Multi modal training  
  
Abstract:

This report presents a comprehensive study of two tasks, First task is getting to know how the nnUNet software (a deep learning model designed for the Biomedical care) is designed and how the training process is done, and the second task is on the training and evaluation of a deep learning model designed for early Alzheimer's disease detection using structural MRIs. A comparison with traditional volume/thickness models is conducted, showcasing the deep learning model's superior accuracy, speed, and potential for forecasting disease progression.

1. Introduction:

1.1 Task 1 - nnUNetv2 Model Training:

A deep learning-based segmentation method that automatically configures itself, including preprocessing, network architecture, training and post-processing for any new task. The key design choices in this process are modeled as a set of fixed parameters, interdependent rules and empirical decisions. Without manual intervention, nnU-Net surpasses most existing approaches, including highly specialized solutions on 23 public datasets used in international biomedical segmentation competitions.

1.2 Task 2 - Generalizable Deep Learning Model:

This task focuses on designing a CNN for Alzheimer's detection. Evidence is provided, highlighting the effectiveness of instance normalization over batch normalization, the negative impact of early spatial downsampling, the consistent gains of model widening, and the moderate improvement achieved by incorporating age information. Compared with the volume/thickness model, the deep-learning model is accurate, significantly faster, and capable of forecasting disease progression.

3. Datasets and Preprocessing:

3.1 Task 1 – Dataset:

nnU-Net will systematically analyze the provided training cases and create a “dataset fingerprint”. nnU-Net then creates several U-Net configurations for each dataset:

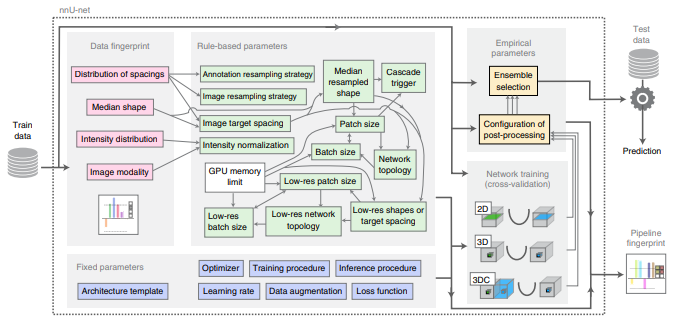
* 2d: a 2D U-Net (for 2D and 3D datasets)
* 3d\_fullres: a 3D U-Net that operates on a high image resolution (for 3D datasets only)
* 3d\_lowres → 3d\_cascade\_fullres: a 3D U-Net cascade where first a 3D U-Net operates on low resolution images and then a second high-resolution 3D U-Net refined the predictions of the former (for 3D datasets with large image sizes only)

Utilizing datasets such as BraTs2021 (Brain Tumor Segmentation), AMOS22 (Abdominal Multi organ Segmentation), KiTS23(Kidney Tumor Segmentation), and BTCV(Beyond The Cranial Vault Segmentation) for nnUNetv2 training.  
  
3.2 Task 2 - Dataset:

For deep learning model training, T1-weighted structural MRI scans from the ADNI dataset are used. Over 3000 preprocessed scans are categorized into AD, MCI, and CN based on memory tasks, adjusted for education level. Specific preprocessing procedures involve multiplanar reconstruction, Gradwarp, B1 non-uniformity correction, and N3 intensity normalization.

4. Data Preprocessing:

4.1 Task 1 - Dataset Preprocessing:



4.2 Task 2 - Data Preprocessing:

Utilizing Clinica, FSL, SPM, and Freesurfer, the ADNI dataset undergoes preprocessing involving Dartel template registration, normalization into MNI coordinate space, and careful subject splitting into training, validation, and test sets to prevent data leakage.

5. Methodology:

5.1 Task 1 - Preprocessing for Training:

Steps for preprocessing the data for nnUNetv2 training, including downloading raw datasets and adjusting batch sizes to avoid memory errors.

5.2 Task 1 - Training the Dataset:

Description of the folds in training, along with the creation of Dataset\_generator python files for nnUNetv2.

6. Results:

6.1 Task 1 Results –

The nnU-Net, a segmentation method rooted in deep learning, exhibits the capability to dynamically configure itself for various tasks within the biomedical domain. This encompasses automatic adjustments in preprocessing, network architecture, training, and post-processing.

As an illustration, the configuration of batch size, patch size, and network topology is determined by principles such as the preference for larger batch sizes for accurate gradient estimates, the importance of larger patch sizes for enhanced contextual information, and the necessity of a deep network topology to accommodate a sufficiently large receptive field. The resulting heuristic rule advises initializing the patch size to the median image shape and iteratively reducing it while adjusting the network topology until the network can be trained with a batch size of at least two, considering GPU memory constraints.

7. Model Design Insights:

Insights into the design choices for the CNN, emphasizing the effectiveness of instance normalization, spatial downsampling impact, model width vs. depth, and the incorporation of age information.

8. Model Comparison:

A comprehensive comparison with volume/thickness models, emphasizing the deep learning model's accuracy, speed, and forecasting capabilities.

9. Visualization and Interpretability:

Discussion on visualizations of voxel importance to aid in interpreting the deep learning model's decision-making process.

10. Prerequisites and Data:

Requirements for running the code, including Python, PyTorch, and other libraries. Instructions for downloading ADNI data and obtaining proper approvals.

11. Neural Network Training:

Step-by-step instructions for training the neural network on the ADNI dataset using the provided code.

12. Model Evaluation:

Code and procedures for loading and evaluating the trained model.

13. Progression Analysis:

Analysis of disease progression, particularly for Mild Cognitive Impairment (MCI) subjects, showcasing the predictive capabilities of the deep learning model.

14. Conclusion:

Summarized key findings, emphasizing the deep learning model's superiority in accuracy and speed, with potential for early detection and progression forecasting.

15. Future Work:

Suggested avenues for future research, including exploring different neural network architectures.

16. Acknowledgments:

Recognition of individuals, organizations, and data sources that contributed to the research.

17. References:

Citations of relevant papers, tools, and frameworks used in the project.