Detecting Lung Cancer on CT Scan Images

Afra Seeyad, B210017CS
Siri Pagadala B210033CS

Yasaswini B210503CS

Gayatri B210558CS
Vijayalakshmi B211260CS

Proposed Methodology

The proposed methodology aims to address the critical need for early detection of lung cancer using CT scan images. The approach combines segmentation techniques with Convolutional Neural Networks (CNNs) for accurate diagnosis. The process involves several key steps:

Preprocessing and Visualization of Dataset: Utilizing the LUNA16 dataset, which consists of CT scan images of lung nodules, preprocessing is performed to filter relevant annotations and discard irrelevant scans. Visualization of the dataset is facilitated using the Pydicom library, providing insights into the structure and metadata of the CT images.

Watershed Algorithm for Segmentation: The watershed algorithm is employed for segmentation, separating lung structures and cancerous nodules from surrounding tissues. This process involves the extraction of internal and external markers from CT images, followed by the application of the watershed transformation to delineate regions of interest.

Integration of Sobel Filter: To enhance segmentation accuracy, the Sobel filter is integrated with the watershed algorithm, effectively removing external noise and isolating lung structures. Further morphological operations are applied to refine the segmentation results.

CNN Models for Classification: Three different CNN models are explored for lung cancer classification:

Sequential_1: Basic CNN architecture with convolutional layers, max-pooling, and dropout.

Sequential_2: Deep CNN with additional layers for improved feature extraction.

VGG16-net (Transfer Learning): Leveraging pre-trained VGG-16 architecture with modifications to adapt to the dataset.

Training and Evaluation: The models are trained on segmented lung images using a combination of data augmentation techniques and optimized hyperparameters. Model performance is evaluated based on accuracy, loss, and validation metrics.

Tools / Techniques Utilized:

1. Pydicom library for dataset visualization and metadata extraction

- Pydicom is a Python library specifically designed for working with DICOM (Digital Imaging and Communications in Medicine) files, which are commonly used for medical imaging, including CT scans.
- The library allows for the parsing and extraction of metadata stored within DICOM files, such as patient information, image properties, and acquisition parameters.
- Pydicom also provides functionality for visualizing DICOM images and performing basic image processing tasks, enabling researchers and practitioners to gain insights into the dataset's structure and content.

2. Watershed algorithm for image segmentation:

- The watershed algorithm is a classical image processing technique used for segmentation, particularly in scenarios where distinct object boundaries need to be delineated.
- In the context of lung cancer detection, the watershed algorithm can be applied to separate lung structures and cancerous nodules from surrounding tissues in CT scan images.
- The algorithm treats the grayscale image as a topographic map, with pixel intensities representing elevations, and identifies ridges and valleys to partition the image into distinct regions.

3. Sobel filter integration for noise reduction and feature enhancement:

- The Sobel filter is a gradient-based edge detection operator used to highlight edges and discontinuities in images.
- By integrating the Sobel filter with the watershed algorithm, noise in the CT scan images can be reduced, and relevant features, such as lung boundaries and nodule contours, can be enhanced.
- This integration improves the accuracy of segmentation by effectively isolating lung structures and distinguishing them from background noise.

4. Convolutional Neural Networks (CNNs) implemented using TensorFlow/Keras for lung cancer classification:

- CNNs are a class of deep learning models particularly well-suited for image analysis tasks, including classification, segmentation, and object detection.
- In the context of lung cancer classification, CNNs can be trained on segmented CT scan images to distinguish between cancerous and non-cancerous nodules.
- TensorFlow and Keras are popular deep learning frameworks that provide high-level APIs for building, training, and deploying neural networks, simplifying the implementation and experimentation process.

5. Transfer learning with VGG-16 architecture for leveraging pre-trained models:

- Transfer learning involves leveraging knowledge gained from pre-trained models on large datasets and applying it to related tasks or domains with limited labeled data.
- VGG-16 is a pre-trained CNN architecture widely used for image classification tasks, initially trained on the ImageNet dataset.
- By fine-tuning the VGG-16 model on segmented CT scan images of lung nodules, researchers can benefit from the learned feature representations and potentially achieve better performance compared to training from scratch.

