

## Vision Language Model Fall 2024-25

Prof Matthew Hamilton

Monish Kamwal

Chowdhury Nafis Saleh

Pedram Abdolahi Darestani

## Introduction

Vision Language Models (VLMs) are a branch of Machine Learning Model that utilize both Computer Vision (CV) Models and Natural Language Processing (NLP) Models to use visual data (like images) and textual information to analyze data and make predictions by correlating the information. Here at VAC Lab, we aim to use VLMs to detect breast cancer and provide a human-readable description of the features the model based its prediction on. The VLM is trained on databases composed of x-ray images of breast with front and lateral views along with radiology reports. The VLM generates a descriptive textual response to an image input.

## Articles, Research Papers and Models Explored

- “Vision-Language Models for Vision Tasks: A Survey” [link](#)
- “Exploring the Frontier of Vision-Language Models: A Survey of Current Methodologies and Future Directions” [link](#)
- BioMedCLIP [link](#)
- BioVil-T [link](#)
- “An Introduction to Vision-Language Modeling” [link](#)
- “Vision-Language Models for Medical Report Generation and Visual Question Answering: A Review” [link](#)
- “Vision-Language Modelling for Radiological Imaging and Reports in the Low Data Regime” [link](#)
- SERPENT-VLM [link](#)
- “BiomedGPT: A generalist vision-language foundation model for diverse biomedical tasks” [link](#)
- Visual Med-Alpaca [link](#)
- LLaVa-Med [link](#)
- Med-Flamingo [link](#)
- Llama 3.2 (not pretrained on medical data.) [link](#)
- BioMedVLP
- “seeMore- Implement a Vision Language Model from Scratch” [link](#)
- DeepSeek-VL [link](#)
- Bio-VIL [link](#)

- “RE-tune: Incremental Fine Tuning of Biomedical Vision-Language Models for Multi-label Chest X-ray Classification” [link](#)
- PubMedBert [link](#)
- CXR-BERT general [link](#)
- CXR-BERT specialized [link](#)
- Qwen-2VL [link](#)
- “Cross-modal Prototype Driven Network for Radiology Report Generation” [link](#)

## Databases explored

- BIMCV PADCHEST [link](#)
- Vindr Mammo [link](#)
- Indiana University Chest X-rays [link](#)
- Mendeley Data - Labeled Optical Coherence Tomography (OCT) and Chest X-Ray Images for Classification [link](#)
- Kaggle Chest X ray images - uses Mendeley Data mentioned above [link](#)
- MIMIC-CXR [link](#)
- PubMed [link](#)
- MIMIC-III [link](#)

## Model selected

### BioViL-T [link](#)

We have selected the the model for the following reasons:

- **BioViL-T** is a radiology-focused model trained using temporal multi-modal pre-training, distinguishing it from its predecessor, **BioViL**.
- Uses the same training databases as **BioViL**, but leverages temporal structure between data points, improving downstream performance.
- Displays significant improvement in embedding temporal information present in both image and text modalities, as well as in the joint representation space.
- Adaptable to single- and multi-image downstream applications such as:
  1. Natural Language Inference
  2. Phrase Grounding
  3. Image/Text Classification
  4. Language Decoding
- The **BERT** language model is trained in two stages:
  1. Pretrain **CXR-BERT-general** using Masked Language Modeling (MLM) on **PubMed** abstracts and clinical notes from **MIMIC-III** and **MIMIC-CXR**.
  2. Continual pre-training of **BioViL-T** from **CXR-BERT-general** using a multi-modal pre-training procedure involving radiology reports and chest X-ray sequences.
- Latent representation of the [CLS] token is used to align text and image embeddings.

# Training Data for BioVIL-T

This model builds upon existing publicly-available databases:

- [PubMed](#)
- [MIMIC-III](#)
- [MIMIC-CXR](#)

These databases reflect a broad variety of sources ranging from biomedical abstracts to intensive care unit notes to chest X-ray radiology notes. The radiology notes are accompanied with their associated chest x-ray DICOM images in the MIMIC-CXR database.

## Progress and next steps

We have spent a bulk of our time researching the various implementations of VLM and determining our development goals in a manner that makes further research easier and makes the model more scalable and extensible. A detailed description can be found at the end of this document under “Further Reading” by clicking [here](#).

