

**Multi-Head Encoder (MHEnc)-based Fusion Strategies for Land Cover  
and Land Use Classification Using Sentinel-1 and Sentinel-2**

Taewoong Ham

A Thesis Submitted to  
The Faculty of Graduate Studies  
In Partial Fulfillment of the Requirements  
for the Degree of  
Master of Science

Graduate Program in  
Earth and Space Science and Engineering  
York University  
Toronto, Ontario  
October 2025

© Taewoong Ham, 2025

## Abstract

This study aims to develop a fusion strategy guided by data characteristics to improve LULC classification from multi-source data. The Multi-Head Encoder framework was designed as a backbone, integrating parallel convolutional heads with different dilation rates ( $\delta=1, 2, 3$ ) through learned attention mechanisms to process features across multiple spatial scales—from fine local detail to broad landscape context. Building on this framework, fusion strategies were systematically investigated for heterogeneous sensor combinations and homogeneous temporal scenarios, establishing data-driven guidelines. To understand the underlying mechanisms, Grad-CAM analysis was employed, revealing that Early Fusion can lead to optical dominance over SAR information while Middle Fusion preserves modality-specific contributions. These insights guided the development of a hybrid fusion strategy that achieved 96.83% Overall Accuracy on the spatially independent test area. Patch boundary effects were investigated, and a central-area approach was evaluated, improving performance by up to 13% for attention-based SAR architectures. This integrated framework of multi-scale processing, systematic fusion evaluation, and interpretability analysis enables principled, data-driven classification design.



## Acknowledgments

I am deeply grateful to Professor Baoxin Hu for his supervision, guidance, and encouragement throughout this research. I also thank my committee member, Professor Costas Armenakis, for his valuable comments and suggestions. I am indebted to my lab colleagues for their support and feedback.

This work was supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada through Discovery Grant RGPIN-2021-03624 and Alliance Grant ALLRP 597627-24.

Contains modified Copernicus Sentinel-1 and Sentinel-2 data (2019); © European Union. The 2020 Land Cover of Canada dataset (published 2022) was provided by Natural Resources Canada's Canada Centre for Remote Sensing (NRCan-CCRS). The 2019 Annual Crop Inventory was supplied by Agriculture and Agri-Food Canada (AAFC). Ontario Land Cover (OLC) v1.0 (2024) was obtained from the Ontario Ministry of Natural Resources and Forestry (OMNRF). Processing and analysis employed the Sentinel Application Platform (SNAP) v11.0.0 developed by the European Space Agency (ESA).

## Table of Contents

|                                                                         |             |
|-------------------------------------------------------------------------|-------------|
| <b>ABSTRACT .....</b>                                                   | <b>II</b>   |
| <b>ACKNOWLEDGMENTS.....</b>                                             | <b>III</b>  |
| <b>TABLE OF CONTENTS .....</b>                                          | <b>IV</b>   |
| <b>LIST OF TABLES.....</b>                                              | <b>VII</b>  |
| <b>LIST OF FIGURES.....</b>                                             | <b>VIII</b> |
| <b>CHAPTER 1 INTRODUCTION.....</b>                                      | <b>1</b>    |
| <b>CHAPTER 2 BACKGROUND .....</b>                                       | <b>8</b>    |
| 2.1 CONVOLUTIONAL NEURAL NETWORKS .....                                 | 8           |
| 2.2 FULLY CONVOLUTIONAL NETWORKS .....                                  | 13          |
| 2.3 THE U-NET FOR PIXEL-BASED CLASSIFICATION .....                      | 16          |
| 2.4 U-NET ARCHITECTURE ENHANCEMENTS.....                                | 19          |
| 2.4.1 Multi-Scale Feature Extraction Through Dilated Convolutions ..... | 19          |
| 2.4.2 Multi-Head Architectures for Context Aggregation .....            | 21          |
| 2.4.3 Attention Mechanisms in Convolutional Networks.....               | 22          |
| 2.5 MULTIMODAL DATA FUSION STRATEGIES.....                              | 25          |
| 2.5.1 Early Fusion.....                                                 | 26          |
| 2.5.2 Middle Fusion .....                                               | 27          |
| 2.5.3 Late Fusion .....                                                 | 27          |
| 2.6 BOUNDARY EFFECTS IN CONVOLUTIONAL NETWORKS .....                    | 28          |
| 2.7 INTERPRETABILITY METHODS FOR DEEP LEARNING NETWORKS.....            | 30          |
| 2.7.1 Local Interpretable Model-agnostic Explanations (LIME) .....      | 30          |
| 2.7.2 SHapley Additive exPlanations (SHAP) .....                        | 32          |
| 2.7.3 Class Activation Mapping (CAM).....                               | 34          |

|                                                                        |           |
|------------------------------------------------------------------------|-----------|
| 2.7.4 Gradient-weighted Class Activation Mapping (Grad-CAM).....       | 35        |
| <b>CHAPTER 3 STUDY AREA AND DATASETS .....</b>                         | <b>38</b> |
| 3.1 STUDY AREA AND SOURCE DATA .....                                   | 38        |
| 3.2 REFERENCE DATA GENERATION .....                                    | 43        |
| 3.3 NORMALIZATION OF REMOTELY SENSED DATA .....                        | 49        |
| <b>CHAPTER 4 DEEP LEARNING FRAMEWORK AND EXPERIMENTAL DESIGN .....</b> | <b>51</b> |
| 4.1 MULTI-HEAD ENCODER (MHENC) FRAMEWORK .....                         | 51        |
| 4.2 EXPLORATION OF FUSION STRATEGIES .....                             | 57        |
| 4.3 DESIGN OF THE HYBRID FUSION STRATEGY .....                         | 63        |
| 4.4 EXPERIMENTAL SETUP .....                                           | 64        |
| 4.4.1 Dataset Preparation and Splitting .....                          | 64        |
| 4.4.2 Training Procedures .....                                        | 65        |
| 4.4.3 Edge Effect Investigation .....                                  | 66        |
| 4.5 MODEL EVALUATION .....                                             | 67        |
| 4.5.1 Comparison with existing models .....                            | 67        |
| 4.5.2 Evaluation Metrics .....                                         | 68        |
| <b>CHAPTER 5 EXPERIMENTAL RESULTS.....</b>                             | <b>71</b> |
| 5.1 SINGLE-SOURCE PERFORMANCE OF MHENC AND BASELINES.....              | 71        |
| 5.2 RESULTS FROM DIFFERENT FUSION STRATEGIES .....                     | 76        |
| 5.3 PERFORMANCE OF THE PROPOSED HYBRID FUSION STRATEGY .....           | 83        |
| <b>CHAPTER 6 DISCUSSION .....</b>                                      | <b>88</b> |
| 6.1 ON THE MHENC FRAMEWORK .....                                       | 88        |
| 6.2 DESIGN PRINCIPLES FOR MULTI-SOURCE DATA FUSION.....                | 89        |
| 6.3 HYBRID FUSION DESIGN.....                                          | 91        |

|                                     |            |
|-------------------------------------|------------|
| 6.4 EDGE EFFECT INVESTIGATION ..... | 95         |
| <b>CHAPTER 7 CONCLUSION.....</b>    | <b>97</b>  |
| <b>REFERENCES .....</b>             | <b>100</b> |
| <b>APPENDIX A.....</b>              | <b>108</b> |

