UNIVERSITY OF
BIRMINGHAM

Transfer Learning for Alzheimer's Disease Detection: Adapting Video Classification Models for MRI Scans

Rhys W. Alexander (2458177)

Final project report submitted
in partial fulfilment for the degree of
B.SCI. IN ARTIFICIAL INTELLIGENCE AND COMPUTER SCIENCE

Date: 2nd April 2025
Word count: X,XXX

Project supervisor:
Dr Rickson Mesquita

Contents

1 Abstract

2 Introduction

3 Literature review

4 Methodology

4.1 Data Acquisition and Characteristics

The Alzheimer's Disease Neuroimaging Initiative (ADNI) database served as the primary data source, providing standardized MRI acquisitions with corresponding clinical diagnoses. ADNI was selected over alternatives (including OASIS) for its comprehensive coverage, acquisition protocols, and expert-validated diagnoses.

4.1.1 Dataset Composition

All selected scans were T1-weighted MPRAGE sequences (1.5T or 3T, $1mm^3$ isotropic resolution), chosen for optimal gray/white matter contrast, standardized acquisition parameters, and sensitivity to atrophy biomarkers. Additionally, the widespread clinical availability and established role of MPRAGE in AD assessment made it an ideal choice for this study. The final dataset contained 1,300 scans from 408 unique subjects, balanced between diagnostic categories:

Partition	**AD**	**CN**
Training	512 scans (133 subjects)	511 scans (115 subjects)
Validation	69 scans (35 subjects)	70 scans (45 subjects)
Test	69 scans (35 subjects)	69 scans (45 subjects)

Table 1: Distribution of scans and subjects across dataset partitions

4.1.2 Diagnostic Criteria

Subjects were classified as Alzheimer's Disease (AD) or Cognitively Normal (CN) based on NINCDS-ADRDA criteria. Initially, the dataset contained approximately 33% AD and 67% CN cases. To address class imbalance and potential overfitting issues identified during preliminary experiments, additional AD scans were incorporated and CN subjects carefully sampled to achieve a balanced 50/50 diagnostic distribution.

The binary classification focus (excluding Mild Cognitive Impairment) reflects the clearer structural changes observable in established AD, particularly hippocampal atrophy, which serves as a primary biomarker for disease progression. Subject-level isolation between

dataset partitions was strictly enforced to prevent data leakage, ensuring realistic performance assessment for unseen individuals.

4.2 Preprocessing Pipeline

4.2.1 Initial Processing and Skull Stripping

Raw DICOM images were converted to NIfTI format using `dicom2nifti` with reorientation and compression enabled. This created unified volumetric files suitable for 3D analysis. Skull stripping was performed using SynthStrip, a deep learning-based method that represents the current state-of-the-art for brain extraction. It was selected for its superior performance with atrophied brains. Unlike traditional threshold-based methods (e.g., BET), SynthStrip preserved critical cortical boundaries even with atrophied brains and better handled the variability in the ADNI dataset. Despite requiring 2.5 minutes per scan, the improved quality justified this approach by preventing potential misinterpretation of artifacts as disease-related changes.

4.2.2 Volume Standardization

All volumes were resampled to isotropic 1×1×1mm voxels using ANTs with third-order spline interpolation. This standardization ensured consistent spatial representation, eliminated scanner-specific resolution variability, and enabled uniform convolutional filter operations across all dimensions.

4.2.3 Adaptive Cropping Strategy

A key methodological innovation was the implementation of an adaptive cropping procedure followed by reshaping to 128×128×128 dimensions. The approach:

1. Identified brain-containing regions using intensity thresholding
2. Applied cropping with minimal padding (3 voxels)
3. Used cubic interpolation to reach the target dimensions

This method preserved approximately 35% more effective resolution for critical structures like the hippocampus compared to naive downsampling. The 128^3 dimension balanced preserving anatomical detail with memory constraints for model training.

