, attempts to install ConvnextV2 on a Windows system encountered errors during the installation of the MinkowskiEngine plugin, which is only supported on Linux and macOS. Furthermore, the documentation failed to specify the required version of the operating system distribution. ConvnextV2 is not supported on Ubuntu 21.04, but testing confirmed successful deployment on Linux 22.04. Additionally, there was no mention of the required CUDA version, despite the fact that ConvnextV2 specifically requires CUDA version 11.8. The aforementioned issues represent only a portion of the identified shortcomings.

Relying on the official resources and training methods would significantly delay project progress. Consequently, this project employed the pretrained "facebook/Convnextv2-tiny-1k-224" model provided by Hugging Face [27]. This Convnext V2 model was pretrained using the FCMAE framework and fine-tuned on the ImageNet-1K dataset at a resolution of 224x224.

## 5.7 Constructing the Joint Model

Constructing a Joint Model is a crucial step in connecting the vision component and the language component. In the previous steps, the visual model and language model have already been set up. The next step involves handling the data flow between these two components, where the visual model extracts visual features and passes them to the language model.

### *5.7.1 Embedding vectors and space*

Convnext V2 is a Convolutional Neural Network (CNN) architecture composed of multiple hidden layers, where convolutional layers are used for feature extraction and processing of the input image. In Convnext V2, feature extraction is progressively accomplished through layer-by-layer convolution operations. The shallower layers capture local features of the image, such as edges and textures, while the deeper layers capture more abstract global features. The final convolutional layer, which is also the last hidden layer, plays a crucial role in further processing the high-dimensional features extracted

from the previous layers, outputting a feature tensor. This tensor serves as a high-dimensional representation of the input image after comprehensive processing, containing abstract features of the image [15].

The output of the last hidden layer can be obtained using the `last_hidden_state` method provided by Convnext V2 [28]. The shape of the last hidden state in Convnext V2 is `[batch_size, feature_channels, height, width]`, which represents a four-dimensional tensor. This output is then passed as an embedding vector to the embedding layer of the Llama 3 language model. The shape of the embedding layer in Llama 3 is `[vocab_size (tokens), embedding dimension]`, which is a two-dimensional tensor. Clearly, the shape of the embedding vector and the embedding layer do not match, as the embedding vector's dimensionality is too high. To resolve this, a pooling operation must be applied to reduce the dimensionality.

Pooling is a fundamental operation in Convolutional Neural Networks (CNNs) that serves to reduce the dimensionality of feature maps, emulating the process of information extraction in the human visual system. Often referred to as subsampling or downsampling, pooling compresses feature representations while retaining crucial features, which aids in reducing computational complexity and mitigating noise by filtering out less significant details, thereby decreasing information redundancy. In a pooling layer, a sliding window with a defined stride moves across the input feature map. The stride controls the distance the window shifts at each step, directly influencing the degree of dimensionality reduction. Following pooling, the output is a downsampled version of the input, characterized by reduced spatial dimensions but preserving essential information [29]. In this project, the `pooler_output` method provided by Convnext V2 is utilized to pool the embedding vectors [28]. This method returns the last layer hidden-state after applying a pooling operation over the spatial dimensions. The resulting vector has the shape `[batch_size, hidden_size]`, forming a two-dimensional tensor. This ensures compatibility with the embedding layer of the Language Model, as the shapes now align.