## List of Tables

|                                                                                                                                          |    |
|------------------------------------------------------------------------------------------------------------------------------------------|----|
| TABLE 3.1. SENTINEL-2 MSI BANDS USED IN THIS STUDY AND THEIR PROPERTIES.....                                                             | 39 |
| TABLE 3.2. DESCRIPTION OF THE SEVEN TARGET LULC CLASSES USED IN THIS STUDY. ....                                                         | 43 |
| TABLE 3.3. CLASS DISTRIBUTION OF THE TRAINING/VALIDATION REFERENCE MAP AFTER CONSENSUS<br>LABELING AND CHM FILTERING.....                | 47 |
| TABLE 4.1. CLASS DISTRIBUTION BETWEEN THE TRAINING AND VALIDATION SETS. ....                                                             | 65 |
| TABLE 5.1. SINGLE-SOURCE PERFORMANCE ON THE TEST AREA .....                                                                              | 71 |
| TABLE 5.2. CLASS-WISE F1-SCORES (%) ON THE TEST AREA FOR SINGLE-SOURCE IMAGERY. ....                                                     | 73 |
| TABLE 5.3. LEARNED ATTENTION WEIGHTS FOR THE PARALLEL MULTI-SCALE HEADS.....                                                             | 75 |
| TABLE 5.4. PERFORMANCE OF FUSION STRATEGIES ON THE TEST AREA. ....                                                                       | 76 |
| TABLE 5.5. CLASS-WISE F1-SCORES (%) FOR ALL FUSION STRATEGIES ON THE TEST AREA. ....                                                     | 78 |
| TABLE 5.6. CONFUSION MATRIX OF THE PROPOSED HYBRID FUSION STRATEGY ON THE TEST AREA (PIXEL<br>COUNTS). ....                              | 84 |
| TABLE 6.1. PERFORMANCE COMPARISON OF FUSION STRATEGIES COMBINING THREE DATA SOURCES<br>(S2-AUG + S2-OCT + S1-AUG) ON THE TEST AREA. .... | 93 |
| TABLE 6.2. CLASS-WISE F1-SCORES (%) FOR THREE-SOURCE FUSION STRATEGIES. ....                                                             | 94 |
| TABLE 6.3. EFFECT OF EDGE-PIXEL EXCLUSION ON MODEL PERFORMANCE (mIoU, %). ....                                                           | 95 |

## List of Figures

|                                                                                                                                                                                                                         |    |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----|
| FIGURE 2.1. AN ILLUSTRATIVE DIAGRAM OF A CNN ARCHITECTURE. ....                                                                                                                                                         | 9  |
| FIGURE 2.2 FCN ARCHITECTURE. ....                                                                                                                                                                                       | 14 |
| FIGURE 2.3. THE U-NET ARCHITECTURE. ....                                                                                                                                                                                | 17 |
| FIGURE 2.4. COMPARISON OF DILATED CONVOLUTION OPERATIONS. ....                                                                                                                                                          | 20 |
| FIGURE 2.5. PATCH-BASED PROCESSING CHALLENGES AND PADDING STRATEGIES. ....                                                                                                                                              | 29 |
| FIGURE 3.1. THE TRAINING/VALIDATION AREA (OUTLINED IN BLUE) AND THE TESTING AREA (OUTLINED<br>IN PURPLE) ARE SHOWN WITHIN THE GREAT CLAY BELT REGION.....                                                               | 38 |
| FIGURE 3.2. VISUAL REPRESENTATION OF THE REMOTELY SENSED IMAGERY OVER A REPRESENTATIVE<br>SUB-AREA WITHIN THE TEST AREA.....                                                                                            | 41 |
| FIGURE 3.3. COMPARISON MAP AMONG THE THREE RECLASSIFIED DATASETS IN THE TRAINING AREA. ....                                                                                                                             | 45 |
| FIGURE 3.4. PRE-NORMALIZATION PIXEL-VALUE DISTRIBUTIONS FOR S2-AUG IN THE TRAINING AND<br>VALIDATION AREA (COLORED) AND THE TEST AREA (GREY).....                                                                       | 49 |
| FIGURE 4.1. MHENC ARCHITECTURE OVERVIEW. ....                                                                                                                                                                           | 51 |
| FIGURE 4.2. BASICUNET ARCHITECTURE.....                                                                                                                                                                                 | 52 |
| FIGURE 4.3. MULTI-HEAD BLOCK ARCHITECTURE. ....                                                                                                                                                                         | 53 |
| FIGURE 4.4. ATTENTION-WEIGHTED FUSION MECHANISM. ....                                                                                                                                                                   | 55 |
| FIGURE 4.5. DIAGRAM OF EARLY FUSION. ....                                                                                                                                                                               | 59 |
| FIGURE 4.6. DIAGRAMS OF MIDDLE FUSION STRATEGIES: (A) SYMMETRIC MIDDLE FUSION; (B)<br>CONVOLUTIONAL MIDDLE FUSION.....                                                                                                  | 60 |
| FIGURE 4.7. DIAGRAMS OF LATE FUSION VARIANTS: (A) SIMPLE-AVERAGE LATE FUSION, WHERE OUTPUT<br>LOGITS ARE AVERAGED; (B) LEARNABLE-WEIGHTED LATE FUSION, WHERE OUTPUT LOGITS ARE<br>COMBINED USING LEARNABLE WEIGHTS..... | 62 |
| FIGURE 4.8. HYBRID FUSION ARCHITECTURE.....                                                                                                                                                                             | 63 |
| FIGURE 5.1. COMPARISON OF SINGLE-SOURCE PERFORMANCE (MIOU) ON THE TEST AREA.....                                                                                                                                        | 72 |

|                                                                                     |    |
|-------------------------------------------------------------------------------------|----|
| FIGURE 5.2. PERFORMANCE OF HETEROGENEOUS FUSION STRATEGIES .....                    | 77 |
| FIGURE 5.3. PERFORMANCE OF HOMOGENEOUS FUSION STRATEGIES.....                       | 78 |
| FIGURE 5.4. HETEROGENEOUS SENSOR FUSION ANALYSIS FOR THE URBAN/DEVELOPED CLASS..... | 80 |
| FIGURE 5.5. HETEROGENEOUS SENSOR FUSION ANALYSIS FOR THE WETLAND CLASS.....         | 81 |
| FIGURE 5.6. HOMOGENEOUS TEMPORAL FUSION ANALYSIS FOR THE MIXED FOREST CLASS.....    | 82 |
| FIGURE 5.7. VISUAL COMPARISON ACROSS THE TEST AREA.....                             | 86 |
| FIGURE 6.1. PERFORMANCE COMPARISON OF THREE-SOURCE FUSION STRATEGIES. ....          | 93 |

# Chapter 1 Introduction

Land use/land cover (LULC) maps underpin environmental management, sustainable resource planning, and disaster monitoring (Zhang and Li, 2022). Satellite remote sensing enables consistent, large-area mapping (Hermosilla et al., 2022), yet pixel-wise classification remains difficult due to within-class variability, between-class similarity, and the inherently multi-scale character of LULC patterns. Within this study, this "multi-scale" challenge refers not to variations in input image resolution, but to the need to account for the diverse spatial contexts and scales of ground features themselves; for example, models simultaneously recognize small, detailed objects (e.g., isolated buildings) and large, homogeneous regions (e.g., extensive forests). Addressing these challenges requires models that can both integrate complementary information across sensors and reason effectively over these multiple spatial scales.

