



Medical Report Generation using Chest X Ray Images

Final Project Report (Team 6)

**for the
course**

**DATS 6312 ‘Natural
Language Processing’**

Fall 2024

| | |
|-------------------------------|-----------|
| INTRODUCTION | 1 |
| DESCRIPTION OF DATASET | 1 |
| MY CONTRIBUTIONS | 1 |
| RESULTS | 8 |
| CONCLUSION | 9 |
| FURTHER IMPROVEMENTS | 10 |
| REFERENCES | 10 |

INTRODUCTION

In modern healthcare, radiology plays a crucial role in diagnosing and managing a wide array of medical conditions. Chest X-rays, in particular, are among the most frequently used diagnostic tools for detecting abnormalities such as Pneumonia, Hernia and Cardiomegaly. The motivation behind this project is to create a tool that supports radiologists in their demanding roles by automating the generation of preliminary radiology reports. By leveraging advanced computer vision and large language models, the system aims to draft accurate and detailed reports from chest X-ray images, serving as a starting point for radiologists to review and refine. This collaboration between AI and radiologists enhances productivity, reduces delays, and minimizes the risk of errors stemming from workload fatigue.

DESCRIPTION OF DATASET

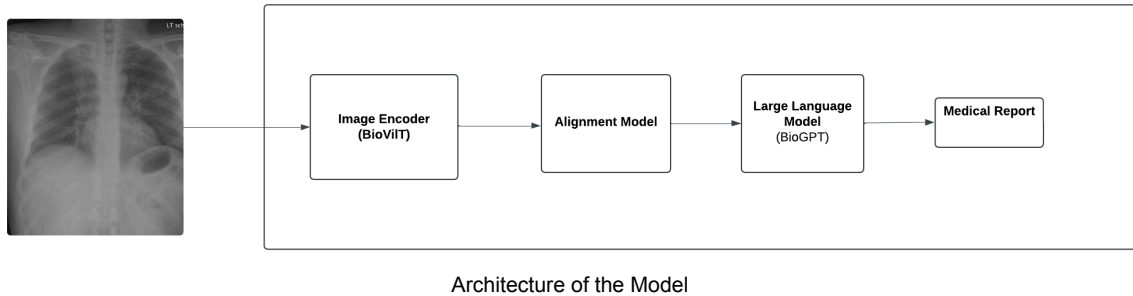
The MIMIC-CXR is a publicly available chest X-ray dataset for chest radiography research. It comprises 15,000 chest X-ray images in dicom format and their associated radiology reports in xml format. The dataset has the following key features:

- **Image File Path:** Location or link to the corresponding chest X-ray image.
- **Findings:** A textual description of abnormalities or observations made by the radiologist.
- **Impression:** A concise summary of the radiologist's primary conclusions.

The dataset has 14 labels corresponding to common chest X-ray pathologies. The pathology labels include Atelectasis, Cardiomegaly, Consolidation, Edema, Enlarged Cardiomediastinum, Fracture, Lung Lesion, Lung Opacity, Pleural Effusion, Pleural Other, Pneumonia, Pneumothorax, Support Devices, and No Finding.

MY CONTRIBUTIONS

I was responsible for multiple critical components of this project, from data preprocessing to training models and designing the report generator. I was also responsible for creating the biovil_t module. Following image shows the architecture of the BioViIT with BioGPT model



My work was divided into the following modules:

Detailed Description of Work

The `biovil_t` module, sourced from the Radiology GitHub repository, provided a robust image encoder leveraging pre-trained weights from the BioViL model. Key takeaways included:

- Gained practical knowledge on modifying ResNet architectures for specific use cases.
- Understood the concept of global and patch embeddings and their significance in representing image features at different granularities.
- 1. **Data Preprocessing (BioGPT_Base_data_processing.py)**
 - **Purpose:** To prepare a clean, augmented dataset of chest X-rays and associated findings for training the models.