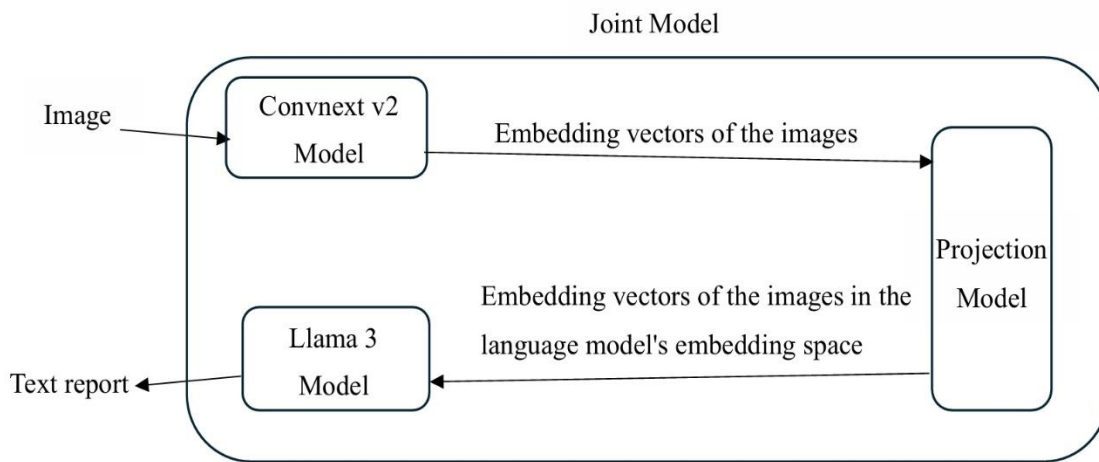
### 5.7.2 Projection Model

Although both the embedding vectors and the embedding space become two-dimensional tensors after the pooling operation, the dimensions in the channels are still not aligned. The embedding vector from the visual model has a shape of `[24, 768]`, where 768 represents the hidden size, i.e., the dimensionality of the hidden states. In contrast, the embedding space of LLaMA 3 has a shape of `[128256, 4096]`, where 4096 corresponds to the dimension of the feature vector for each token in the embedding layer. To align these dimensions, a Projection Model is constructed to act as a bridge between Convnext V2 and LLaMA 3. The purpose of this projection model is to map the visual features into the same dimensional space as the language input, allowing for further integration or comparison between the visual and textual information. This process involves two main steps: applying a linear transformation followed by an activation function.

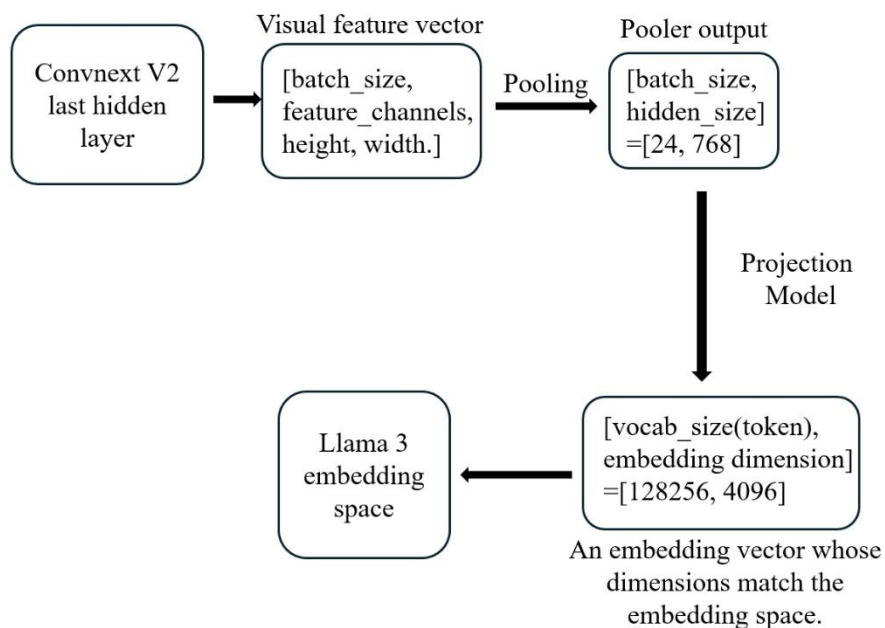
The Fully Connected Layer (a linear layer) transforms the output of the visual model into the embedding dimensions required by the language model. This linear transformation is performed via matrix multiplication and addition, using a weight matrix to project the image embedding vector into the target embedding dimensions. This ensures that the representations of the images and the textual

reports are mapped into the same embedding space, facilitating their effective combination and comparison.

Following the linear transformation, the projection model applies an activation function. This activation function introduces non-linearity by mapping the output of the linear transformation into a specific range, enhancing the model's expressive capacity. In this case, the Tanh activation function is used, which restricts the output range to  $[-1, 1]$ , ensuring that the projected values remain within a manageable range. This step ensures that the projection model's output is more expressive, capable of capturing more complex features, and effectively handling the relationship between images and textual data. The joint model's block diagram as shown in the Figure 5.2 and the vector dimension changes of the joint model are illustrated in the block diagram shown in Figure 5.3.



