Intelligent Radiologist Assistant (Lungs Tumor segmentation)

Detailed Project Report

Author: Tanjina Proma

Date: 1/04/2024

Table of Contents

Abstract

The accurate classification of lung tumors from chest CT scans plays a pivotal role in early diagnosis and treatment planning for patients with lung cancer. This study presents a comprehensive framework for Lung Tumor Classification from Chest CT Scans using the VGG16 Model implemented in TensorFlow, Python, Docker, MLflow, DagsHub, and DVC. The system leverages state-of-the-art deep learning techniques to automate the classification process, enhancing efficiency and accuracy in clinical practice.

The workflow begins with data acquisition and preprocessing, ensuring standardization and augmentation of chest CT scan datasets. The VGG16 model is then fine-tuned on the preprocessed data using transfer learning, facilitated by TensorFlow and MLflow for experiment tracking and hyperparameter optimization. Evaluation metrics such as accuracy, sensitivity, and specificity validate the model's performance.

Deployment is achieved through Docker containers, ensuring portability and reproducibility across different environments. MLflow manages model serving, allowing seamless integration with existing healthcare systems. Continuous integration and deployment pipelines, orchestrated by DagsHub, automate testing and deployment processes, enhancing efficiency and reliability.

This comprehensive system architecture provides a scalable and efficient solution for lung tumor classification, empowering healthcare professionals with a reliable tool for early detection and personalized treatment planning in the fight against lung cancer.

1 Introduction

Our system aims to automate Lung Tumor Classification from Chest CT Scans using the VGG16 Model in TensorFlow, Python, Docker, MLflow, DagsHub, and DVC. Leveraging deep learning, we fine-tune the VGG16 model on annotated CT scans, achieving accurate classification. MLflow tracks experiments, while DVC ensures data versioning. Docker containers facilitate seamless deployment. Integrated with DagsHub, our CI/CD pipelines automate testing and deployment, ensuring reliability and reproducibility. This comprehensive architecture empowers healthcare professionals with efficient and scalable tools for timely lung cancer diagnosis and treatment planning.

1.1 Objective

The objective is to develop an automated system for lung tumor classification from chest CT scans using the VGG16 Model in TensorFlow, Python, Docker, MLflow, DagsHub, and DVC. This system aims to improve the accuracy and efficiency of lung cancer diagnosis by leveraging deep learning techniques. By integrating Docker for containerization, MLflow for experiment tracking, DVC for data versioning, and DagsHub for collaboration, the system ensures reproducibility, scalability, and collaboration throughout the machine learning pipeline, ultimately benefiting patients and healthcare providers.

1.2 Benefits

Lung Tumor Classification from Chest CT Scans offers numerous benefits. Firstly, the VGG16 model, fine-tuned with TensorFlow, ensures accurate classification of lung tumors, aiding in early diagnosis and treatment planning. Docker containerization enables seamless deployment across diverse environments, ensuring consistency and scalability. MLflow and DVC streamline the machine learning lifecycle, facilitating efficient experimentation, versioning, and reproducibility of models and datasets. Additionally, DagsHub fosters collaboration and transparency through version-controlled project management. Overall, this integrated approach enhances the reliability, efficiency, and accessibility of lung tumor classification, ultimately improving patient outcomes and healthcare delivery.