Multi-source remote sensing imagery reduces spectral and temporal limitations of single sensors. Combining Sentinel-2 (S2) optical reflectance with Sentinel-1 (S1) Synthetic Aperture Radar (SAR) backscatter has improved LULC mapping compared to using either source alone (Steinhausen et al., 2018; Mahdianpari et al., 2019; Tavares et al., 2019; Solórzano et al., 2021; Hu et al., 2021). However, integrating different sensors presents challenges. Specifically, heterogeneous sensors have different measurement physics and produce data with distinct statistical distributions. In contrast, homogeneous inputs, such as temporal Sentinel-2 imagery, capture phenological changes between acquisition dates (Ghamisi et al., 2019; Li et al., 2022). To address these differing characteristics, fusion methods were therefore systematically tested for both heterogeneous (S1 and S2) and homogeneous (S2-Aug and S2-Oct) combinations within a consistent deep-learning framework. In contrast to other modalities such as hyperspectral or LiDAR—which were excluded primarily due to their limited availability and higher acquisition



costs—this study focused exclusively on freely accessible Sentinel-1 and Sentinel-2 data. Finally, a convolutional-network-based segmentation framework was adopted, as it has been widely demonstrated to perform effectively for pixel-level land-cover classification (Ma et al., 2019; Yuan et al., 2021).

Encoder-decoder architectures progressively downsample input features to extract semantic information (encoding) and then upsample to restore spatial resolution (decoding). U-Net exemplifies this design and is widely used for LULC mapping because it balances semantic abstraction with spatial detail (Ma et al., 2019; Yuan et al., 2021; Minaee et al., 2022). Recent developments have extended this encoder–decoder paradigm beyond convolutional operations, where self-attention mechanisms are used to model long-range spatial dependencies more explicitly. Vision Transformers show promise but require larger labeled datasets and more computational resources (Dosovitskiy et al., 2021; Khan et al., 2024). Therefore, convolutional networks were selected for this study. A compact U-Net (BasicUNet) was used as the reference model to enable controlled comparisons across experiments. U-Net (Ronneberger et al., 2015) has a limitation in capturing features at multiple spatial scales: fixed  $3\times 3$  kernels create narrow receptive fields at each layer, making it difficult to simultaneously capture both broad spatial context needed for large homogeneous areas (such as extensive conifer stands) and fine spatial detail required for small features (such as narrow watercourses or isolated buildings) (Luo et al., 2016).

To improve encoder-decoder performance, multiple architectural enhancements have been proposed. Some emphasize multi-scale context capture, including atrous spatial pyramid pooling (ASPP) in DeepLab (Chen et al., 2018) and the pyramid pooling module in Pyramid Scene Parsing Network (PSPNet) (Zhao et al., 2017). Others emphasize feature selection through

attention mechanisms, such as the Convolutional Block Attention Module (CBAM) (Woo et al., 2018). These approaches typically combine features from parallel branches (e.g., different dilation rates or pooling bins) via learned mixing layers (often  $1 \times 1$  convolutions) and/or apply learned channel/spatial reweighting before classification. While these designs improve accuracy, they generally fix the set of scales a priori and use mixing functions that are shared across spatial locations. CBAM's channel attention is largely global (derived from pooled statistics), and its spatial gate does not explicitly govern trade-offs among dilation scales. In settings where the optimal balance between fine, intermediate, and coarse context varies across datasets or regions, a lightweight module that makes the scale trade-off explicit and learnable remains desirable. These considerations point to the need for a compact design that learns to mix parallel dilation scales from data, rather than fixing scale sets and global mixing a priori.

Accurate LULC classification depends not only on architectural design but also on how models are trained on imagery patches. Patch-based processing creates boundary artifacts when convolutional receptive fields extend beyond patch edges and require padding (zero, reflection, or replication). The padding introduces artificial patterns not present in the original imagery, particularly near boundaries. These artificial values affect feature extraction in early layers and produce incorrect gradients for edge pixels during backpropagation. To address this effect, two training configurations were compared: full  $25 \times 25$  patch training versus central  $20 \times 20$  region training that excludes boundary pixels most affected by padding.

Prior to the emergence of deep learning, multimodal data fusion was generally performed using traditional statistical or machine-learning techniques. Pixel-, feature-, and decision-level fusion schemes were developed to combine complementary information from different sensors; however, these methods typically relied on manual feature design and linear modeling

assumptions, limiting their ability to capture complex cross-modal relationships (Ghassemian, 2016; Jin et al., 2022; Vivone et al., 2015). The limitations of such conventional approaches have motivated the transition toward end-to-end deep learning frameworks, which can automatically learn spatial and cross-modal representations directly from the data.

Within these deep learning frameworks, the integration of multi-source data such as Optical reflectance (S2) and SAR backscatter (S1) provide complementary information for LULC mapping under different atmospheric and illumination conditions. The integration of such multi-source data is commonly approached at three distinct stages: Early (input-level), Middle (feature-level), or Late (decision-level) (Ghamisi et al., 2019; Li et al., 2022; Mena et al., 2024). Data characteristics determine the optimal strategy, particularly whether modalities are similar (homogeneous) or different (heterogeneous) and their relative contributions (Li et al., 2022; Mena et al., 2024). For heterogeneous combinations like S2 reflectance with S1 backscatter, separate early processing (Middle or Late Fusion) allows modality-specific representations to be learned before integration (Zhang and Xu, 2018). Performance increases have been reported from S1+S2 integration; for example, Overall Accuracy increased from 89.53% (S2 only) to 91.07% with S1+S2 in urban mapping (Tavares et al., 2019) and reached 92.12% using combined S1+S2 with a composite-kernel SVM (Hu et al., 2021). Direct comparison remains difficult because studies use different backbones, training protocols, and test data; controlled comparisons of fusion levels based on data heterogeneity for Sentinel-based LULC mapping are limited.

Beyond the challenge of optimal fusion design, the interpretability of multi-source fusion poses an additional challenge. It remains unclear how each source contributes to predictions and whether specific fusion strategies exploit complementarity, particularly for heterogeneous

combinations such as optical and SAR within deep architectures. Model-agnostic methods such as Local Interpretable Model-agnostic Explanations (LIME) (Ribeiro et al., 2016) and SHapley Additive exPlanations (SHAP) (Lundberg and Lee, 2017) estimate feature importance by perturbing inputs, but they require numerous forward passes and are computationally intensive for imagery with large spatial dimensions (i.e., high pixel counts). Class Activation Mapping (CAM) (Zhou et al., 2016) localizes discriminative regions by using global average pooling and class-specific weights, but it requires architectural constraints. Gradient-weighted Class Activation Mapping (Grad-CAM) (Selvaraju et al., 2017) overcomes this limitation by using gradients flowing into a selected convolutional layer to produce localization maps without architectural modification. Because Grad-CAM can be applied to different branches and layers, it is well suited to analyzing contribution patterns in multi-source fusion networks and to providing empirical evidence of complementarity versus dominance between modalities.