**Figure 5.2:** Joint Model's block diagram.



**Figure 5.3:** Vector Dimension Change Block Diagram

## 5.8 Joint Model Training

The training aims to train the Joint Model defined in the previous chapter. This training follows traditional methods, involving forward propagation, backpropagation, and parameter updates, with slight variations in model definition. The training steps are as follows:

### 5.8.1 Freezing the Parameters of the Vision Model

In this training, the parameters of the vision model will be frozen, meaning that the parameters of Convnext V2 will not be updated. This is done to execute transfer learning, and freezing parameters is a classic operation in transfer learning. By freezing the parameters of the pre-trained model, the learned knowledge can be transferred to a new task, avoiding the need to train from scratch and speeding up the transfer process [30].

In this project, the vision model, Convnext V2, is already a pre-trained model that has learned rich visual features from the ImageNet-1K dataset. If the parameters of the vision model are not frozen, its features might change during training, leading to the loss of effective features accumulated during pre-training. Freezing the vision model preserves these learned visual features and helps avoid overfitting. Furthermore, freezing the parameters of the vision model ensures that Convnext V2 serves only as a feature extractor, allowing the training to focus on adjusting the language model and projection model, which are the critical components of the current task.

### 5.8.2 Computing Visual Embeddings: Visual Feature Extraction

First, the input image tensors are flattened. The purpose of this flattening is to merge multiple images from each sample into a single dimension so that all images in the batch can be processed simultaneously by the vision model. After that, the `vision_model` is used to extract the image embedding vectors, obtaining the `pooler_output`, which has an output shape of `[batch_size, hidden_size]`.

### 5.8.3 Embedding Dimension Mapping

Through the projection model (`projection_model`), the image embeddings (`img_embed`) are mapped to the embedding dimensions required by the language model. This process effectively transforms the output of the vision model from 768 dimensions to 4096 dimensions required by the language model. This mapping is accomplished through linear transformations.

By implementing these steps, the training process effectively utilizes the strengths of both the visual and textual domains to enhance the performance of the Joint Model.

### 5.8.4 Calculating Language Embeddings

For the input medical report text tokens, the embedding layer of the language model is used to map each token to an embedding vector. This process converts the discrete token representations into continuous vector representations that the model can understand.

### 5.8.5 *Concatenating Visual and Language Embeddings*

The visual embeddings and language embeddings are concatenated to form a complete input embedding, resulting in a combination of image and language embeddings. The purpose of this step is to merge the image information with the text information so that both can be fed into the language model for processing. By fusing multimodal information, the model is enabled to process inputs from both visual and linguistic domains within the same context, enhancing its ability to comprehend the relationships between them.

### 5.8.6 *Forward Propagation*

In this phase, the concatenated visual and language embeddings, which combine both the image features and the textual information, are passed into the language model, LLaMA 3. The model processes this fused input to generate predictions that integrate both modalities. Specifically, the embeddings undergo a series of transformations within the deep layers of the LLaMA 3 model, including attention mechanisms and non-linear activations, which allow the model to capture complex patterns and relationships between the visual and language components.

### 5.8.7 *Loss Calculation*

The model computes the loss based on the output from forward propagation and the provided labels. The loss serves as a measure of the error between the model's current predictions and the true labels. The calculated loss is the cross-entropy loss, a commonly used loss function in classification tasks that quantifies the difference between the predicted distribution and the actual label distribution.

### 5.8.8 *Backward Propagation*

The gradients of the loss function with respect to the model parameters are computed. This is accomplished through the chain rule, which automatically calculates the partial derivatives of each model parameter under the current loss, providing essential gradient information for optimizing the model parameters.

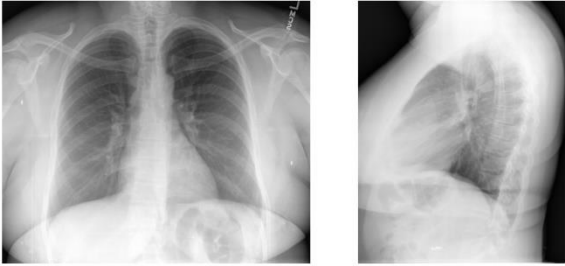
### 5.8.9 *Parameter Updates*

The optimizer is used to update the model parameters. It adjusts the trainable parameters based on the gradients computed during backpropagation, with the aim of reducing the loss. Different parameter groups have different learning rates (lr). Since the LLaMA 3 model has already undergone fine-tuning, the language model uses a smaller learning rate of  $2e-5$ , while the projection model utilizes a larger learning rate of  $5e-5$ . This distinction allows the model to effectively learn from both the pre-trained and newly integrated components while managing convergence efficiently.