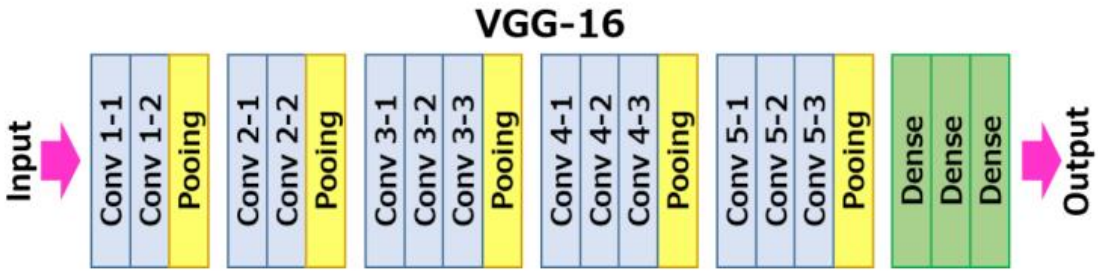
2. Literature Review

2.1 Image Classification Techniques

Traditional techniques for lung tumor classification from chest CT scans primarily involve manual interpretation by radiologists and clinicians, aided by computer-aided diagnosis (CAD) systems that utilize handcrafted features and rule-based algorithms. Radiologists analyze CT images to identify suspicious lesions based on visual characteristics such as size, shape, density, and location. They may use standardized criteria such as the Fleischner Society guidelines to classify nodules into categories like solid, part-solid, and ground-glass opacities, which provide indications of malignancy. CAD systems assist radiologists by extracting quantitative features from CT images, such as nodule size, shape irregularity, and texture patterns, and applying predefined rules or machine learning algorithms to classify nodules as benign or malignant. Common traditional techniques include thresholding, morphological operations, and rule-based decision-making. For example, threshold-based segmentation methods delineate tumor boundaries based on intensity thresholds, while morphological operations like erosion and dilation refine the segmentation results. However, traditional techniques often rely on manual input and suffer from limitations in capturing subtle image features and variability in tumor characteristics. As a result, they may lack the accuracy and efficiency required for reliable lung tumor classification in clinical practice, especially for complex cases or early-stage tumors.

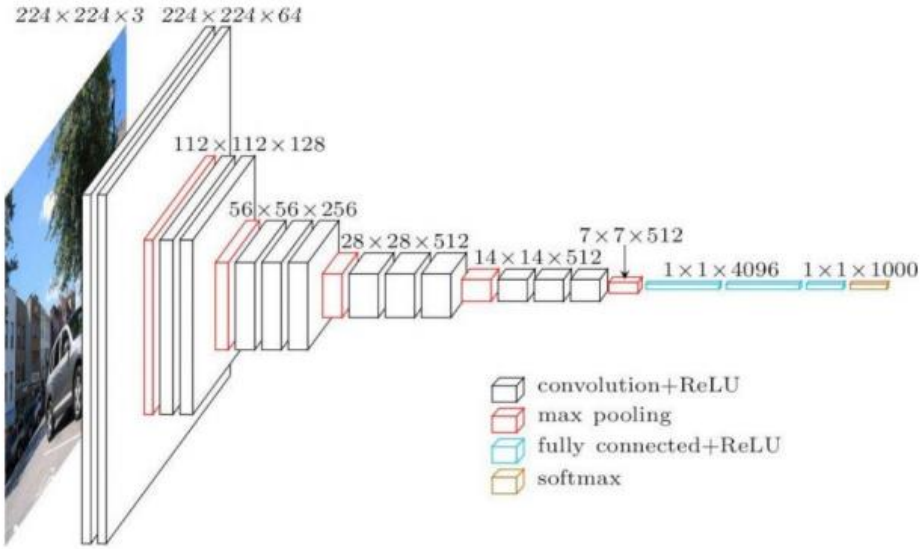
2.2 VGG16 Model Overview

The VGG16 model is a deep convolutional neural network (CNN) architecture that has gained significant popularity and recognition in the field of computer vision. Developed by the Visual Geometry Group (VGG) at the University of Oxford, VGG16 is renowned for its simplicity and effectiveness in image classification tasks.



The Architecture

The architecture depicted below is VGG16.



The architecture of the VGG16 model consists of 16 convolutional layers, followed by three fully connected layers and an output layer. Each convolutional layer is equipped with 3x3 filters and uses a rectified linear unit (ReLU) activation function to introduce non-linearity. The network architecture is characterized by its deep stacking of convolutional layers, resulting in a highly expressive feature representation hierarchy.

One of the key strengths of the VGG16 model lies in its uniformity and simplicity. The consistent use of 3x3 convolutional filters and max-pooling layers (2x2) throughout the network contributes to its ease of understanding and implementation. Additionally, the VGG16 architecture's modular design facilitates experimentation and customization, allowing researchers and practitioners to adapt the model to different tasks and datasets easily.