4.2.4 Intensity Normalization and Orientation

N4 bias field correction was applied to mitigate intensity inhomogeneities from magnetic field variations. This prevents intensity variations that might be misinterpreted as structural changes. All volumes were reoriented to Right-Anterior-Superior (RAS) orientation

to ensure consistent directionality, allowing the model to focus solely on relevant structural differences rather than arbitrary orientation variations.

4.2.5 Omission of Spatial Normalization

Despite its common use in neuroimaging pipelines, registration to standard space (e.g., MNI152) was deliberately omitted for several reasons:

1. Preservation of native atrophy patterns that could be distorted during normalization
2. Reliance on CNN translation invariance to identify structures without explicit alignment
3. Avoidance of interpolation artifacts that might smooth critical structural boundaries
4. Computational efficiency gains without compromising classification performance

Validation experiments confirmed that models trained on native-space data performed comparably to or better than those using normalized data, supporting this methodological decision and aligning with recent literature suggesting deep learning models for brain MRI benefit from native-space learning.

The entire pipeline produced 1,300 preprocessed volumes with consistent dimensions, orientation, and intensity characteristics while preserving the structural variations essential for AD classification.

4.3 Data Splitting Strategy

A methodologically rigorous data splitting approach was implemented to prevent data leakage while maintaining diagnostic balance across partitions. Unlike conventional image classification tasks, neuroimaging datasets require subject-level rather than scan-level splitting since multiple scans often exist for the same individual.

4.3.1 Subject-Level Isolation

A strict subject-level isolation approach ensured no individual appeared in multiple dataset partitions—a critical decision after initial experiments revealed artificially inflated performance metrics (90% accuracy) when subjects were allowed to cross partition boundaries. Complete subject isolation produced a more realistic performance assessment (70% accuracy), better reflecting the model's generalization capability to unseen individuals.

4.3.2 Partition Distribution

The dataset was divided following an 80/10/10 (train/validation/test) ratio using a round-robin algorithm that:

1. Grouped subjects by diagnostic condition
2. Sorted subjects in ascending order by scan count
3. Allocated subjects to partitions round robin to insure subject diversity across partitions
4. Final scan counts were balanced to maintain equal scan counts per diagnostic category

This approach yielded a balanced distribution with 1,023 training scans (512 AD/511 CN), 139 validation scans (69 AD/70 CN), and 138 test scans (69 AD/69 CN). The strict isolation maintained 203 unique subjects in training, 80 in validation, and 80 in test sets, with diagnostic balance preserved in each partition.

Data Leakage Prevention To prevent subtle forms of data leakage, subject identifiers were rigorously tracked and preprocessing parameters (such as intensity normalization statistics) were computed independently within each partition. This methodologically sound approach ensured that performance metrics would accurately reflect the model's ability to generalize to entirely new individuals, rather than merely recognizing previously seen subjects in different scans.

4.4 Data Augmentation

Data augmentation was strategically implemented to improve model generalization while preserving diagnostically relevant features. Through systematic experimentation, a minimal yet effective set of transformations was identified:

```
tio.Compose([
    tio.RandomNoise(mean=0.0, std=0.1, p=0.3),
    tio.RandomGamma(log_gamma=(-0.2, 0.2), p=0.3),
    tio.ZNormalization(),
])
```

This approach was applied exclusively to the training set, while validation and test sets received only Z-normalization to maintain evaluation consistency.

Each technique addressed specific neuroimaging considerations: Random noise (30% probability, $=0.1$) simulated scanner variability and promoted robustness to image quality differences; Gamma adjustment (± 0.2 range, 30% probability) mimicked contrast variations between scanners; Z-normalization standardized intensity values across all scans for consistent feature extraction.

Notably, several common augmentation techniques were deliberately excluded after experimental evaluation showed either no benefit or negative impact:

- **Geometric transformations** (rotations, flips) significantly increased training time (20 vs 5 epochs) without improving validation accuracy, likely due to inherent orientation variability already present in MRI data.
- **Random scaling** (0.9-1.1) showed no generalization improvement and potentially disrupted the carefully standardized voxel dimensions.