## 6. Model Testing and Evaluation

### 6.1 Testing

During the testing and evaluation phase, the model loads the best checkpoints and tokenizers saved during the training process and utilizes the test set as input to assess its performance. The figure below demonstrates this process. By inputting images from the test set, the model generates corresponding medical reports. The model's test results example, including the inputs, outputs, and the reference ground truth report, are shown in Figure 6.1.

- Input: 
- Ground truth report
  - The cardiomediastinal silhouette is normal in size and contour. There is no focal consolidation, pleural effusion or pneumothorax. There is no acute bony abnormality.
- Generated Report:
  - The heart is normal in size and contour. The mediastinum is unremarkable. The lungs are clear without evidence of acute infiltrate, pleural effusion, or pneumothorax. There is no evidence of acute bony abnormality.

**Figure 6.1:** Test results example

### 6.2 Evaluation

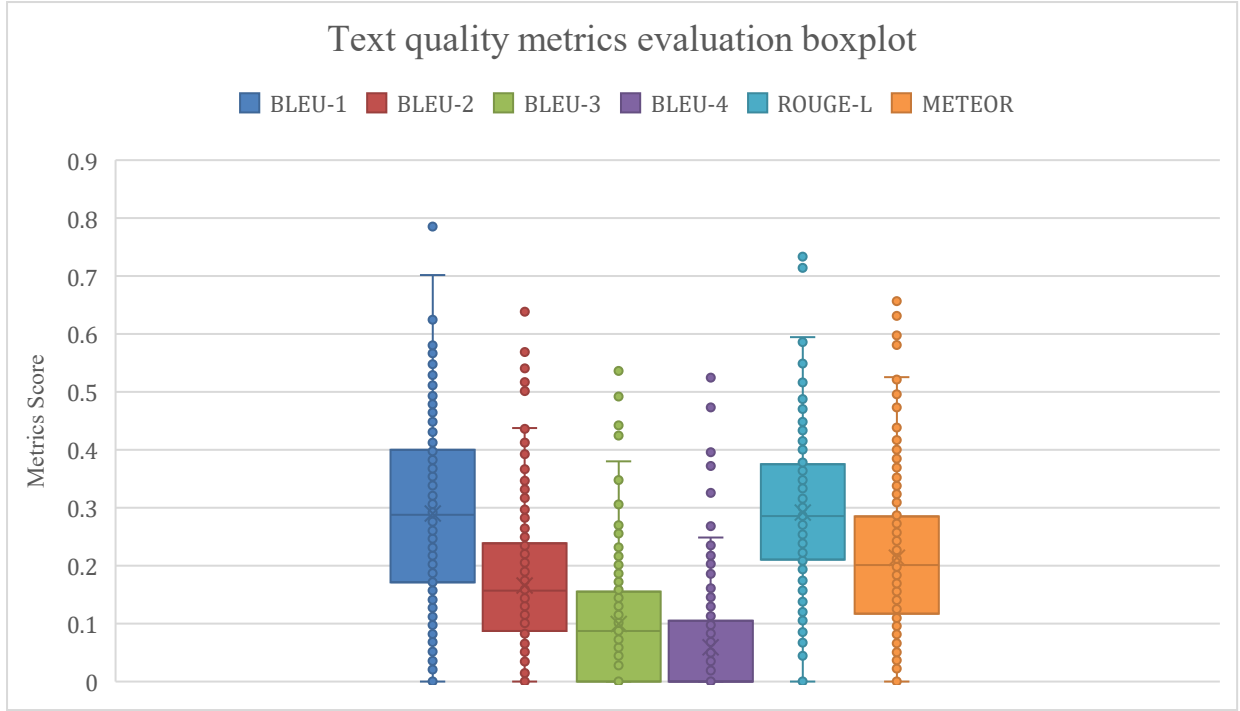
This project evaluates the generated reports using text quality standards, specifically employing metrics such as BLEU-1, BLEU-2, BLEU-3, BLEU-4, ROUGE-L, and METEOR. BLEU has been discussed in previous chapters. ROUGE-L (Recall-Oriented Understudy for Gisting Evaluation) is a metric used for the automatic evaluation of text summarization and machine translation quality, focusing particularly on the similarity between the generated text and the reference text. ROUGE-L specifically emphasizes the Longest Common Subsequence (LCS) of the text, assessing whether the generated content retains the structure and important information of the original text [31].