Despite its simplicity, the VGG16 model demonstrates remarkable performance in image classification tasks. Pre-trained versions of the VGG16 model, trained on large-scale image datasets like ImageNet, have been made available, enabling transfer learning for various computer vision applications. By fine-tuning the pre-trained VGG16 model on specific datasets,

researchers can leverage the learned feature representations to achieve competitive performance with reduced computational resources and data requirements.

However, the VGG16 model also has some limitations. Its deep architecture results in a large number of parameters, leading to increased computational complexity and memory requirements during training and inference. Additionally, the spatial information captured by the VGG16 model may be limited due to the extensive use of max-pooling layers, potentially impacting its performance in tasks requiring precise localization or spatial understanding.

In summary, the VGG16 model stands as a seminal contribution to the field of deep learning, offering a simple yet powerful architecture for image classification tasks. Its modular design, uniform structure, and strong performance have established it as a benchmark model in computer vision research and applications.

3. Dataset

Link: https://www.kaggle.com/datasets/mohamedhanyyy/chest-ctscan-images/data

Data

Images are not in dcm format, the images are in jpg or png to fit the model. Data contain 3 chest cancer types which are Adenocarcinoma, Large cell carcinoma, Squamous cell carcinoma, and 1 folder for the normal cell. Data folder is the main folder that contain all the step folders inside Data folder are test, train and valid.

test represent testing set consisting of **20%** of the data.
train represent training set consisting of **70%** of the data.
valid represent validation set consisting of **10%** of the data.

Adenocarcinoma

Adenocarcinoma of the lung: Lung adenocarcinoma is the most common form of lung cancer accounting for 30 percent of all cases overall and about 40 percent of all non-small cell lung cancer occurrences. Adenocarcinomas are found in several common cancers, including breast, prostate and colorectal. Adenocarcinomas of the lung are found in the outer region of the lung in glands that secrete mucus and help us breathe. Symptoms include coughing, hoarseness, weight loss and weakness.