The final strategy evolved from extensive transformations to this focused set through iterative evaluation of validation performance and convergence speed, representing an optimal balance between enhancing robustness and preserving critical structural features essential for AD classification.

4.5 Model Architectures

4.5.1 3D ResNet Architecture

The primary model architecture employed was a modified 3D ResNet-18 (r3d_18), selected for several key characteristics:

1. **Residual connections**: These skip connections mitigate the vanishing gradient problem in deep networks, allowing effective training even with limited data.
2. **3D convolutions**: Unlike 2D approaches that process each slice independently, 3D convolutions capture volumetric patterns across all three dimensions, preserving spatial relationships critical for detecting hippocampal atrophy.
3. **Parameter efficiency**: With approximately 33 million parameters, ResNet-18 offered a balance between model capacity and computational efficiency, enabling training on consumer hardware.
4. **Proven effectiveness**: The ResNet architecture family has demonstrated robust performance across numerous computer vision tasks, including medical imaging applications.

The implementation utilized the PyTorch `torchvision.models.video` module, specifically the `r3d_18` model pre-trained on the Kinetics400 action recognition dataset.The first convolutional layer was modified to accept single-channel MRI volumes rather than the three-channel RGB videos used in the original architecture. The final fully connected layer was replaced to output two classes (AD vs. CN) instead of the 400 action classes in the original model. These architectural modifications preserved the core feature extraction capabilities of the ResNet model while adapting it to the specific requirements of binary volumetric MRI classification.

4.5.2 Transfer Learning Strategy

A systematic transfer learning approach was implemented with layer freezing to leverage the pre-trained weights from video classification. Early convolutional layers (stem, layer1, layer2, layer3) were frozen to preserve general low-level feature detectors learned from video data. The final residual block (layer4) and fully connected layer were unfrozen to allow adaptation to MRI-specific features. This approach maintained approximately 25% of parameters as frozen (8.2 million) while fine-tuning the remaining 75% (24.9 million), striking a balance between preserving pre-trained knowledge and adapting to the target domain. Initial experiments with more aggressive freezing (keeping only the final fully connected layer trainable) resulted in numerical instabilities during training, manifested as NaN losses, suggesting that significant domain adaptation was necessary given the substantial differences between video action recognition and MRI classification.

Notably, the learning rate strategy was aligned with this transfer learning approach, implementing a higher learning rate (10×) for the newly initialized fully connected layer compared to the pre-trained but unfrozen convolutional layers. This differential learning rate strategy allowed more aggressive adaptation in the task-specific output layer while making more conservative updates to the pre-trained feature extraction layers.

4.5.3 Alternative Architecture Exploration

To validate the architectural choices, and to justify using 3D convolutioons as opposed to the previously researched 2D methods, alternative models were explored:

1. **Mixed Convolution 3D Network**: This model (MC3-18) uses a hybrid approach combining 2D and 3D convolutions, hypothesized to potentially offer computational efficiency while maintaining performance.

   Experimental results with MC3-18 showed less stable training dynamics and inferior performance compared to the pure 3D approach of R3D-18, supporting the importance of fully volumetric feature extraction for structural MRI analysis. The differences in performance provided empirical justification for the primary architectural choice.

2. **(2+1)D Convolution Network**: Following the investigation of MC3-18, a (2+1)D architecture was also evaluated. This approach decomposes 3D convolutions into separate spatial (2D) and temporal (1D) convolutions, a technique that has shown promise in video classification tasks.

   Results with the (2+1)D architecture revealed performance that was slightly worse than MC3-18, continuing the observed trend that classification accuracy decreased as the model architecture incorporated more 2D elements. This progression (R3D

> MC3 > (2+1)D) strongly suggests that preserving the full 3D spatial context through pure 3D convolutions is critical for detecting the subtle volumetric patterns associated with Alzheimer's disease in MRI data.