Implementation Details:

- Dataset Class (ChestXrayDataset):
 - Created a custom PyTorch Dataset class to handle data loading and preprocessing.

```

class ChestXrayDataset(Dataset):
    def __init__(self, data_frame: pd.DataFrame, transform:
Optional[A.Compose] = None, is_training: bool = True):
        self.data = data_frame
        self.transform = transform or
self._get_default_transforms(is_training)
        self._preprocess_dataset()
  
```

- Added preprocessing steps to:
 - Validate image paths and ensure the images exist.
 - Remove rows with missing or invalid findings (reports).
 - Check that images are readable by trying to load each file using OpenCV.

- Implemented image augmentation using the Albumentations library for better generalization:
 - Random flips, rotations, noise addition, and brightness/contrast adjustments were included for the training dataset.
 - Normalized images to standardize pixel values.

```
def _get_default_transforms(self, is_training: bool) -> A.Compose:
    if is_training:
        return A.Compose([
            A.Resize(224, 224),
            A.HorizontalFlip(p=0.5),
            A.RandomRotate90(p=0.5),
            A.OneOf([A.GaussNoise(p=1), A.GaussianBlur(p=1)], p=0.3),
            A.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]),
            ToTensorV2()
        ])
```

- Data Loaders:
 - Split the dataset into training and validation sets (85% training, 15% validation) using train_test_split from scikit-learn.
 - Created PyTorch DataLoader objects to enable efficient batch processing with features like shuffling and multiprocessing.

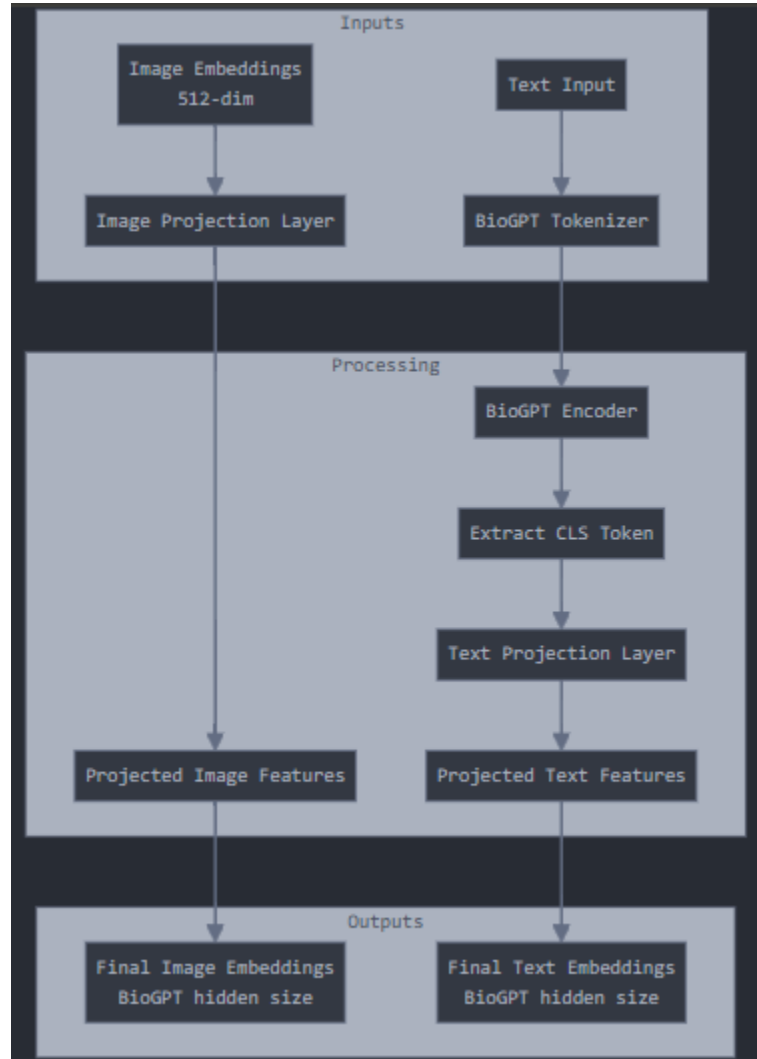
What I Learned:

Data augmentation helped me understand how synthetically generating variations of images (e.g., through flips, rotations, and noise) can improve the model's robustness and generalizability. This was especially important in medical imaging, where the dataset might be limited.