Large cell carcinoma

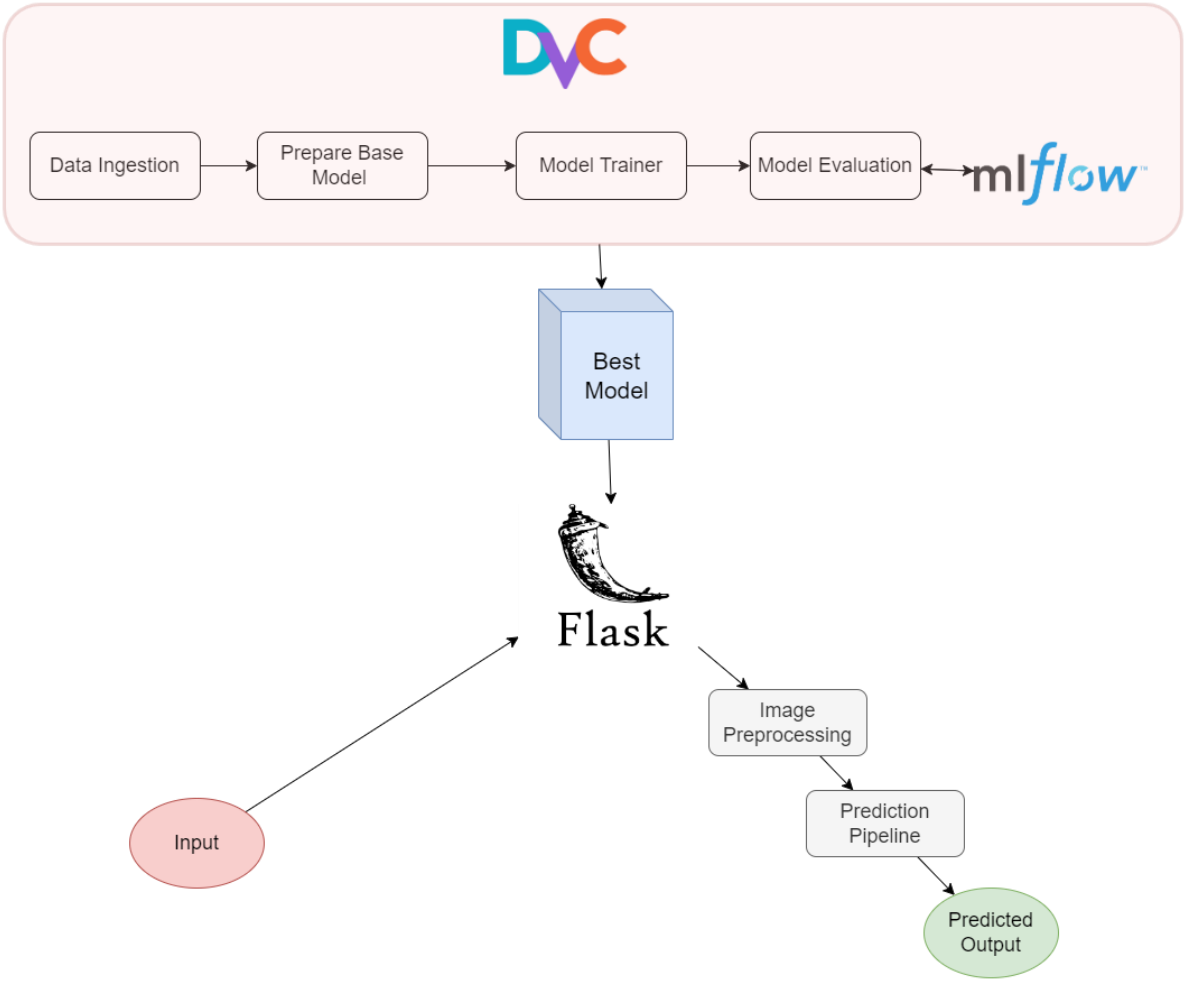
Large-cell undifferentiated carcinoma: Large-cell undifferentiated carcinoma lung cancer grows and preads quickly and can be found anywhere in the lung. This type of lung cancer usually accounts for 10 to 15 percent of all cases of NSCLC. Large-cell undifferentiated carcinoma tends to grow and spread quickly.

Squamous cell carcinoma

Squamous cell: This type of lung cancer is found centrally in the lung, where the larger bronchi join the trachea to the lung, or in one of the main airway branches. Squamous cell lung cancer is responsible for about 30 percent of all non-small cell lung cancers, and is generally linked to smoking.

And the last folder is the normal CT-Scan images.

4. System Architecture



System Requirements:

Hardware Requirements:

- CPU: A multi-core processor (Intel Core i5 or equivalent) for running training and inference tasks. For faster processing, a high-performance CPU or GPU is recommended.

- GPU (optional but recommended): NVIDIA CUDA-compatible GPU (e.g., NVIDIA GeForce GTX or RTX series) with CUDA support for accelerated deep learning computations.
- RAM: At least 8 GB of RAM for processing large datasets and training deep learning models efficiently.
- Storage: Sufficient storage space for storing datasets, model checkpoints, and Docker images. SSD storage is preferred for faster data access.

Software Requirements:

- Operating System: Compatible with major operating systems such as Windows, macOS, or Linux distributions like Ubuntu.
- Python: Python 3.8 for developing and executing the machine learning pipeline.
- TensorFlow: TensorFlow library (version 2.12.0) for building, training, and deploying deep learning models. Install TensorFlow with GPU support if using a compatible GPU.
- MLflow: MLflow library for managing the machine learning lifecycle, including experiment tracking, model packaging, and deployment. Install MLflow using pip.
- Docker: Docker engine for containerizing the application and ensuring consistent execution environments across different systems. Install Docker Desktop for Windows or Docker CE for Linux.
- DagsHub: DagsHub platform for version control, collaboration, and continuous integration of machine learning projects. Sign up for a DagsHub account and install the DagsHub CLI.
- DVC (Data Version Control): DVC for managing data versioning and reproducibility in machine learning projects. Install DVC using pip or package manager.