To address the challenges above, this thesis was designed to develop effective fusion methods for multi-source remote sensing in LULC classification, with an emphasis on multi-scale feature processing and integration of heterogeneous data. The specific objectives and contributions are as follows:

(1) To develop a multi-scale feature processing framework, the Multi-Head Encoder (MHEnc). The framework integrates parallel multi-scale convolutional heads with an attention-based mechanism that learns weights for fusing multi-scale features. Its performance was evaluated against BasicUNet and CBAMUNet, and it showed improved accuracy for LULC classification.

(2) Patch-boundary effects were investigated by evaluating a training strategy that mitigates edge artifacts. Loss computation was confined to the central region of each patch, and

the influence of this decision on model capability was quantified. This setting provides more reliable performance assessment for the architectural and fusion strategies examined.

(3) To explore and evaluate fusion strategies (Early, Middle, and Late Fusion) based on data characteristics. This systematic investigation aims to develop empirical guidelines for matching fusion architectures to data heterogeneity (i.e., heterogeneous S1-S2 sensor fusion versus homogeneous S2 temporal fusion). Building on these principles, a Hybrid Fusion Strategy is designed and assessed. This strategy employs a multi-stage, or 'hybrid,' approach: it first applies an optimal fusion strategy for the homogeneous temporal data (S2-Aug and S2-Oct) before integrating the resulting features with the heterogeneous SAR data (S1-Aug) at a subsequent stage. This design aims to improve classification accuracy by aligning the fusion architecture with the specific characteristics of the data at each integration step.

(4) To interpret and validate the internal mechanisms of data fusion using Grad-CAM. This analysis is intended to move beyond overall accuracy by visualizing how different strategies integrate sources. The objective is to use this approach to investigate modality-specific contributions and examine potential phenomena such as feature dominance or suppression, thereby providing empirical support for the selection of fusion strategies in complex environments.

The remainder of this thesis is organized as follows. Chapter 2 provides the Background and literature review relevant to LULC classification, deep learning, and data fusion. The Methodology is described across two chapters: Chapter 3 details the study area and data used, while Chapter 4 presents the proposed MHEnc framework and experimental design. The Experimental Results are reported in Chapter 5. These findings are analyzed and interpreted in

the Discussion in Chapter 6. Finally, Chapter 7 provides the Conclusions of this research, summarizing the contributions and suggesting directions for future work.

## Chapter 2 Background

This chapter provides an overview of the deep-learning concepts and multi-modal methods foundational to this study's LULC classification framework. The review begins with the fundamental principles of Convolutional Neural Networks (CNNs) and their evolution into Fully Convolutional Networks (FCNs) for semantic segmentation. The core U-Net architecture is then detailed, followed by key enhancements, including dilated convolutions and attention mechanisms. Subsequently, common multimodal data fusion strategies (Early, Middle, and Late Fusion) are examined. The chapter concludes by reviewing the challenges of patch boundary effects in convolutional processing and the interpretability methods, such as Grad-CAM, used for model analysis.

### 2.1 Convolutional Neural Networks

In this section, the components of convolutional networks that form the basis of the models applied in this study are introduced. Hierarchical patterns are extracted from multispectral and SAR imagery by CNNs through the application of stacked convolutions and related operations.

As shown in Figure 2.1, a traditional CNN architecture is illustrated, such as one that might be applied to LULC classification using a patch-based (or "block window") approach. The input is an image patch, illustratively sized at  $256 \times 256$  pixels, with 10 channels representing multispectral data (e.g., from a sensor like Sentinel-2). In this classification scheme, the network processes the entire patch to produce a single classification label for that block. The spatial dimensions are progressively reduced and the number of feature channels is increased by three convolutional blocks. The spatial resolution is maintained by the first block, and 32 feature maps are produced. After max pooling halves the spatial dimensions, 64 channels are produced by the

second block. This pattern is continued by the third block, and 128 feature maps at  $64 \times 64$  resolution are produced. Global average pooling (GAP) is applied to condense the feature maps into a compact representation, which is then passed to fully connected layers for final classification.

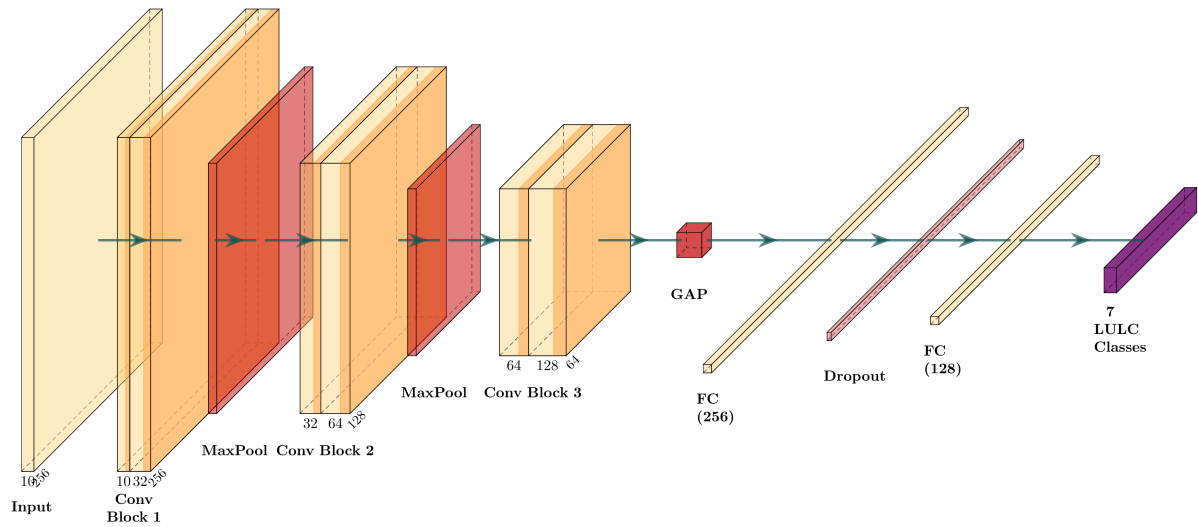


Figure 2.1. An illustrative diagram of a CNN architecture. The model processes a 10-channel 256x256 input through three convolutional (Conv) blocks, where spatial dimensions are reduced via Max Pooling and feature channels are increased (32, 64, 128). Global Average Pooling (GAP) and Fully Connected (FC) layers transform the feature maps into a final 7-class LULC prediction. Note: This diagram is for exposition; patch sizes used in this study are different.

A small set of fundamental layer types is used in CNNs to process features at different scales. Learned filters are applied by convolutional layers to extract spatial patterns across the input. The spatial dimensions are reduced by max pooling layers by selecting the maximum value within local neighborhoods, which improves computational efficiency and provides limited translation invariance. Extracted features are aggregated into class predictions by fully connected layers. Nonlinear activation functions, such as the rectified linear unit (ReLU), are applied between layers, thereby enabling complex decision boundaries.



Convolution is regarded as the core operation by which pattern detection across spatial locations is enabled. For a two-dimensional input, the discrete convolution with a kernel is expressed in Equation (2.1):

$$(f * k)(i, j) = \sum_m \sum_n f(i - m, j - n) \cdot k(m, n) \quad (2.1)$$

For each output position  $(i, j)$ , a weighted sum over a local neighborhood is computed by the operation. For a  $3 \times 3$  kernel, the indices  $m$  and  $n$  are set to range from  $-1$  to  $1$ , while for a  $5 \times 5$  kernel, they would range from  $-2$  to  $2$ . During training, the kernel parameters are learned, allowing different kernels to encode patterns (including edges, spectral contrasts, and cross-channel interactions) that support land-cover discrimination.