METEOR (Metric for Evaluation of Translation with Explicit Ordering) is an automatic evaluation metric for assessing the quality of machine translation and text generation. Unlike BLEU and ROUGE, METEOR considers not only lexical matching but also morphological variations, synonyms, word order, and grammatical structure, providing a more comprehensive reflection of the similarity between the generated text and the reference text [32].

Table 6.1 presents the Text Quality Metrics Evaluation of the generated reports along with their targets. Figure 6.2 shows the boxplot of the respective Text Quality Metrics Evaluation, with the boxplots from left to right representing BLEU-1, BLEU-2, BLEU-3, BLEU-4, ROUGE-L, and METEOR.

**Table 6.1:** Text quality metrics evaluation

	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE-L	METEOR
<b>Objectives</b>	0.482	0.325	0.226	0.199	0.380	0.223
<b>Convnextv2+Llama3</b>	0.290	0.166	0.099	0.059	0.292	0.214

**Figure 6.2:** Text quality metrics evaluation boxplot

As the n-gram length increases (from BLEU-1 to BLEU-4), the scores tend to decrease, and the box in the boxplot moves lower. This is due to the inherent characteristics of the BLEU metric, where matching longer word sequences between the generated and reference texts becomes more challenging. This trend is consistent with the evaluation of language models, where shorter n-gram matches (such as in BLEU-1) are more frequent, resulting in higher scores, whereas longer n-grams (such as in BLEU-4) lead to lower scores due to stricter matching criteria.

Additionally, in Figure 13, there are some outliers above the maximum values, and as the n-gram length increases, these outliers persist, with their scores declining at a slower rate. For instance, from BLEU-3 to BLEU-4, their maximum values only drop from 0.542 to 0.524. This reflects a characteristic of the UI database, where the test set contains some data that are extremely similar to the training set. During the evaluation of these test samples, the model generates answers that are very close to the ground truth, achieving high scores across various n-gram lengths.

Compared to BLEU and ROUGE, the METEOR metric undoubtedly performed the best in this project, with its score being only 4% lower than the target. As mentioned earlier, METEOR provides a more comprehensive reflection of the similarity between the generated text and the reference text. This suggests that while LLaMA 3 may not have precisely matched the vocabulary used in the ground truth reports, it was still able to convey a meaning that is close to the intended message overall.



## 7. Research Reflection

It is evident that the text quality metrics in this project have not met the expected standards. This section attempts to analyze potential causes and solutions.

### 7.1 Low precision of the quantized model.

This project utilizes a 4-bit quantized model provided by Unsloth. By reducing the model weights from 32-bit floating point to 4-bit, the model's storage requirements and inference computation resource demands are significantly lowered. This decision was necessary due to insufficient GPU memory. However, since the 4-bit quantization has a smaller representation range, the precision of the weights is limited, leading to increased quantization errors and introducing more quantization noise. This can result in the model being less capable of capturing subtle data differences, as well as inaccurate weight updates, which in turn affects the model's learning ability. In tasks that require high precision, such as complex language understanding, this may lead to a decline in performance [33].

The root cause of this issue is the insufficient GPU memory, which necessitates the use of a low-precision quantized model. A straightforward solution would be to employ a GPU with larger memory capacity and utilize the original 32-bit model parameters, which would resolve this problem.

Besides using a GPU with more memory, distributed training is another potential solution. Distributed training refers to the process of splitting the model and data across multiple GPUs or machines to parallelize the computation. This approach allows the model to handle larger datasets and more complex architectures without being constrained by the memory of a single GPU. By distributing both the data and computational load, distributed training can enable the use of full-precision models (32-bit) while reducing the overall training time and memory bottlenecks.

### 7.2 Choice of Pooling Operations

In Chapter 5.7, it was mentioned that this project employs pooling operations. While this operation can reduce the dimensionality of the embedding vectors, it may also result in the loss of some information from the original image feature vectors. Therefore, selecting an appropriate pooling operation based on the characteristics of the images is crucial.

There are two primary types of pooling: Max Pooling and Average Pooling. Average Pooling calculates the average value of the elements within the grid, creating a smoother feature representation. Average Pooling can maintain the overall structure, preserving global patterns and context while reducing noise, which is useful in tasks that require global information. However, it can blur and lose some important distinguishing features. In contrast, Max Pooling selects the maximum value from a small grid within the input feature map, focusing on the most prominent features in each region while ignoring less relevant information. This is particularly useful in tasks that require the identification of significant features, such as object detection [29].