Dependencies and Libraries:

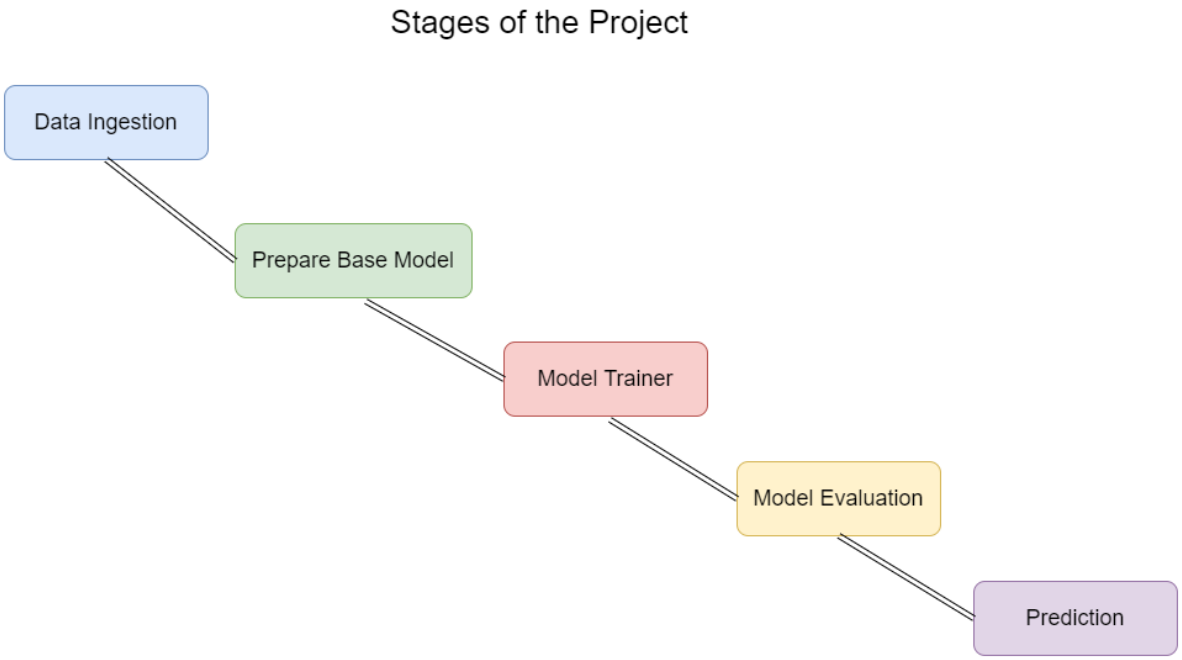
- NumPy, Pandas: Essential libraries for data manipulation, preprocessing, and analysis.
- scikit-learn: Library for machine learning algorithms, including data splitting, evaluation metrics, and model selection.
- Matplotlib, Seaborn: Visualization libraries for creating plots, charts, and visualizing model performance.
- OpenCV: Library for image processing and computer vision tasks, including loading and preprocessing CT scan images.

Additional Tools and Services:

- Git: Version control system for tracking changes in code, configurations, and project files.
- GitHub: Online platforms for hosting code repositories, collaborating with team members, and managing project development.
- Docker Hub: Registry service for storing and sharing Docker images, facilitating Docker image distribution and deployment.
- MLflow Tracking Server: Centralized server for storing experiment metadata, tracking metrics, and organizing MLflow experiments.
- Continuous Integration (CI) Services: Integration with CI services like GitHub Actions for automating testing, building, and deploying machine learning models.

- Cloud Services: AWS for scalable compute resources, storage, and deployment of machine learning applications.

5. Methodology



5.1 Data Ingestion:

- Collect images
- from Kaggle dataset link which included dataset created from BBC Images data.

5.2 Prepare Base Model:

- Choose the VGG16 convolutional neural network architecture as the base model for feature extraction due to its effectiveness in image classification tasks.
- Import the VGG16 model architecture and weights pretrained on ImageNet from the TensorFlow library.
- Choose the VGG16 convolutional neural network architecture as the base model for feature extraction due to its effectiveness in image classification tasks.
- Import the VGG16 model architecture and weights pretrained on ImageNet from the TensorFlow library.