3. **Multiscale Vision Transformer**: Recent advances in vision transformers prompted investigation of their potential for 3D MRI classification. However, initial implementation attempts revealed significant computational barriers:

   (a) Memory requirements exceeded available hardware capabilities (32GB RAM requirement for 128×128×128 volumes)

   (b) Architectural mismatch between the input dimensions required by MViT (designed for 16×224×224 video clips) and the cubical 128×128×128 MRI volumes

   (c) Transformer architectures typically require substantially larger training datasets than were available

   These constraints prevented full evaluation of transformer-based approaches, highlighting an important practical limitation in applying state-of-the-art vision models to medical imaging with limited computational resources.

4.5.4 Parameter Counts and Computational Considerations

The final model architecture parameters were:

- **Total parameters**: 33,148,482
- **Trainable parameters**: 24,909,826 (75.15%)
- **Frozen parameters**: 8,238,656 (24.85%)

These figures represent a significant reduction compared to larger architectures like ResNet-50 or ViT variants, making training feasible on consumer-grade hardware while maintaining sufficient capacity for the classification task. The reduced parameter count also potentially mitigated overfitting given the relatively small dataset size.

4.6 Training Framework and Implementation

4.6.1 Computational Environment

Model training was conducted on an M1 Mac using the Metal Performance Shaders (MPS) acceleration framework. This hardware configuration imposed certain constraints on the implementation, with each epoch requiring approximately one hour of computation time and total training runs taking upwards of 20 hours. These hardware limitations influenced several implementation decisions, including batch size selection and model architecture choices.

While attempts were made to optimize training speed through techniques like mixed precision training and CPU-GPU synchronization optimization, performance improvements were minimal. The majority of computational time was consumed by the model's forward pass through the 3D volumes, which could not be significantly accelerated without more powerful hardware.

4.6.2 Hyperparameter Selection

Hyperparameters were carefully selected through systematic experimentation and validation performance tracking. The final configuration included:

- **Learning rate strategy**: A dual learning rate approach was implemented, with the newly initialized fully connected layer receiving a 10× higher learning rate (0.001) compared to the fine-tuned convolutional layers (0.0001). This differential strategy allowed more aggressive adaptation in the task-specific output layer while making conservative updates to the pre-trained feature extraction layers.
- **Optimizer**: AdamW with weight decay (0.01) was selected to mitigate overfitting given the relatively small dataset size. The weight decay regularization helped constrain the model's parameter space, particularly important when working with pre-trained features.
- **Batch size**: Limited to 2 samples per batch due to memory constraints of processing 128×128×128 volumetric inputs. Despite the small batch size, training stability was maintained through appropriate learning rate selection.
- **Learning rate scheduling**: Cosine annealing with warm restarts ($T_0 = 5$) was implemented to prevent convergence to local minima. This scheduling strategy periodically reduces the learning rate following a cosine curve before resetting it, allowing the model to escape suboptimal solutions.

The hyperparameter selection process was guided by both theoretical considerations and empirical validation, with each configuration tracked in Weights & Biases to enable systematic comparison.

4.6.3 Loss Function and Class Weighting

A weighted cross-entropy loss function was implemented to address potential class imbalance, particularly important during initial experiments when the dataset had not yet been fully balanced:

```
# Calculate class weights for imbalanced data
num_ad = train_dataset.labels.count(1)
num_cn = train_dataset.labels.count(0)
```

```
total = num_ad + num_cn

# Inverse frequency weighting
weight_cn = total / (2 * num_cn) if num_cn > 0 else 1.0
weight_ad = total / (2 * num_ad) if num_ad > 0 else 1.0

class_weights = torch.tensor([weight_cn, weight_ad], device=device)
criterion = nn.CrossEntropyLoss(weight=class_weights)
```

This weighting strategy ensured that both diagnostic classes contributed equally to the loss function regardless of their representation in the training set. The weights were dynamically calculated for each training run based on the actual class distribution, providing robustness to dataset modifications.