Hierarchical representations are formed in CNNs through successive layers of abstraction. Simple patterns, such as edges and boundaries, are captured by early layers. These primitives are combined into textures and spatial configurations by deeper layers, and broader structure and context are represented at the deepest layers. This progression is facilitated by decreasing spatial resolution and increasing channel depth across layers, thereby enabling the representation of both local details and global patterns required for accurate classification.

Spectral information beyond the visible range is provided by satellite imagery. Whereas three visible channels (red, green, blue) are typically captured by consumer cameras, 10 or more channels spanning from the visible to the shortwave infrared are provided by sensors such as S2. These channels reveal phenomena that are often not discernible by the human eye, such as the early onset of vegetation water stress, subtle soil moisture differences, or mineralogical variations.

Millions of parameters are optimized during CNN training to minimize a loss function. For multi-class LULC mapping, the pixel-wise cross-entropy loss shown in Equation (2.2) is commonly used:

$$L_{CE} = -\frac{1}{H \times W} \sum_{i=1}^H \sum_{j=1}^W \sum_{c=1}^C y_{i,j,c} \log(\hat{y}_{i,j,c}) \quad (2.2)$$

In Equation (2.2),  $H$  and  $W$  are the spatial dimensions of the output map,  $C$  is the number of classes,  $y_{i,j,c}$  is set to 1 if the pixel at location  $(i, j)$  belongs to class  $c$  and to 0 otherwise, and  $\hat{y}_{i,j,c}$  is the predicted probability for class  $c$  at that pixel.

Class imbalance and boundary ambiguity are commonly encountered in LULC datasets. Far fewer pixels may be assigned to urban areas than to forests, and class boundaries are often composed of mixed pixels. To address these issues, alternative or complementary loss functions are used. Easy examples are down-weighted and hard ones are emphasized by Focal Loss, as formulated in Equation (2.3) (Lin et al., 2017):

$$L_{focal} = -\alpha_t(1 - p_t)^\gamma \log(p_t) \quad (2.3)$$

where  $p_t$  is the predicted probability for the true class,  $\gamma$  is a focusing parameter that controls the rate of down-weighting, and  $\alpha_t$  is a class-weighting factor. The overlap between predictions and validation is evaluated by Dice Loss, which is effective for segmentation tasks with class imbalance, as shown in Equation (2.4) (Milletari et al., 2016):

$$L_{\text{dice}} = 1 - \frac{2 \sum_i p_i g_i + \epsilon}{\sum_i p_i + \sum_i g_i + \epsilon} \quad (2.4)$$

where  $p_i$  and  $g_i$  are the predicted and reference values for pixel  $i$ , and  $\epsilon$  is a small constant for numerical stability. In practice, these losses are often combined as in Equation (2.5):

$$L_{\text{total}} = L_{\text{focal}} + L_{\text{dice}} \quad (2.5)$$

A balance between pixel-wise accuracy and region-level overlap is thereby achieved by this combination.

Optimization is performed through backpropagation, by which gradients of the loss with respect to all network parameters are computed. Starting from the final loss value, the chain rule of calculus is used to compute each parameter's contribution to the error, with propagation proceeding backward through the network. For convolutional layers, gradients are obtained for both the kernel weights and the input feature maps; kernel gradients drive weight updates, whereas feature-map gradients are propagated to earlier layers. Parameters are then updated by an optimizer. In basic stochastic gradient descent (SGD), a single learning rate is used for all parameters, yielding the same update magnitude regardless of gradient characteristics. In Adam (Adaptive Moment Estimation), per-parameter learning rates are adapted by tracking the gradient history for each parameter. Running averages of both the gradients and their squared values (first and second moments) are maintained. Through this parameter-specific adaptation, updates are shrunk for parameters with consistently large gradients (mitigating overshooting) and are enlarged for those with small gradients to ensure adequate updates. The application of weight decay regularization is modified in AdamW. In Adam, weight decay is applied before adaptive

learning rates are computed, leading to variation in the regularization effect. In AdamW, this decoupling is achieved by applying the decay directly to the parameters after the gradient step, which helps maintain a consistent regularization effect that is independent of gradient magnitudes.

## 2.2 Fully Convolutional Networks

In early deep-learning architectures, a single label per image was produced. By contrast, a label for every pixel is required by LULC mapping, a task referred to as pixel-wise classification (semantic segmentation). A straightforward adaptation was implemented with a sliding-window process: for each pixel, a patch centered on that pixel was classified, and the resulting class was assigned to the center. For an image of size  $H \times W$  with  $C$  channels,  $H \times W$  forward passes were required by this approach, each on a patch of size  $P \times P \times C$ . Significant computational expense and redundancy were incurred, because heavy overlap among adjacent patches prevented reuse of shared computations.

Dense prediction in a single forward pass is enabled by Fully Convolutional Networks (FCNs) (Long et al., 2015). In traditional image classifiers, fully connected layers are placed at the end; fixed-size inputs are required by these layers, and spatial arrangement is discarded through flattening of features into a one-dimensional vector. In FCNs, these fully connected layers are removed and replaced with convolutional layers that preserve spatial structure, and class-score maps are produced instead of a single vector of class scores.

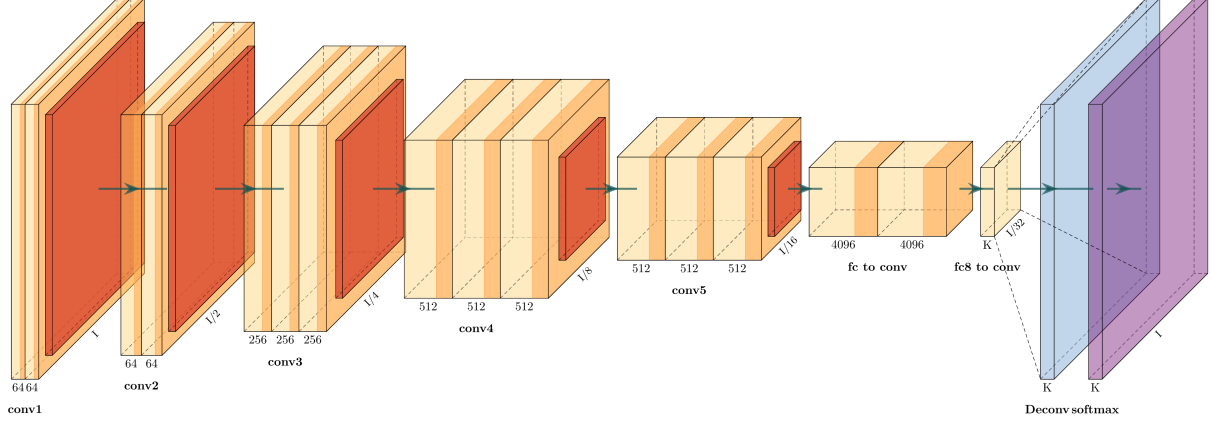


Figure 2.2 FCN architecture.

As shown in Figure 2.2, increasingly abstract features are extracted by convolutional blocks (conv1-conv5) while spatial resolution is reduced through pooling. Instead of flattening features and using fully connected layers, additional convolutions (“fc-to-conv”) that maintain spatial structure while performing the same computational role are applied in FCNs. Through this modification, images of arbitrary size can be processed and spatially dense predictions can be produced efficiently. Furthermore, when the whole image is processed in a single pass by FCNs, overlapping regions are not recomputed, so redundant work is avoided (Long et al., 2015).