### Next steps

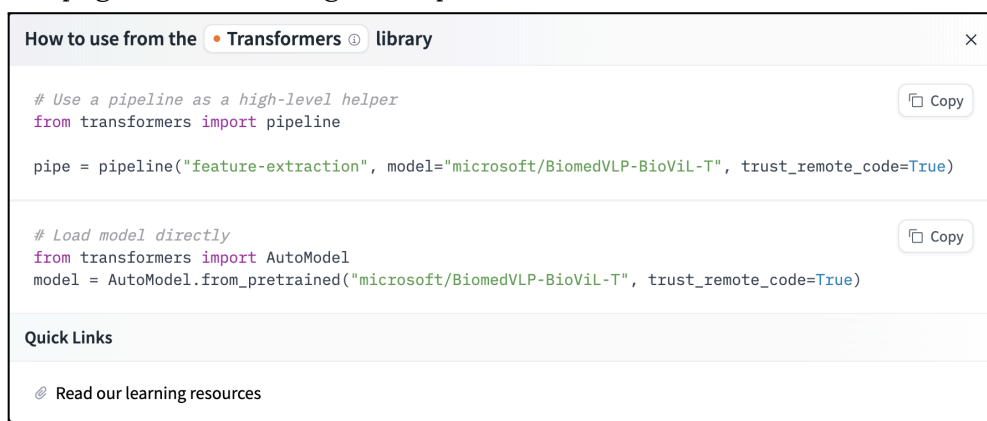
Here is a step-by-step guide on how to proceed with the project in the most efficient manner we could find avoiding any pitfalls to save time and resources.

1. Apply for CAIR credentials [here](#)
2. Learn how to use the CAIR system by following the guide
3. Search for suitable databases that have both X-Ray Images along with Radiology reports for breast cancer detection
4. Set-up **BioVIL-T** as mentioned in “Setting up BioVIL-T”

### Setting up BioVIL-T

Do NOT use the code under “[How to use](#)” section in the “[“BiomedVLP-BioVIL-T”](#)” article since the implementation mentioned there is misleading and only provides code to test the model by using textual prompts

1. Define the model - Steps to define the model can be found under the “Use this model” button on [this](#) page. Here’s an image example



```
How to use from the Transformers library

# Use a pipeline as a high-level helper
from transformers import pipeline

pipe = pipeline("feature-extraction", model="microsoft/BiomedVLP-BioViL-T", trust_remote_code=True)

# Load model directly
from transformers import AutoModel
model = AutoModel.from_pretrained("microsoft/BiomedVLP-BioViL-T", trust_remote_code=True)
```

Quick Links

Read our learning resources

2. Data Preprocessing - Follow guidelines of your chosen databases to load the data and split the data into Testing and Training sets
3. Train and Test the model on the Training and Testing data
4. Optional: Try to fine-tune the model based on the databases used

## Challenges

However, we encountered meaningful challenges during our research as follows:

- Need for large amounts of high-quality medical training data
- Ensuring clinical accuracy and safety
- Meeting regulatory requirements for medical AI
- Maintaining consistency with standard reporting practices
- Dealing with medical terminology and specific formatting requirements

## Alternate approach to consider

If you wish to steer clear of pre-trained models, you could take a different route to achieving the project's goals. This process would involve taking a model like LLaVA-1.5, Qwen-2-VL, or Llama and specializing it for medical imaging through careful fine-tuning. You'd start with the model's existing visual and language capabilities as a foundation, then gradually teach it to understand medical images and generate appropriate reports through exposure to high-quality medical image-report pairs.

The training would happen in stages:

1. First, helping the model understand medical imaging concepts and terminology through exposure to medical knowledge
2. Then teaching it to recognize specific findings in medical images
3. Finally, training it to generate structured, accurate medical reports that follow proper conventions

One such alternative will be discussed at length in the following section.

## Possible Alternative:

LLaVA-1.5 [link](#)

### Dataset Format:

As shown [here](#), in order to fine-tune the model on any custom dataset, we would have to format the data accordingly. For the MIMIC-CXR dataset, it would look something like this:

```
[
  {
    "id": "(mimic-cxr-sample-photoid)",
    "image": "(Path to image)/(image).dcm",
    "conversations": [
      {
        "from": "human",
```

```

    "value": "<image>\nDescribe the findings in this chest X-ray and write a report."
  },
  {
    "from": "gpt",
    "value": (Corresponding medical report)
  }
]
}
]

```

This is for one sample, we would have to do the same for the entire dataset.

