ChatGPT

Duke Breast MRI Segmentation Project

Project Overview

This project implements a convolutional neural network (CNN), specifically a U-Net architecture, to perform **semantic segmentation** of breast MRI scans. The goal is to identify and segment regions of interest (e.g., tumors or abnormalities) in the MRI scans using corresponding binary mask annotations.

Project Structure and Files

1. `dataset.py`

- Defines the `BreastMRIDataset` class, a custom PyTorch dataset that loads pairs of MRI scan slices and their corresponding binary masks.
- Handles loading, preprocessing, and transforming images.
- Assumes data is stored in structured directories:
- `data/processed/images/[patient_id]/[slice].png`
- `data/processed/masks/[patient_id]/[slice].png`

2. `model.py`

- Implements a U-Net architecture:
- **Encoder (Downsampling)**: Captures contextual features via convolution + pooling.
- **Bottleneck**: Highest level features.
- **Decoder (Upsampling)**: Combines low-level spatial features from encoder with decoder via skip connections.
- **Output**: Final segmentation map (1-channel output for binary classification).

3. `train.py`

- Trains the U-Net on the dataset.
- Loss function: Combination of:
- **Binary Cross-Entropy with Logits**: Penalizes incorrect pixel-wise classifications.
- **Dice Loss**: Encourages spatial overlap between prediction and ground truth.
- Training loop:
- Optimizer: Adam
- Learning rate: 1e-3
- Epochs: 10
- Batch size: 8

4. `visualize.py`

- Loads a trained model and evaluates it on samples from the dataset.
- For each of 5 patients:
- Visualizes all slices.

- Displays:
    - Original image
    - True mask
    - Predicted mask with pixel-wise accuracy

Mathematical Foundation of Training

1. Binary Cross-Entropy with Logits Loss

This loss is defined as:

$$\text{BCEWithLogitsLoss}(p, y) = -[y \cdot \log(\sigma(p)) + (1 - y) \cdot \log(1 - \sigma(p))]$$

where: - $y \in \{0, 1\}$: Ground truth label per pixel - p : Raw output (logits) from the model - $\sigma(p)$: Sigmoid function, converting logits to probabilities

2. Dice Loss

Measures overlap between predicted and true segmentation masks:

$$\text{Dice} = \frac{2|A \cap B|}{|A| + |B|}$$

In PyTorch terms:

$$\text{DiceLoss}(\hat{y}, y) = 1 - \frac{2 \cdot (\hat{y} \cdot y).\text{sum}() + \epsilon}{\hat{y}.\text{sum}() + y.\text{sum}() + \epsilon}$$

- Encourages greater overlap in prediction vs ground truth. - ϵ is a small constant to prevent division by zero.

3. Combined Loss

The total loss during training is:

$$\mathcal{L}_{total} = \mathcal{L}_{BCE} + \mathcal{L}_{Dice}$$

This allows the model to both learn accurate pixel-level classification and maintain coherent, spatially accurate segmentations.

4. Accuracy Metric

During visualization:

$$\text{Accuracy} = \frac{\text{Number of correct pixels}}{\text{Total number of pixels}} \times 100$$

Pixels are considered correct if: - Predicted probability > 0.5 for true label = 1 - Predicted probability <= 0.5 for true label = 0

Results and Visualization

Example Visualization:

For 5 different patients, each slice is visualized in a row of 3 columns: 1. Original Image 2. True Segmentation Mask 3. Predicted Segmentation Mask (Hot colormap, with accuracy %)

This helps assess both the qualitative and quantitative performance of the model.

Key Takeaways

- U-Net is effective for biomedical image segmentation due to its symmetry and skip connections.
- Combining BCE and Dice loss improves both pixel-wise and spatial performance.
- Visualization of slices per patient is crucial for debugging and clinical interpretability.

Potential Improvements

- Add validation set and report metrics (Dice score, IoU, etc.).
- Use 3D U-Net for volumetric context.
- Apply post-processing (morphological operations) for cleaner masks.
- Train longer or with more data to reduce plateau in loss after epoch 4.

Final Notes

This project represents a full segmentation pipeline: - Data loading and preprocessing - Model training - Quantitative and qualitative evaluation It can be extended into clinical workflows with minimal modifications for production environments.