Upon verification, the only pooling method provided by ConvnextV2, specifically the `pooler_output`, is global average pooling. However, since this project aims to extract key features from images, it is more appropriate to use Max Pooling to emphasize significant features.

### 7.3 Objective Setting Issue: Random Train-Test Split with Different Ratios

This project considers all previous research reports and combines the best scores achieved by each study in various text quality metrics to establish the objectives. In other words, the objectives are composed of results from multiple studies. However, setting the objectives in this way introduces an unavoidable issue: the IU X-ray dataset does not provide a predefined train-test split ratio. This leads to varying training and test set sizes in different studies. For instance, in the study by Xue and Huang et al. [34], they randomly used 3,031 data points for the training set and 300 data points for the test set. In the study by Xue and Xu et al. [35], they randomly used 2,525 data points for the training set and 250 data points for the test set. Moreover, even when the same ratio is applied for splitting the dataset, the data itself is still randomly assigned. For example, in the studies by Li et al. [14] and Li and Liang et al. [36], both split the data by patients into training, validation, and testing according to a predefined ratio. However, they explicitly stated that the data was randomly split.

The issue of randomly splitting the training and test sets with different ratios makes it difficult to fairly compare results with previous studies. If the dataset were large enough, this issue might be mitigated. However, compared to other chest X-ray datasets such as the MIMIC-CXR dataset, which has 370,000 images and 220,000 reports, and the PadChest dataset, which has 160,000 images and 100,000 reports, the IU X-ray dataset is significantly smaller, with only 7,000 images and 4,000 reports. This exacerbates the unfairness caused by the issue of train-test splits, making it harder to achieve the objectives set in this study.

## 8. Conclusion

The objective of this project is to create a medical image automatic reporting model that outperforms previous models in text quality metrics. To achieve this goal, a joint model was developed using the visual model Convnext V2, the language model LLaMA 3, and a projection model. Testing confirmed that this model is capable of generating medical reports. However, the evaluation of text quality metrics indicated that the model did not meet the established targets, showing average performance in BLEU and ROUGE metrics, while excelling in METEOR, with scores very close to the target.

In the research reflection section, it was suggested that the low precision of the quantized model and the inappropriate choice of pooling operations could be reasons for the suboptimal performance in text quality metrics. Although this project did not achieve its intended goals, it demonstrated that ConvNext V2 is a viable tool for medical feature extraction, and LLaMA 3 is effective for medical report generation. For future research, similar approaches can be adopted by combining novel CNN architectures with large language models, such as LLaMA 3.2. Additionally, optimization strategies can be applied to further improve performance, such as using auxiliary medical image tasks like classification and segmentation to pre-train or fine-tune ConvNext V2 models. Another potential direction is to implement Reinforcement Learning to design reward functions that guide the model's learning process.

Moreover, this project only evaluated the results using text quality metrics, lacking more comprehensive assessments such as medical accuracy and interpretability. Medical accuracy refers to the correctness and reliability of the medical information presented in the generated reports. Ensuring medical accuracy is crucial, as inaccuracies can lead to misdiagnoses, inappropriate treatments, or other detrimental outcomes for patients. On the other hand, interpretability pertains to the clarity and understandability of the generated reports for medical professionals and other stakeholders. It is vital for healthcare practitioners to easily comprehend the information presented in these reports to make informed decisions. High interpretability ensures that clinicians can trust and act upon the findings without misinterpretation, fostering effective communication among healthcare teams. Therefore, both medical accuracy and interpretability are important aspects that future research could explore to enhance the effectiveness and reliability of automated medical report generation systems.