### **Training:**

The following is the script for distributed training using Deepspeed, as shown in the model's [Github page](#):

```

#!/bin/bash

deepspeed llava/train/train_mem.py \
  --deepspeed ./scripts/zero3.json \
  --model_name_or_path liuhaotian/llava-v1.5-13b \
  --version v1 \
  --data_path ./playground/data/mimic_cxr_train.json \
  --image_folder ./playground/data/mimic-cxr-images \
  --vision_tower openai/clip-vit-large-patch14-336 \
  --mm_projector_type mlp2x_gelu \
  --mm_vision_select_layer -2 \
  --mm_use_im_start_end False \
  --mm_use_im_patch_token False \
  --image_aspect_ratio pad \
  --group_by_modality_length True \
  --bf16 True \
  --output_dir ./checkpoints/llava-v1.5-13b-mimic \
  --num_train_epochs 3 \
  --per_device_train_batch_size 8 \
  --per_device_eval_batch_size 4 \
  --gradient_accumulation_steps 2 \
  --evaluation_strategy "steps" \
  --eval_steps 500 \
  --save_strategy "steps" \
  --save_steps 1000 \
  --save_total_limit 3 \
  --learning_rate 1e-5 \
  --weight_decay 0.01 \
  --warmup_ratio 0.03 \
  --lr_scheduler_type "cosine" \
  --logging_steps 10 \
  --tf32 True \
  --model_max_length 2048 \
  --gradient_checkpointing True \

```

```
--dataloader_num_workers 4 \  
--lazy_preprocess True \  
--report_to wandb
```

The DeepSpeed configuration will probably look something like this:

```
{  
  "zero_optimization": {  
    "stage": 3,  
    "offload_optimizer": {  
      "device": "cpu",  
      "pin_memory": true  
    },  
    "offload_param": {  
      "device": "cpu",  
      "pin_memory": true  
    },  
    "overlap_comm": true,  
    "contiguous_gradients": true,  
    "reduce_bucket_size": 5e7,  
    "stage3_prefetch_bucket_size": 5e7,  
    "stage3_param_persistence_threshold": 1e5  
  },  
  "bf16": {  
    "enabled": true  
  },  
  "gradient_clipping": 1.0,  
  "train_batch_size": 32,  
  "train_micro_batch_size_per_gpu": 8,  
  "wall_clock_breakdown": false  
}
```

Please note that this might not be fully correct. You'll have to test it and play around a little to find the proper configuration.

To proceed with the training, you'll have to take the following steps:

- 1) Clone the LLaVA-1.5 repository

```
git clone https://github.com/haotian-liu/LLaVA.git
```

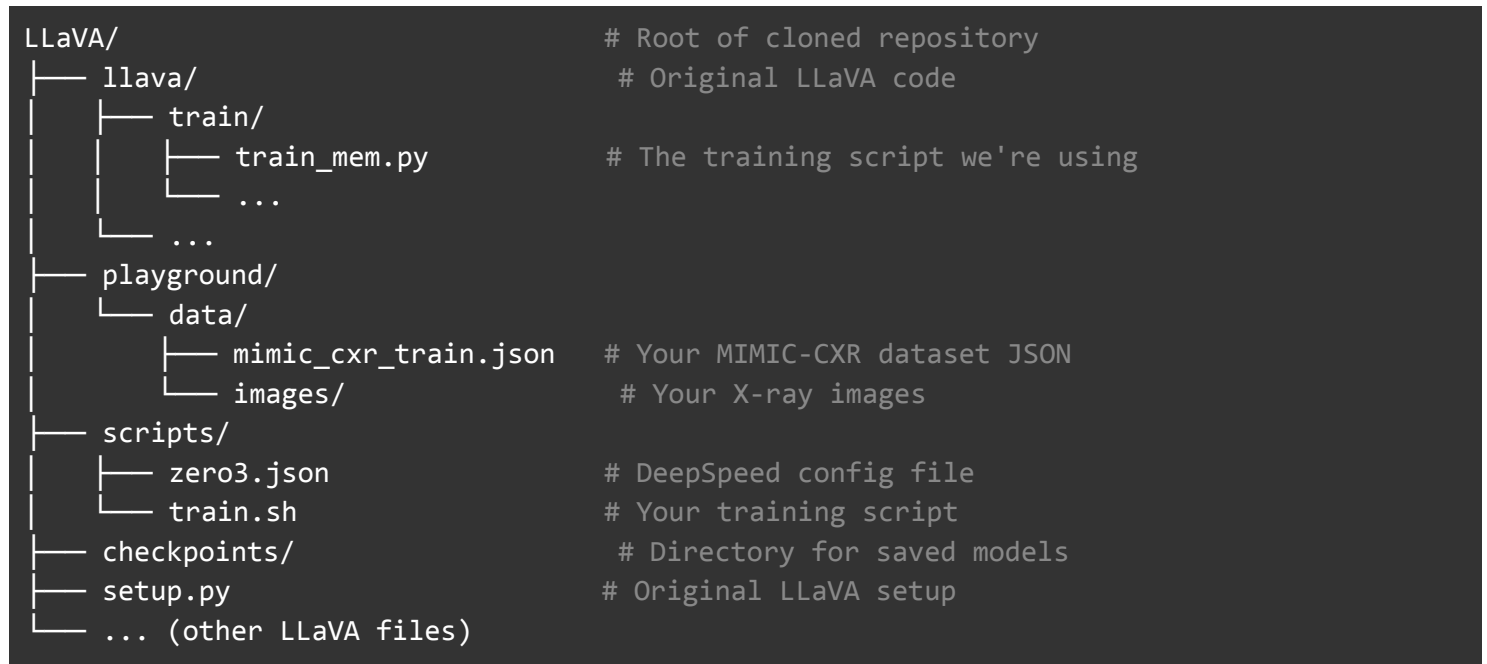
- 2) Navigate to the root directory of the cloned repository.
- 3) Create the recommended directory structure inside this cloned repository:

```
mkdir -p playground/data  
mkdir -p scripts  
mkdir -p checkpoints
```

