🧬 Breast Cancer Histopathology Image Classification using CNN

📌 Project Overview

This project focuses on classifying **histopathology images of breast tissue** into two categories:

- `0 → Benign` (non-cancerous)
- `1 → Malignant` (cancerous)

The goal is to support early cancer detection by building a **Convolutional Neural Network (CNN)** using **TensorFlow/Keras** that learns features from labeled histology images and accurately predicts the presence of malignancy.

🧠 Model Architecture

Input (224x224x3)
↓
Conv2D → ReLU (Extracts features like edges, cell patterns)
↓
MaxPooling2D (Reduces dimensionality, retains key features)
↓
Conv2D → ReLU
↓
MaxPooling2D
↓
Flatten (Converts feature maps to 1D vector)
↓
Dense → ReLU (Learns decision boundaries)
↓
Dropout (optional) (Prevents overfitting)
↓
Dense → Sigmoid (Binary output: Benign or Malignant)

✅ What I Did

- 📁 Structured dataset into `train/`, `validation/`, and `test/` folders with class labels `0` and `1`

- 🧠 Built and trained a CNN model using TensorFlow/Keras

- 📊 Achieved **~77% test accuracy**

- 🛑 Implemented **EarlyStopping** to address overfitting

- 📈 Plotted training/validation **accuracy and loss curves**

- 🔍 Developed a script to classify **new input images**

🧪 Example Output

```bash
img1.jpg: 1 → Malignant (Confidence: 0.59)
img2.jpg: 0 → Benign (Confidence: 0.91)
```

🛠 Technologies Used

- TensorFlow / Keras
- Matplotlib
- NumPy
- PIL (Python Imaging Library)
- ImageDataGenerator (for real-time image augmentation)
- EarlyStopping (Keras callback)

📁 Project Structure

```
Breast_Cancer_Classifier/
├── data/
│   ├── train/0/1/
```

```
│   ├── validation/0/1/
│   └── test/0/1/
├── outputs/
│   ├── model.h5
│   ├── accuracy_plot.png
│   └── loss_plot.png
├── inference/
│   └── img1.jpg, img2.jpg
├── src/
│   ├── train.py
│   ├── model_builder.py
│   ├── data_loader.py
│   └── evaluate.py
├── predict_single.py
└── predict_multiple.py
```

🚀 How to Run

1. Clone this repo and install requirements:
2. pip install -r requirements.txt
3. Prepare dataset as shown in data/ structure.
4. Train the model:
5. python src/train.py
6. Evaluate:
7. python src/evaluate.py
8. Predict on a single image:
9. python predict_single.py
10. Predict on all images in a folder:
11. python predict_multiple.py

🎯 Future Work

- Add **Grad-CAM** visualizations for interpretability
- Deploy as a **web app** using Flask or Streamlit
- Expand to multi-class classification with more subtypes

👩‍💻 Author

Shreya Umesh Naidu
Graduate Student in Computer Science
University of Texas at Arlington