Formally, an input image  $\mathbf{X} \in \mathbb{R}^{H \times W \times C}$  is mapped by the network to a spatial map of predictions  $\mathbf{Y} \in \mathbb{R}^{H \times W \times K}$ , as expressed in Equation (2.6):

$$\mathbf{Y} = f_{\text{conv}}(\mathbf{X}; \boldsymbol{\theta}) \quad (2.6)$$

where the output dimensions  $H \times W$  are designed to match the input spatial dimensions,  $K$  is the number of classes, and  $\boldsymbol{\theta}$  is the parameter set.

Within the FCN framework, spatial resolution is progressively reduced by the encoder through pooling. Through this compression, high-level semantic information is captured while precise spatial information is reduced. To obtain pixel-wise predictions from these compressed features, spatial dimensions must be restored. This objective is achieved in FCNs by a decoder path in which the resulting low-resolution, class-conditional maps are upsampled by learnable transposed convolutions.

Learnable upsampling that inverts the spatial reduction produced by strided convolutions or pooling is provided by transposed convolution. For an upsampling factor  $s$ , a transposed convolution is formulated in Equation (2.7):

$$y_{i,j} = \sum_{m,n} x_{\lfloor i/s \rfloor + m, \lfloor j/s \rfloor + n} \cdot k_{m,n} \quad (2.7)$$

In Equation (2.7), the low-resolution feature map is denoted by  $x$ , the upsampled output by  $y$ , the learnable kernel by  $k$ , the upsampling factor (stride) is denoted by  $s$ , and the floor operation by  $\lfloor \cdot \rfloor$ . Unlike fixed interpolation methods, the kernel parameters are learned so that high-resolution features are reconstructed from compressed representations.

Spatial resolution is progressively restored by the decoder through multiple upsampling stages. If a total reduction factor is introduced by the encoder, this reduction is reversed by the decoder. Skip connections can be used to combine upsampled features with features from earlier layers, thereby providing high-resolution details that complement deeper semantic features. Therefore, FCN outputs are shaped by a balance between semantic information (gained from large receptive fields) and spatial precision (gained from high-resolution features). Motivation

for encoder-decoder architectures that preserve and reconstruct spatial information more explicitly for pixel-wise classification has been provided by this trade-off.

## **2.3 The U-Net for Pixel-Based Classification**

It was demonstrated by FCNs that dense predictions can be produced through learnable upsampling by convolutional networks. Building on this foundation, ways to preserve spatial information while extracting semantic features were explored by subsequent architectures. A symmetric encoder-decoder structure connected by skip connections is adopted in U-Net (Ronneberger et al., 2015). Originally developed for biomedical image segmentation, the architecture has been widely adopted for remote sensing pixel-wise classification.

As shown in Figure 2.3, spatial dimensions are progressively reduced in the encoder path while channel depth is increased. The first encoder block is configured to produce 64 feature channels without changing spatial resolution; the input-channel count is determined by the data. For each subsequent encoder block, the channel count is doubled (from 64 to 128, then to 256 and 512), while the spatial dimensions are halved through  $2 \times 2$  max pooling. The spatial resolution is decreased from the original size to  $1/2$ ,  $1/4$ ,  $1/8$ , and finally  $1/16$  at the deepest layer, where 1024 channels are present. This layer serves as the "bottleneck" of the architecture, as it is the point of maximum spatial compression ( $1/16$  resolution) before the network path expands in the decoder.

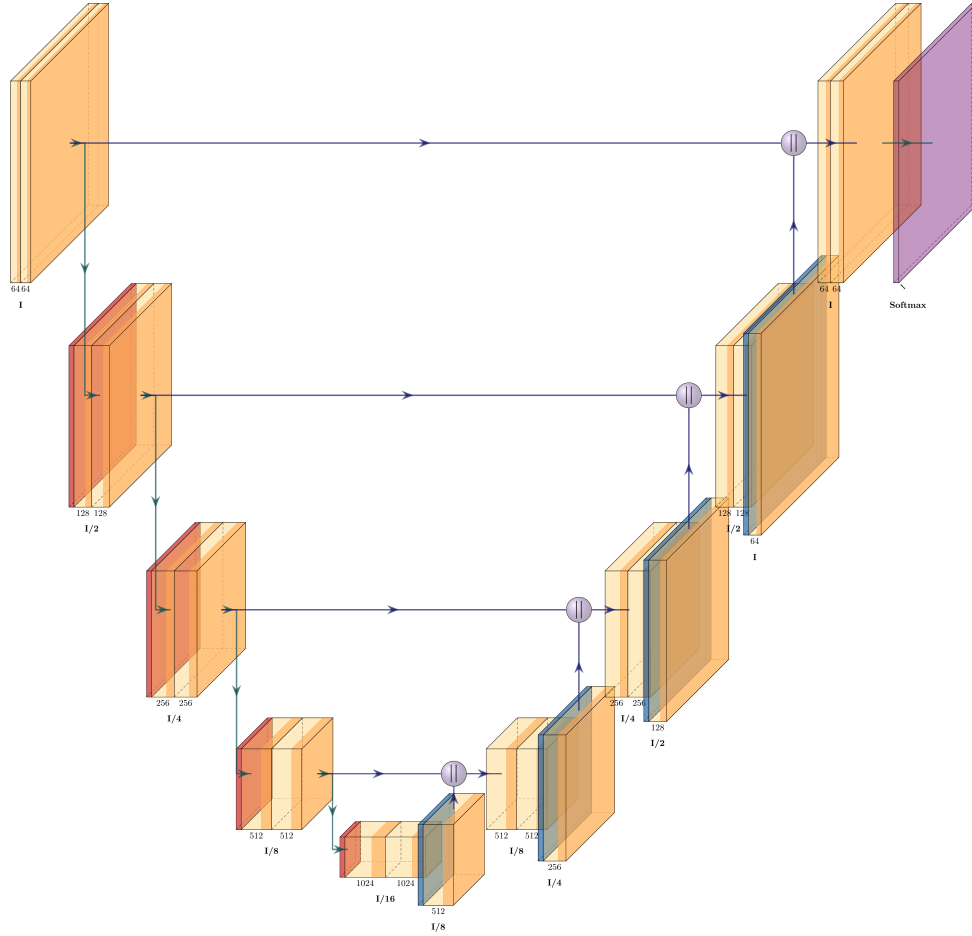


Figure 2.3. The U-Net architecture. The model consists of a symmetric encoder path (left side, shaded orange/red) that progressively downsamples spatial resolution while increasing feature channels, and a decoder path (right side, shaded blue/orange) that upsamples features to restore resolution. Skip connections (horizontal arrows) pass high-resolution features from the encoder to the decoder, where they are concatenated (grey circles) with the upsampled features. This process allows the network to combine deep semantic information with shallow spatial detail. The final layer (purple) produces the pixel-wise probability map.