5.3 Model Training:

- Split the preprocessed data into training, validation, and test sets.

- Choose the VGG16 convolutional neural network architecture as the base model for feature extraction due to its effectiveness in image classification tasks.
- Import the VGG16 model architecture and weights pretrained on ImageNet from the TensorFlow library.

5.4 Evaluation:

- Assess the model's performance using evaluation metrics on the validation set.
- Fine-tune hyperparameters based on evaluation results.
- Evaluate the model using metrics such as accuracy, sensitivity, specificity, and AUC-ROC on a validation dataset. Use MLflow to log and visualize metrics.
- Conduct qualitative assessment by visualizing model predictions overlaid on CT scan images.

5.5 Inference:

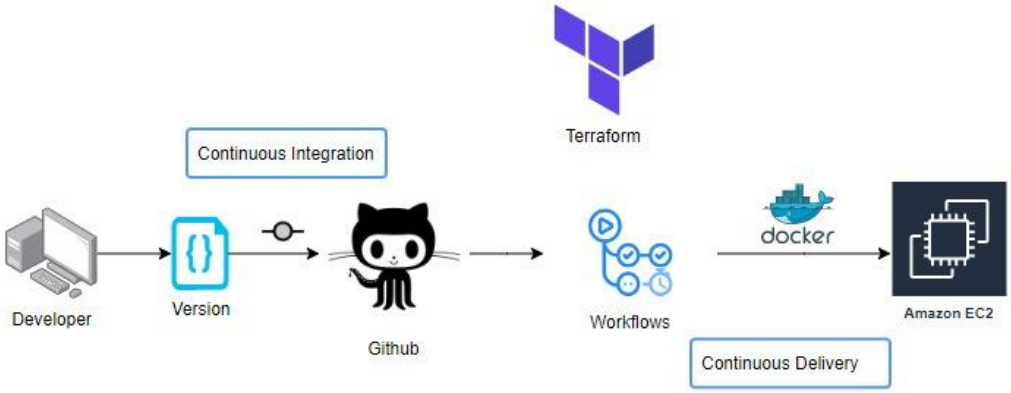
- Deploy the trained model to make predictions on new, unseen images.
- Classify incoming images into predefined categories using the deployed model.