## References

- [1] Radiology Template Reports. (2022, October 31). a Site for Radiology Templates. <https://www.radiologytemplates.com.au/home-page/>
- [2] Xiong,Z. (2015, April 30). *Wei lai shi nian hu chan ren li gong xu ping gu yan jiu ji hua* [ Research plan for assessment of supply and demand of obstetric nursing manpower in the next ten years ] . <https://www.mohw.gov.tw/dl-47187-0a053798-2038-4cd3-a3ab-b05e431d42dd.html>
- [3] Lin,C.E. (2023,October 12). *Fang she xu qiu fan bei ren li zeng bu dao san cheng gong hui :zheng ce zhi biao bu zhi ben* [ Radiation demand has doubled, and manpower has increased by less than 30%. Trade unions: Policies address the symptoms rather than the root cause ] . United News Network. <https://udn.com/news/story/7266/7500035><https://udn.com/news/story/7266/7500035>
- [4] National Federation of Medical Doctor Associations of the Republic of China. (2011,June.26).*Fang she zhen duan ke zhuan ke yi shi xun lian ke cheng ji zhun* [ Diagnostic Radiology Specialist Training Curriculum Benchmarks ] . <https://www.tma.tw/files/FilesDown/Cstandard/%E6%94%BE%E5%B0%84%E8%A8%BA%E6%96%B7%E7%A7%91.pdf>
- [5] Lin, P.W. ( 2023,June 05). *Xin guan fei yan qi jian yi shi fang she shi de jiao lu ,gong zuo ya li ,gong zuo man yi du xiang guan yan jiu* [ Research on anxiety, work stress and job satisfaction of medical radiologists during the COVID-19 period ] .National Digital Library of Theses and Dissertations in Taiwan. <https://hdl.handle.net/11296/mvqdmx>
- [6] Li,P.C. (2007). *Yi shi fang she shi zhi chang pi lao yu gong zuo man yi du xiang guan xing yan jiu* [ Study on the correlation between workplace fatigue and job satisfaction among medical radiologists ] . Huayi Online Library. <https://www.airitilibrary.com/Article/Detail?DocID=U0099->
- [7] Messina, P., Pino, P., Parra, D., Soto, A., Besa, C., Uribe, S., Andía, M., Tejos, C., Prieto, C., & Capurro, D. (2020, October 20). *A Survey on Deep Learning and Explainability for Automatic Report Generation from Medical Images*. arXiv.org. <https://arxiv.org/abs/2010.10563>
- [8] Zhang, Y., Wang, X., Xu, Z., Yu, Q., Yuille, A., & Xu, D. (2020). *When radiology report generation meets knowledge graph*. Proceedings of the AAAI Conference on Artificial Intelligence, 34(07), 12910–12917. <https://doi.org/10.1609/aaai.v34i07.6989>
- [9] ILSVRC2014 Results. (n.d.). <https://image-net.org/challenges/LSVRC/2014/results.php>
- [10] ILSVRC2015 Results. (n.d.). <https://image-net.org/challenges/LSVRC/2015/results>
- [11] Huang, G., Liu, Z., Laurens, V. D. M., & Weinberger, K. Q. (2016, August 25). *Densely connected convolutional networks*. arXiv.org. <https://arxiv.org/abs/1608.06993>
- [12] IEEE Computer Society. (2017,July 21). *Computer vision and pattern recognition*. [https://cvpr2017.thecvf.com/program/main\\_conference](https://cvpr2017.thecvf.com/program/main_conference)
- [13] Wei, C., Ren, S., Guo, K., Hu, H., & Liang, J. (2022, July 23). *High-Resolution SWIN transformer for automatic medical image segmentation*. arXiv.org. <https://arxiv.org/abs/2207.11553>
- [14] Li, C. Y., Liang, X., Hu, Z., & Xing, E. P. (2019, March 25). *Knowledge-driven encode, retrieve, paraphrase for medical image report generation*. arXiv.org. <https://arxiv.org/abs/1903.10122>