Spatial resolution is progressively restored by the decoder path. At the beginning of each decoder block, an upsampling step that doubles the spatial dimensions is applied. Features are transferred from the encoder to the corresponding decoder level by skip connections; these encoder features are concatenated with the upsampled decoder features, and the channel count is doubled at each concatenation. For example, when 512 upsampled channels are concatenated



with 512 encoder channels, a total of 1024 channels is obtained. Within each decoder block, subsequent convolutions are used to reduce these concatenated channels by half: from 1024 to 512, 512 to 256, and so forth. In the final layer, a  $1 \times 1$  convolution is applied. This final convolution acts as a pixel-wise classifier, mapping the 64-channel feature vector at each spatial location to a vector with a depth equal to the number of output classes. These raw, unnormalized output values are known as logits. To convert these logits into meaningful class probabilities, a softmax activation function is applied. The softmax function normalizes the logits for each pixel independently, producing a probability distribution (where all class probabilities for that pixel sum to 1.0) that indicates the likelihood of each class.

Operation through concatenation is employed by the skip connections, as expressed in Equation (2.8), which shows the decoder feature computation at each level:

$$x_{dec}^l = g(W_{dec}^l * \text{Concat}[x_{enc}^l, \text{Up}(x_{dec}^{l-1})] + b_{dec}^l) \quad (2.8)$$

where the encoder features at level  $l$  are denoted by  $x_{enc}^l$ , the decoder features at level  $l$  are denoted by  $x_{dec}^l$ , the decoder features from the previous level are denoted by  $x_{dec}^{l-1}$ , a  $2 \times 2$  transposed convolution is denoted by  $\text{Up}(\cdot)$ , concatenation along the channel dimension is denoted by  $\text{Concat}[\cdot, \cdot]$ , the convolutional parameters at level  $l$  are denoted by  $W_{dec}^l$  and  $b_{dec}^l$ , and the activation function is denoted by  $g(\cdot)$ .

In the final layer, a  $1 \times 1$  convolution is applied to produce class predictions, as shown in Equation (2.9), which yields pixel-wise class probabilities:

$$y_{i,j} = \text{softmax}(W_{final} * x_{final,i,j} + b_{final}) \quad (2.9)$$

where the feature map from the final decoder stage is denoted by  $x_{final}$ , which has  $F$  channels (e.g.,  $F=64$  in Figure 2.3), and  $x_{final,i,j}$  is the feature vector at spatial position  $(i, j)$ . The mapping from these  $F$  feature channels to  $K$  classes is performed by the  $1 \times 1$  convolution's weights  $W_{final} \in \mathbb{R}^{K \times F}$ . Class-specific biases are provided by  $b_{final} \in \mathbb{R}^K$ , and normalization to class probabilities is performed by the softmax function.

At each pooling operation, the receptive field is increased by a factor of 2. After four pooling operations, the receptive field of a bottleneck-layer feature is extended to cover a large portion of the input. Broader context and relationships between distant regions needed for LULC classification are thereby captured by this expanded field.

Parameters are initialized randomly and optimized through gradient descent. Input images are passed through the encoder, bottleneck, and decoder to produce predictions. The loss is computed against validation labels. Gradients are backpropagated through the network, including skip connections, and parameters are updated.

Fixed  $3 \times 3$  kernels are used throughout U-Net. The effective receptive field is increased with depth through successive operations, but features at each layer are processed at a fixed scale. Multi-scale patterns in the input are therefore captured with limited flexibility.

## 2.4 U-Net Architecture Enhancements

A strong foundation for pixel-wise classification tasks is provided by the U-Net architecture. Two enhancement strategies are presented in this section: dilated convolutions for expanded spatial context and attention mechanisms for adaptive feature selection.

### 2.4.1 Multi-Scale Feature Extraction Through Dilated Convolutions

When fixed  $3 \times 3$  kernels are used within each layer, a static, per-layer receptive field is obtained. Consequently, simultaneous capture of diverse spatial scales is made challenging. This

limitation is important for LULC classification, in which features span multiple spatial contexts and extents.

As shown in Figure 2.4, contiguous kernel elements are used in standard convolution. A  $3 \times 3$  kernel is used to sample a  $3 \times 3$  neighborhood, thereby limiting the spatial context captured in a single operation. Spacing between kernel elements is introduced by dilated (atrous) convolution to expand the sampling area without increasing the number of parameters (Yu and Koltun, 2016).

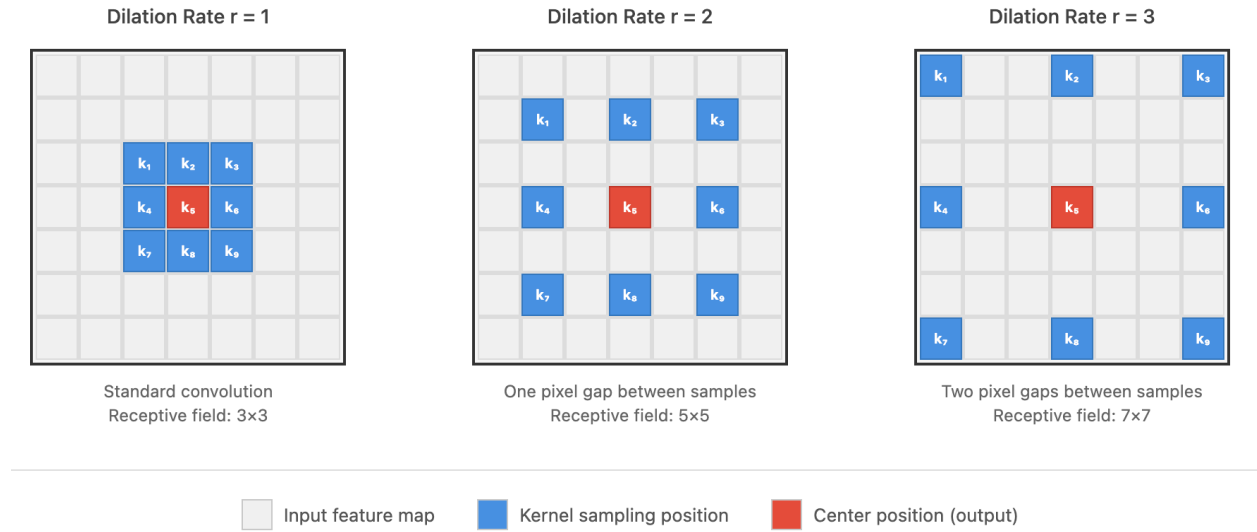


Figure 2.4. Comparison of dilated convolution operations at dilation rates  $\delta = 1, 2$ , and  $3$ .

The operation is parameterized by a dilation rate  $\delta$ , as defined in Equation (2.10):

$$(f * k)(i, j) = \sum_m \sum_n f(i - \delta \cdot m, j - \delta \cdot n) \cdot k(m, n) \quad (2.10)$$

where the input feature map is denoted by  $f$ , the convolutional kernel is denoted by  $k$ , the dilation rate is denoted by  $\delta$ , and the output position is denoted by  $(i, j)$ . For a  $3 \times 3$  kernel, the indices  $m$  and  $n$  are set to range from -1 to 1. The spacing between sampled pixels is controlled by the dilation rate. When  $\delta = 1$ , standard convolution is recovered. When  $\delta = 2$ , a one-pixel gap is introduced between sampled positions, which produces an effective receptive field of  $5 \times 5$  while maintaining 9 learnable parameters.