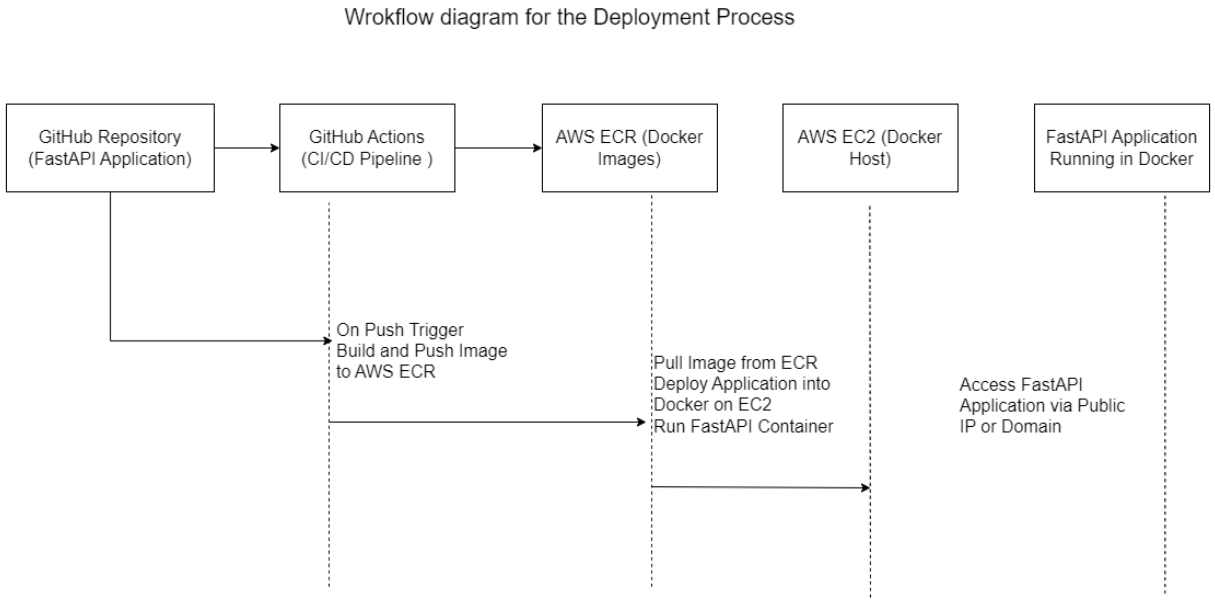
5.5 Deployment Process

- **API endpoint:** Flask serves as access points for interacting with the model through APIs, allowing users to input data and receive processed results. Accepts inputs and categorizes them into predefined classes or labels.
- **Containerization**: Containerize the model using Docker for reproducible deployment across different environments.
- **MLflow Model Packaging:** Package the trained model using MLflow's model packaging capabilities.
- **Deployment Pipeline**: Create a deployment pipeline using DVC to manage data and MLflow to manage model versions.
- **Version Control**: Use Git for version control, tracking changes in code, configurations, and datasets.
- **Continuous Integration**: Integrate CI/CD pipelines with DagsHub to automate testing, building, and deployment processes.
- **Deployment**: Deploy the packaged model to production environments or cloud platforms using Docker containers.
- **Model Monitoring**: Implement monitoring mechanisms to track model performance and detect drifts in predictions over time.

- **Reproducibility**: Ensure reproducibility of experiments and results using DVC for data and MLflow for model tracking.
- **Regular Updates:** Update the model periodically with new data and retrain as necessary to adapt to evolving patterns in lung tumor characteristics.
- **Cloud** : AWS ECR (Elastic Container Registry) is a service for storing, managing, and deploying Docker container images. EC2 (Elastic Compute Cloud) provides scalable virtual servers for various computing needs on AWS.

Deployment Architecture





6. Results and Discussion

7. Future Scope:

The future scope for Lung Tumor Classification from Chest CT Scans holds promising opportunities for advancement and innovation.

1. **Integration of Multi-Modal Data:** Incorporating additional imaging modalities such as PET scans or clinical data like patient demographics and medical history can provide complementary information for improved tumor classification and personalized treatment strategies.
2. **Enhanced Model Interpretability:** Developing techniques to interpret and visualize the decisions made by the VGG16 model can enhance trust and understanding of its predictions, facilitating collaboration between clinicians and AI systems in clinical decision-making.
3. **Real-Time Deployment:** Streamlining the deployment process to enable real-time classification of lung tumors directly within clinical workflows can enhance efficiency and enable timely interventions for patients.
4. **Continual Model Improvement:** Leveraging continuous integration and deployment pipelines with DVC and MLflow, along with collaborative platforms like DagsHub, facilitates iterative model improvement through feedback loops, dataset updates, and algorithmic enhancements.

5. **Clinical Validation and Regulatory Approval:** Conducting rigorous clinical validation studies and seeking regulatory approvals (such as FDA clearance) for AI-driven lung tumor classification systems can pave the way for their integration into routine clinical practice, thereby improving patient care and outcomes.

Overall, the future of lung tumor classification using the VGG16 model and associated technologies holds great promise for advancing the field of medical imaging and contributing to more accurate, efficient, and personalized diagnosis and treatment of lung cancer.

8. Conclusion

In conclusion, the integration of the VGG16 model in TensorFlow, alongside supporting technologies such as Docker, MLflow, DagsHub, and DVC, provides a robust framework for lung tumor classification from chest CT scans. Leveraging deep learning techniques, this approach offers accurate and efficient identification of malignant and benign tumors, aiding in early diagnosis and treatment planning. Through seamless deployment, version control, and experiment tracking facilitated by Docker, MLflow, DagsHub, and DVC, healthcare practitioners can access a reliable and reproducible solution for improved patient outcomes in the fight against lung cancer.