4.6.4 Early Stopping Criteria

To prevent overfitting and optimize computational resource usage, an early stopping mechanism was implemented with a patience of 5 epochs. This approach monitored validation metrics (accuracy and loss) and terminated training when no improvement was observed for five consecutive epochs. The early stopping implementation maintained separate counters for accuracy and loss improvements, ensuring training continued as long as either metric showed enhancement. This dual-metric approach prevented premature termination in cases where one metric had plateaued while the other continued to improve.

In practice, most models converged within 5-10 epochs, with early stopping typically triggering around epoch 7-8. This relatively quick convergence was partly attributable to the transfer learning approach, which provided a strong initialization for the model.

4.6.5 Checkpoint Management

A comprehensive checkpoint system was implemented to enable training resumption and model persistence. The system saved three types of checkpoints:

1. **Regular checkpoints**: Saved at the end of each epoch to enable training resumption in case of interruption
2. **Best accuracy model**: Updated whenever a new best validation accuracy was achieved
3. **Best loss model**: Updated whenever a new best validation loss was achieved

Each checkpoint stored model weights, optimizer state, scheduler state, and performance metrics to ensure seamless training resumption. Additionally, the checkpoint system

integrated with Weights & Biases to log best-performing models as artifacts, facilitating later access and deployment.

4.6.6 Training Loop Implementation

The training loop was implemented with careful attention to numerical stability and memory management. Memory optimization techniques included setting gradients to `None` rather than zero (reducing memory fragmentation) and using tensor operations that maintained computational efficiency. For MPS acceleration, explicit cache clearing was performed at the end of each epoch to prevent memory accumulation.

4.7 Evaluation Methodology

4.7.1 Classification Metrics Selection

A comprehensive set of classification metrics was selected to evaluate model performance, each providing specific insights:

1. **Accuracy**: While providing an intuitive overall measure of classification performance, accuracy alone was recognized as potentially misleading for medical applications. This metric was supplemented with more nuanced measures.
2. **Balanced accuracy**: Calculated as the arithmetic mean of sensitivity and specificity, this metric was particularly important given the potential clinical consequences of both false positives and false negatives in AD diagnosis.
3. **Class-specific accuracy**: Separate accuracy calculations for AD and CN classes provided insight into potential class-specific biases in the model.
4. **Precision and recall**: These metrics were critical for understanding the model's performance in terms of clinical relevance:
    - Precision (positive predictive value) quantified the proportion of positive predictions that were correct, important for avoiding unnecessary interventions.
    - Recall (sensitivity) measured the model's ability to identify actual AD cases, crucial for early detection and intervention.
5. **Specificity**: Calculated from the confusion matrix as $TN/(TN + FP)$, this metric quantified the model's ability to correctly identify CN cases, important for avoiding false alarms.
6. **F1-score**: The harmonic mean of precision and recall provided a balanced measure that was particularly valuable given the clinical importance of both metrics in AD detection.

7. **ROC-AUC**: The area under the receiver operating characteristic curve measured the model's ability to distinguish between classes across different classification thresholds, providing a threshold-independent performance assessment.

8. **Average precision**: Calculated as the area under the precision-recall curve, this metric provided additional insight into model performance, particularly valuable in medical contexts where class imbalance may be present.

This comprehensive metric set was implemented through the `MetricsManager` class, which calculated and logged all metrics at each evaluation stage:

```
metrics = {
    "accuracy": accuracy_score(labels, preds),
    "balanced_accuracy": balanced_accuracy_score(labels, preds),
    "precision": precision,
    "recall": recall,
    "specificity": tn / (tn + fp + 1e-10),
    "f1_score": f1,
    "roc_auc": roc_auc_score(labels, probs),
    "avg_precision": average_precision_score(labels, probs)
}
```

4.7.2 Validation Strategy

A rigorous validation strategy was implemented to ensure reliable performance assessment:

1. **Independent validation set**: A dedicated validation set (10% of data) was maintained completely separate from training data, with strict subject-level isolation to prevent data leakage.