2. Image-Text Alignment (BioGPT_Base_alignment_model.py)

- Purpose: To align chest X-ray image features with textual representations in a shared representation space.

Implementation Details:



Alignment Model Flow

- **Model Design:**
 - Leveraged BioGPT's AutoModel and tokenizer from Hugging Face for text embeddings.
 - Designed a projection layer to map image embeddings (dimension: 512) to BioGPT's hidden size for alignment.

```

class ImageTextAlignmentModel(nn.Module):
    def __init__(self, image_embedding_dim: int = 512):
        super().__init__()
        self.image_projection = nn.Linear(image_embedding_dim, 768)

```

```
self.text_projection = nn.Linear(768, 768)
```

- Forward Pass:
 - The model takes two inputs:
 1. Image embeddings: High-dimensional features extracted from chest X-ray images.
 2. Text descriptions: Processed through BioGPT's encoder to extract text embeddings.
 - Projected both embeddings into a common latent space using linear layers

```
class ImageTextAlignmentModel(nn.Module):
    def __init__(self, image_embedding_dim: int = 512):
        super().__init__()
        self.image_projection = nn.Linear(image_embedding_dim, 768)
        self.text_projection = nn.Linear(768, 768)
```

- Returned the aligned image and text embeddings for subsequent tasks

What I Learned:

- Alignment was a completely new concept to me, especially the idea of projecting image and text features into a shared latent space. Designing projection layers deepened my understanding of how to manipulate embedding dimensions and distributions. I learned how these layers act as bridges, transforming high-dimensional image embeddings into a format compatible with text embeddings. Contrastive Loss is also something I got to learn through this project.

3. Medical Report Generation (BioGPT_Base_report_generator.py)

- Purpose: To generate diagnostic reports based on aligned embeddings, using BioGPT as the language model.

Implementation Details:

- Base Model:
 - Utilized the BioGPT language model from Hugging Face's Transformers library.
 - Enabled gradient checkpointing for efficient memory usage during training.
- PEFT Integration:
 - Incorporated LoRA (Low-Rank Adaptation) for fine-tuning BioGPT while keeping the pre-trained weights intact.
 - Used the following parameters for LoRA:
 - Rank (r): 16 (dimensionality of the adaptation matrices)
 - Alpha (lora_alpha): 32 (scaling factor for LoRA)

- Dropout (lora_dropout): 0.1 (prevents overfitting)
 - Targeted specific layers (e.g., query/key/value projections) for LoRA adjustments to focus on medical domain-specific improvements.
- Report Generation:
 - Designed a custom method to generate text reports from embeddings:
 - Mapped aligned image embeddings into BioGPT's hidden size using a linear projection layer.
 - Concatenated the projected embeddings with token embeddings for decoding.

```
def generate_report(self, input_embeddings: torch.Tensor, max_length: int = 150):
    projected_embeddings =
self.input_projection(input_embeddings).unsqueeze(1)
    outputs = self.model.generate(inputs_embeds=projected_embeddings,
max_length=max_length)
    return self.tokenizer.batch_decode(outputs, skip_special_tokens=True)
```

- Generated textual reports using beam search with controlled parameters (e.g., top-k sampling, temperature).
 - max_length=150
 - temperature=0.8
 - top_k=50
 - top_p=0.85

What I Learned:

- Using PEFT introduced me to the different efficient fine-tuning techniques that optimize a few parameters while freezing the majority of the pre-trained model's weights. I understood how LoRA was resource-efficient and how the appropriate parameters can be selected for LoRA.
- Tuning parameters like temperature, top-k, and top-p sampling allowed me to explore how they impact the diversity and relevance of generated reports. This was a hands-on way to balance creativity and accuracy in text generation.

4. Model Training (BioGPT_Base_train.py)

- Purpose: To train and validate the complete system, ensuring optimal alignment and report generation.