- 4) Put your MIMIC-CXR or IU-XRay JSON file in the `playground/data/` directory
- 5) Put your X-ray images in the `playground/data/images/` directory

- 6) Save the training script and the DeepSpeed configuration in the `scripts/` directory

The directory structure will look something like this:



After everything has been properly set up, you can run the training script from the terminal to begin the training process.

### Reasons for choosing this model:

1. Model Architecture and Capabilities:
  - Built on a strong foundation (Llama 2) with proven language generation abilities
  - Has effective multi-modal understanding across diverse image types
2. Size and Efficiency:
  - More manageable to fine-tune compared to larger models
  - Can run on more modest hardware setups, making it practical for research and development
3. Open Source Nature:
  - Full access to model architecture and weights
  - Ability to modify and adapt the model extensively
  - Strong community support and documentation
  - Active development and improvements

However, there are some considerations to keep in mind:

- May require substantial fine-tuning to match domain-specific medical language
- The visual encoder might need adaptation for medical imaging specifics

## Further Reading

We initially started by looking at the VinDr-Mammo dataset, which is a large-scale mammography dataset containing over 20,000 mammogram images from more than 5,000 patients. Each image is labeled with BI-RADS assessments and includes bounding box annotations for abnormalities when present. However, we didn't proceed with it simply because it didn't fit the goal of the research. The research was mostly concerned with report generation from medical images, but even though this dataset had a good number of high-quality annotated Mammograph images, they're better suited for binary classification instead of report generation. Not to mention, training a VLM with this dataset would be even more challenging since the model would have to learn proper medical terminology and standardized reporting formats like BI-RADS, making the extraction of meaningful information incredibly difficult.

Then we moved on to the other datasets, and we explored all of them simultaneously. PADCHEST, which is a large-scale chest X-ray dataset containing around 160,000 images from 67,000 patients, wasn't suitable for our research for two main reasons. Firstly, the reports were written in Spanish, which would add some complications to the language processing aspect of the project. Secondly, the quality of the data is somewhat questionable as only 27% of the reports were annotated by trained experts, the rest were annotated by a supervised RNN model with attention mechanisms.

For our research, we only considered the MIMIC-CXR and IU-XRay datasets. That's because both these datasets have already been used for training VLMs focused on report generation from medical images. MIMIC-CXR has been used to train Microsoft's BioViL-T model, which is a domain-specific vision-language model designed to analyze chest X-rays (CXRs) and radiology reports. IU-XRay, on the other hand, has been used to train a Cross-modal PROtotype driven NETwork (XPRONET) for generating radiology reports. Besides, both these datasets have reports in English and follow standardized U.S. radiology reporting practices, which would make interpreting the information a little easier for the model. Another good thing about these datasets is that they have reports from multiple experts for each sample, which provides variety in reporting styles while maintaining standards.