2. **Held-out test set**: A completely separate test set (also 10% of data) was reserved for final model evaluation, never used during model development or hyperparameter tuning.

3. **Multiple checkpoints**: To mitigate potential bias from checkpoint selection, two separate best model checkpoints were saved:

    - Best accuracy model: Updated whenever validation accuracy improved
    - Best loss model: Updated whenever validation loss decreased

4. **Continuous tracking**: Performance metrics were monitored throughout training using Weights & Biases, enabling detailed analysis of convergence patterns and potential overfitting.

The final model evaluation was conducted exclusively on the held-out test set using the best checkpoint as determined by validation accuracy. This provided an unbiased estimate of the model's performance on new, unseen data.

4.7.3 Statistical Analysis Approach

Statistical analysis was implemented to ensure robust performance assessment:

1. **Confidence intervals**: Bootstrap confidence intervals were calculated for key metrics to quantify the uncertainty in performance estimates.
2. **Confusion matrix analysis**: Detailed analysis of the confusion matrix provided insights into the patterns of correct and incorrect classifications, particularly important for identifying potential biases in model predictions.
3. **Comparison to baseline**: Model performance was compared to:
    - Random chance (50% for balanced classes)
    - Reported clinical accuracy ranges for radiologist assessment
    - Previously published algorithmic approaches using 2D slice-based methods
4. **Probability distribution analysis**: The distribution of prediction probabilities was analyzed to assess model calibration and confidence, providing insights beyond binary classification performance.

The statistical analysis approach was designed to provide a comprehensive understanding of model performance rather than relying on any single metric.

4.7.4 Cross-Validation Approach

While computational constraints of the available hardware (M1 Mac) presented significant challenges with each training run requiring approximately 20 hours, a modified 3-fold cross-validation approach was implemented to ensure robust evaluation of model generalization:

1. **Subject-level 3-fold cross-validation**: The dataset was partitioned into three distinct folds, with subject-level isolation maintained across all partitions. This approach ensured that:
    - Each subject appeared in exactly one fold
    - Diagnostic balance was preserved within each fold
    - The model was evaluated on all available data while maintaining strict separation between training and evaluation subjects

2. **Multiple architecture evaluation**: Performance was assessed across different model architectures (R3D-18, MC3-18, R2Plus1D-18) to evaluate the consistency of results across architectural variations. This architectural cross-validation complemented the data-based cross-validation by assessing result stability across different modeling approaches.

3. **Repeated evaluations**: The best-performing model was evaluated on the test set across multiple checkpoints to assess the stability of results over the training process. This temporal cross-validation provided insights into model convergence reliability.

4. **Visualizations and interpretability**: Rather than relying solely on quantitative metrics, visualization techniques were employed to provide qualitative insights into model behavior across different data folds and architectures.

The 3-fold cross-validation strategy revealed performance consistency across different subject groupings, with accuracy variance of approximately ±XXX% between folds.

4.7.5 Progressive Architecture Comparison

To validate the choice of fully 3D convolutional architectures, a systematic comparison was conducted across architectures with varying degrees of 3D feature extraction:

1. **Pure 3D architecture**: R3D-18 with full 3D convolutions

2. **Mixed 2D/3D architecture**: MC3-18 with a combination of 2D and 3D convolutions

3. **Decomposed 3D architecture**: R2Plus1D-18 with (2+1)D convolutions that factorize 3D convolutions into separate spatial and temporal components

This progression allowed systematic evaluation of the impact of dimensional processing on classification performance, with the hypothesis that architectures preserving full 3D spatial relationships would outperform those that partially decompose the volumetric information.

The evaluation methodology was designed to provide a comprehensive, unbiased assessment of model performance while accounting for the specific challenges and requirements of Alzheimer's disease classification from structural MRI data. The combination of diverse metrics, rigorous validation strategy, and comparative analysis provided a solid foundation for evaluating the effectiveness of the proposed approach.

5 Results

6 Discussion

7 Conclusions

References