Implementation Details:

- **Model Initialization:**
 - Loaded the BioViL image encoder, alignment model, and report generator.
 - Transferred all models to GPU for faster training.
- **Optimizers:**
 - Used AdamW for weight updates:
 1. Separate learning rates for alignment model and report generator layers.
 - **Alignment Optimizer:** learning rate - $2e-5$.
 - **Generator Optimizer:** learning rates for LoRA parameters ($2e-5$) and the input projection layer ($1e-4$).
 2. Controlled gradients to focus on specific modules during training.
- **Loss Functions:**
 - Alignment Phase: Optimized using cosine embedding loss to align image and text embeddings.
 - Generation Phase: Employed a cross entropy for training the report generator.
- **Training Loop:**
 - Iterated through training data for 30 epochs.
 - Two-step process:
 1. Alignment: Updated alignment model weights using image and text embeddings.
 2. Generation: Updated report generator weights using aligned embeddings and ground-truth reports.

```
for epoch in range(num_epochs):
    projected_image, projected_text = alignment_model(image_embeddings,
    impressions)
    align_loss = contrastive_loss(projected_image, projected_text, labels)
    gen_loss, logits = report_generator(projected_image, target_ids)
```

- **Validation:**
 - Evaluated the model on a separate validation set after each epoch.
 - Calculated metrics, including alignment loss, generation loss, and ROUGE-L scores, to track performance.

```
def compute_metrics(references, predictions):
    scorer = rouge_scorer.RougeScorer(['rougeL'], use_stemmer=True)
    return sum([scorer.score(r, p)['rougeL'].fmeasure for r, p in
    zip(references, predictions)]) / len(references)
```

- Saved checkpoints and the best model based on validation performance.

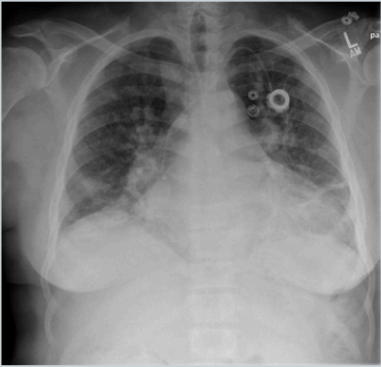
What I Learned:

- Integrating ROUGE-L as an evaluation metric emphasized the value of automated metrics in assessing generative models. I studied this metric and how it is calculated.
- Comparing automated metrics with qualitative feedback (e.g., manual evaluation of report accuracy) showed me the importance of combining multiple evaluation techniques for a holistic assessment.
- Lines copied from internet (BioViLT) = 842
- My own lines = 650
- Total lines = 842 + 650 = 1492

Using the formula: (Original copied lines - Modified lines) / Total lines × 100
 = (842) / (842 + 650) × 100 = 0.5643 × 100 = 56.43%

RESULTS

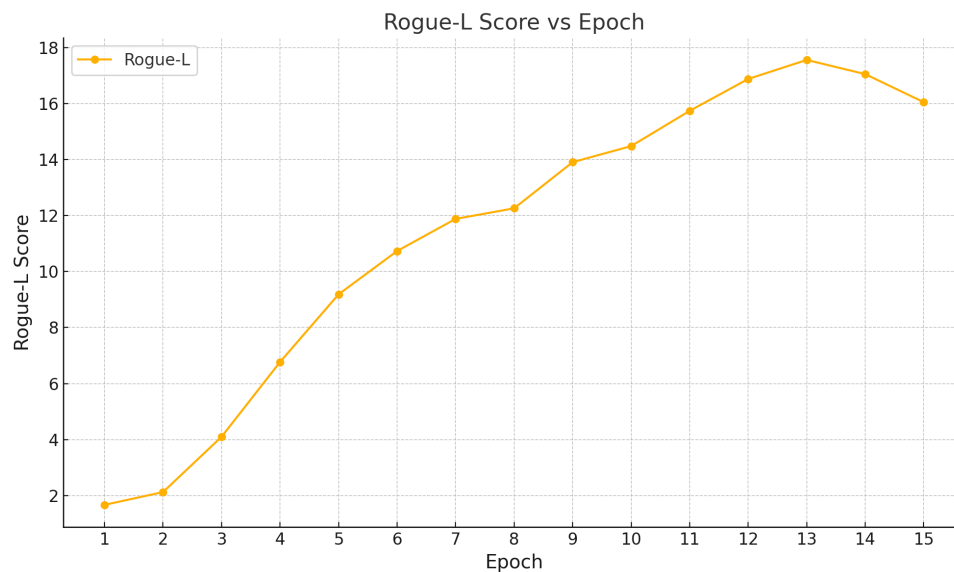
The results section presents the performance of the developed system in generating radiology reports from chest X-ray images. Below image shows a comparison between the ground truth and the report generated by the model for one of the chest x-ray.