- [15] Woo, S., Debnath, S., Hu, R., Chen, X., Liu, Z., Kweon, I. S., & Xie, S. (2023, January 2). *Convnext V2: Co-designing and Scaling ConvNets with Masked Autoencoders*. arXiv.org. <https://arxiv.org/abs/2301.00808>
- [16] What are Large Language Models? - LLM AI Explained - AWS. (n.d.). Amazon Web Services, Inc. [https://aws.amazon.com/what-is/large-language-model/?nc1=h\\_ls](https://aws.amazon.com/what-is/large-language-model/?nc1=h_ls)
- [17] What is RNN? - Recurrent Neural Networks Explained - AWS. (n.d.). Amazon Web Services, Inc. [https://aws.amazon.com/what-is/recurrent-neural-network/?nc1=h\\_ls](https://aws.amazon.com/what-is/recurrent-neural-network/?nc1=h_ls)
- [18] Meta. (n.d.). llama3. <https://llama.meta.com/llama3/>
- [19] Llama3. (2022, September 15). The most capable openly available LLM to date by the meta AI team. Hugging Face. [https://huggingface.co/docs/transformers/main/en/model\\_doc/llama3](https://huggingface.co/docs/transformers/main/en/model_doc/llama3)
- [20] <https://github.com/unslothai/unsloth>
- [21] Demner-Fushman, D., Kohli, M., Rosenman, M., Shooshan, S., Rodriguez, L., Antani, S., Thoma, G., & McDonald, C. (2015, July 1). *Preparing a collection of radiology examinations for distribution and retrieval*. OXFORD UNIVERSITY PRESS. <https://scholarworks.iupui.edu/bitstream/handle/1805/13649/ocv080.pdf?sequence=1&isAllowed=y>
- [22] Chest X-rays (Indiana University). (2020, February 17). Kaggle. <https://www.kaggle.com/datasets/raddar/chest-xrays-indiana-university/data>
- [23] Openi. (2011, December 4). Open Access Biomedical Image Search Engine. National Library of Medicine. <http://openi.nlm.nih.gov/>
- [24] djin31. (2024, April 17). ChestXRay - Transformer. Kaggle.com; Kaggle. <https://www.kaggle.com/code/djin31/chestxray-transformer>
- [25] Wikipedia Contributors. (2019, January 26). BLEU. Wikipedia; Wikimedia Foundation. <https://en.wikipedia.org/wiki/BLEU>
- [26] facebookresearch. (2022). GitHub - facebookresearch/ConvNeXt-V2: Code release for ConvNeXt V2 model. GitHub. <https://github.com/facebookresearch/Convnext-V2>
- [27] facebook/convnextv2-tiny-1k-224 · Hugging Face. (2023, December 21). Huggingface.co. <https://huggingface.co/facebook/convnextv2-tiny-1k-224>
- [28] ConvNeXt V2. (2024). Huggingface.co. [https://huggingface.co/docs/transformers/model\\_doc/Convnextv2](https://huggingface.co/docs/transformers/model_doc/Convnextv2)
- [29] Wikipedia Contributors. (2024, October 10). Pooling layer. Wikipedia; Wikimedia Foundation. [https://en.wikipedia.org/wiki/Pooling\\_layer#Average\\_pooling](https://en.wikipedia.org/wiki/Pooling_layer#Average_pooling)
- [30] Tiya Vaj. (2024, January 24). Why we freeze some layers for transfer learning - Tiya Vaj - Medium. Medium. <https://vtiya.medium.com/why-we-freeze-some-layers-for-transfer-learning-f35d9f67f99c>
- [31] Wikipedia Contributors. (2019b, July 23). ROUGE (metric). Wikipedia; Wikimedia Foundation. [https://en.wikipedia.org/wiki/ROUGE\\_\(metric\)](https://en.wikipedia.org/wiki/ROUGE_(metric))
- [32] Wikipedia Contributors. (2019c, November 14). Meteoroid. Wikipedia; Wikimedia Foundation. <https://en.wikipedia.org/wiki/Meteoroid>
- [33] Huang, W., Zheng, X., Ma, X., Qin, H., Lv, C., Chen, H., Luo, J., Qi, X., Liu, X., & Magno, M. (2024, April 22). An Empirical Study of LLaMA3 Quantization: From LLMs to MLLMs. ArXiv.org. <https://arxiv.org/abs/2404.14047>
- [34] Xue, Y., & Huang, X. (2019, July 2). Improved Disease Classification in Chest X-Rays with Transferred Features from Report Generation. Information Processing in Medical Imaging.

<https://www.semanticscholar.org/paper/Improved-Disease-Classification-in-Chest-X-Rays-Xue-Huang/7f364afcbc3ae1cd6ea6546205aaa90c13cfd553>

- [35] Xue, Y., Xu, T., Long, L. R., Xue, Z., Antani, S., Thoma, G., & Huang, X. (2018, September 16). Multimodal Recurrent Model with Attention for Automated Radiology Report Generation. International Conference on Medical Image Computing and Computer-Assisted Intervention. <https://www.semanticscholar.org/paper/Multimodal-Recurrent-Model-with-Attention-for-Xue-Xu/a5173728c5e7f5f9c3d36a93232147a3fa19e54e>
- [36] Li, C. Y., Liang, X., Hu, Z., & Xing, E. P. (2018). Hybrid Retrieval-Generation Reinforced Agent for Medical Image Report Generation. ArXiv.org. <https://arxiv.org/abs/1805.08298>