6. Data augmentation to enhance model generalization and robustness:

- Data augmentation involves applying a variety of transformations to the training data, such as rotation, scaling, cropping, and flipping, to artificially increase the diversity of the dataset.
- Augmenting the data helps prevent overfitting and improves the generalization ability of the model by exposing it to variations in the input data.
- This technique is particularly useful when working with limited datasets, as it effectively increases the effective size of the training set and improves the model's ability to handle variations in input images.

Existing Approach:

The existing approach to lung cancer detection often relies on invasive methods such as biopsies or surgeries, posing risks to patients and increasing diagnostic complexities. CT imaging offers a non-invasive alternative but suffers from high false-positive rates and radiation exposure concerns. The proposed approach addresses these limitations by combining advanced segmentation techniques with state-of-the-art CNN models for accurate and efficient lung cancer diagnosis.

Design and Coding:

Model_1:

```
def define_model():
model = Sequential()
model.add(Conv2D(32, (3, 3), activation='relu', kernel_initializer='he_uniform',
padding='same', input_shape=(50, 50, 1)))
model.add(BatchNormalization())
model.add(Conv2D(32, (3, 3), activation='relu', kernel_initializer='he_uniform',
padding='same'))
model.add(BatchNormalization())
model.add(MaxPooling2D((2, 2)))
model.add(Dropout(0.1))
model.add(Conv2D(64, (3, 3), activation='relu', kernel_initializer='he_uniform',
padding='same'))
model.add(BatchNormalization())
model.add(Conv2D(64, (3, 3), activation='relu', kernel_initializer='he_uniform',
padding='same'))
model.add(BatchNormalization())
model.add(MaxPooling2D((2, 2)))
model.add(Dropout(0.2))
model.add(Conv2D(128, (3, 3), activation='relu', kernel_initializer='he_uniform',
padding='same'))
model.add(BatchNormalization())
model.add(Conv2D(128, (3, 3), activation='relu', kernel_initializer='he_uniform',
padding='same'))
model.add(BatchNormalization())
model.add(MaxPooling2D((2, 2)))
model.add(Dropout(0.3))
model.add(Conv2D(128, (3, 3), activation='relu', kernel_initializer='he_uniform',
padding='same'))
model.add(BatchNormalization())
model.add(Conv2D(128, (3, 3), activation='relu', kernel_initializer='he_uniform',
padding='same'))
model.add(BatchNormalization())
model.add(MaxPooling2D((2, 2)))
model.add(Dropout(0.4))
model.add(Conv2D(128, (3, 3), activation='relu', kernel_initializer='he_uniform',
padding='same'))
model.add(BatchNormalization())
model.add(Conv2D(128, (3, 3), activation='relu', kernel_initializer='he_uniform',
padding='same'))
model.add(BatchNormalization())
model.add(MaxPooling2D((2, 2)))
model.add(Dropout(0.5))
model.add(Flatten())
model.add(Dense(128, activation='relu', kernel_initializer='he_uniform'))
model.add(BatchNormalization())
model.add(Dropout(0.5))
```

```
model.add(Dense(128, activation='relu', kernel_initializer='he_uniform'));return model

mobile = define_model()
```

Model_2:

```
op_layer = mobile.output
final_layer = Dense(128,activation='relu',kernel_initializer='he_uniform')(op_layer)
final_layer =
Dense(128,activation='relu',kernel_initializer='he_uniform')(final_layer)
final_layer = Dense(2,activation= 'softmax')(final_layer)
from keras.models import Model
# Define model input and output
model = Model(inputs = mobile.input , outputs = final_layer)
opt = SGD(lr=0.001, momentum=0.9)
import keras
optimizer_sgd = keras.optimizers.Adam(learning_rate=0.001)
model.compile(optimizer=optimizer_sgd, loss='categorical_crossentropy',
metrics=['accuracy'])
```

Conclusion

The proposed approach presents a promising strategy for the early detection of lung cancer using CT scan images. By leveraging advanced segmentation algorithms and deep learning models, the methodology achieves accurate classification of cancerous nodules while minimizing false positives. The integration of transfer learning with the VGG-16 architecture enhances the model's ability to generalize across different datasets and imaging modalities. Further refinement and optimization of the methodology hold the potential to significantly impact clinical practice and improve patient outcomes in the diagnosis and treatment of lung cancer.