| Chest X Ray Image | Ground Truth | Prediction |
|---|---|--|
|  | <p>Report:</p> <p>Bilateral patchy pulmonary opacities noted. Interval improvement in left base consolidative opacity. Pulmonary vascular congestion again noted. Stable enlarged cardio mediastinal silhouette. Stable left XXXX. No evidence of pneumothorax. No large pleural effusions.</p> | <p>Report:</p> <p>Sternotomy and repair of a right-sided chest wall pericardiac loop in association with congestive heart failure is unchanged, but on the opposite side of this. The lungs are well expanded without evidence for acute or chronic consolidation. No pleural or pneumothorax found. No new areas of focal airspace disease. There has been interval resolution of left basilar atelectasis.</p> |

Example of predicted findings versus ground truth.

The model demonstrated varying success in predicting findings and impressions from chest X-ray images. Instances of bold text indicate cases where the model failed to predict or align the pathology accurately with the ground truth (basilar atelectasis was predicted instead of consolidative opacity), showing gaps in its understanding or representation. Conversely, areas

highlighted in yellow and green denote successful predictions, reflecting the model's ability to capture and align features effectively between the images and the textual descriptions.



- The above plot illustrates the progression of ROUGE-L scores (y-axis) over epochs (x-axis).
- ROUGE-L is a metric used to evaluate the quality of text generation models by measuring the longest common subsequences (LCS) between the generated and ground-truth reports.

Performance Trend:

- Epochs 1 to 8: There is a significant upward trend in ROUGE-L scores, indicating that the model's performance in generating high-quality and accurate medical reports improves steadily with training. This aligns with the phase where the model learns to align the embeddings and generate meaningful reports.
- Epochs 8 to 13: The growth slows down, showing a saturation in performance improvement. This suggests that the model reaches its peak performance during this phase.
- Epochs 14 to 15: There is a slight decline in ROUGE-L scores, which may indicate the model starts losing its generalization ability and adapts too closely to the training data.

CONCLUSION

The model successfully demonstrated the feasibility of generating preliminary medical reports from chest X-ray images using advanced computer vision and natural language processing techniques. By leveraging the MIMIC-CXR dataset, the BioViL image encoder, and the BioGPT

language model, the system achieved promising results in aligning image features with textual descriptions and generating meaningful radiology reports. The integration of techniques such as image-text alignment and LoRA-based fine-tuning enabled the model to achieve steady performance improvements, as reflected in the ROUGE-L scores.

The performance of the model can be further enhanced by focusing on several areas for future improvement, which could help refine its accuracy and robustness.

FURTHER IMPROVEMENTS

1. More Training Data : Incorporating additional data from the MIMIC-CXR Dataset can help improve model generalization and performance.
2. Multimodal Fusion Techniques: Experimenting with advanced multimodal architectures, such as BLIP could improve image-text alignment and overall report generation quality.
3. Human-in-the-Loop Evaluation: Incorporating feedback from radiologists to iteratively fine-tune the model and evaluate the clinical relevance and accuracy of the generated reports.

REFERENCES

1. S. Bannur et al., "Learning to Exploit Temporal Structure for Biomedical Vision-Language Processing," in 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 2023, pp. 15016-15027, doi: 10.1109/CVPR52729.2023.01442.
2. <https://github.com/microsoft/BioGPT>
3. Pellegrini, C., Özsoy, E., Busam, B., Navab, N., and Keicher, M. (2023). Radialog: a large vision-language model for radiology report generation and conversational assistance. *arXiv Preprint arXiv:2311.18681*. doi: 10.48550/arXiv.2311.18681