#### 2.4.2 Multi-Head Architectures for Context Aggregation

To leverage multiple receptive fields simultaneously, a common architectural pattern is the multi-head design (Yuan et al., 2021). In this approach, an input feature map is processed through several parallel convolutional pathways, or "heads," each specialized to extract a different type of information, such as features at different scales (Minaee et al., 2022). This specialization can be achieved by using different kernel sizes, dilation rates, or pooling operations within each branch. The feature maps produced by these heads are subsequently aggregated, enabling the network to capture a diverse set of representations at the same architectural depth. Note that, unless otherwise stated, ‘multi-head’ in this context refers to multi-branch convolutional or pooling designs (e.g., ASPP, PSPNet), not to multi-head attention as used in Transformers.

This design addresses the challenge of recognizing LULC features that exist at different spatial scales, from small objects to broad, homogeneous regions. Two prominent examples of this strategy are Atrous Spatial Pyramid Pooling (ASPP) and the Pyramid Pooling Module (PSPNet).

Building on dilated convolutions, a prominent example of such a multi-head architecture is the Atrous Spatial Pyramid Pooling (ASPP) module (Chen et al., 2018). ASPP processes

features in parallel using one  $1 \times 1$  convolution, three  $3 \times 3$  atrous convolutions with specific dilation rates (e.g., 6, 12, 18 for an output stride of 16), and an image-level pooling branch. The outputs are concatenated and fused with a final  $1 \times 1$  convolution, as shown in Equation (2.11):

$$y = \text{Conv}_{1 \times 1}(\text{Concat}[x_{1 \times 1}, x_{r=6}, x_{r=12}, x_{r=18}, x_{\text{pool}}]) \quad (2.11)$$

where  $x_{1 \times 1}$  denotes the output of the  $1 \times 1$  convolution branch,  $x_r$  denotes the output of the  $3 \times 3$  atrous convolution with dilation rate  $r$ , and  $x_{\text{pool}}$  represents the features from the image-level pooling branch. By structuring the feature extraction in this multi-head manner, ASPP allows the model to robustly parse scenes containing objects and regions of varying sizes.

A related approach is the Pyramid Scene Parsing Network (PSPNet) (Zhao et al., 2017), which also employs a multi-head strategy but uses multi-scale pooling rather than dilation. The PSPNet module aggregates context via pyramid pooling with fixed bin sizes (e.g.,  $1 \times 1, 2 \times 2, 3 \times 3$ , and  $6 \times 6$ ). These pooled features are then upsampled to the original spatial size and concatenated, providing global and local context information to the final classifier.

While ASPP and PSPNet apply this principle at a single, deep layer, the "multi-head" concept of using parallel branches to capture diverse features is a fundamental strategy. This thesis builds on this principle by integrating a lightweight, multi-head block using parallel dilated convolutions ( $\delta = 1, 2, 3$ ) directly into the final decoder stage of a U-Net backbone, as detailed in Section 4.1.

### 2.4.3 Attention Mechanisms in Convolutional Networks

Spatial-scale limitations are addressed by dilated convolutions, whereas adaptive selection and weighting of relevant features are addressed by attention mechanisms. In pixel-wise classification networks, equal contributions of all features to the final prediction cannot be assumed. Emphasis of informative channels and spatial locations is enabled by attention mechanisms, and improvements in accuracy and interpretability can be obtained.

Sequential channel and spatial attention are implemented by the Convolutional Block Attention Module (CBAM) (Woo et al., 2018). Input features are first processed through channel attention to estimate the most informative channels, after which spatial attention is applied to weight salient locations. Through this two-stage refinement, computational resources are concentrated on informative features.

Channel-wise importance weights are computed by aggregating spatial information, as expressed in Equation (2.12), to produce the channel-attention map (Woo et al., 2018):

$$M_c(\mathbf{F}) = \sigma \left( \text{MLP}(\text{AvgPool}(\mathbf{F})) + \text{MLP}(\text{MaxPool}(\mathbf{F})) \right) \quad (2.12)$$

where input features are denoted by  $\mathbf{F} \in \mathbb{R}^{C \times H \times W}$ ; global pooling across spatial dimensions is performed by AvgPool and MaxPool to produce  $C$ -dimensional vectors; a two-layer perceptron (MLP) with reduction ratio  $r$  (typically 16) followed by expansion to  $C$  channels is applied; and the sigmoid function is denoted by  $\sigma$ . Complementary statistics are captured by the parallel pooling operations; the mean channel response is summarized by average pooling, whereas the strongest response is recorded by max pooling.

Important spatial locations are then identified by spatial attention, as expressed in Equation (2.13), and spatial-attention maps are produced:

$$M_s(\mathbf{X}) = \sigma(\text{Conv}_{7 \times 7}(\text{Concat}[\text{AvgPool}(\mathbf{X}), \text{MaxPool}(\mathbf{X})])) \quad (2.13)$$

where the channel-recalibrated features are denoted by  $\mathbf{X} \in \mathbb{R}^{C \times H \times W}$ ; pooling across the channel dimension produces two spatial maps of size  $H \times W$ ; these are concatenated along the channel dimension, yielding a two-channel input;  $\text{Conv}_{7 \times 7}$  denotes a  $7 \times 7$  convolution applied to the concatenated maps; and the resulting spatial attention map is denoted by  $M_s \in \mathbb{R}^{H \times W}$ . Different statistical properties of the cross-channel feature distribution are captured by the combination of average and max pooling.

These attention mechanisms are applied sequentially in CBAM. Input features are first recalibrated by the channel-attention module through global spatial pooling, dimensionality reduction, and re-expansion to compute per-channel importance weights. The original features are modulated by these weights through element-wise multiplication. Subsequently, important spatial locations are estimated by the spatial-attention module by aggregating channel-wise statistics and applying convolution to produce spatial importance weights. Through this “what-then-where” sequential refinement, focused feature extraction is facilitated.

The two-stage attention process is formalized in Equations (2.14) and (2.15), which show the sequential application:

$$\mathbf{F}' = M_c(\mathbf{F}) \otimes \mathbf{F} \quad (2.14)$$

$$\mathbf{F}'' = M_s(\mathbf{F}') \otimes \mathbf{F}' \quad (2.15)$$

where input features are denoted by  $\mathbf{F} \in \mathbb{R}^{C \times H \times W}$ ; the channel-attention map is denoted by  $M_c \in \mathbb{R}^{C \times 1 \times 1}$ ; the spatial-attention map is denoted by  $M_s \in \mathbb{R}^{1 \times H \times W}$ ; element-wise multiplication with broadcasting is denoted by  $\otimes$ ; channel-attended features are denoted by  $\mathbf{F}'$ ; and the final attended features are denoted by  $\mathbf{F}''$ . The reduction ratio  $r$  and the spatial kernel size ( $7 \times 7$ ) are hyperparameters used to control the capacity and computational cost of the attention modules. The value  $r$  is typically set to 16, following the original CBAM implementation (Woo et al., 2018). This ratio creates a computational bottleneck in the two-layer MLP used for channel attention, significantly reducing the module's parameter count to ensure it remains lightweight.

When integrated into U-Net, CBAM modules can be placed after encoder blocks, at the bottleneck, or within decoder blocks. Emphasis on spectrally discriminative channels and spatially relevant regions is enabled by attention modules, which can be beneficial for classes with subtle spectral differences or complex spatial patterns.

## 2.5 Multimodal Data Fusion Strategies

Images from different sensors are combined by multi-source data fusion to mitigate single-sensor limitations in LULC classification (Ghamisi et al., 2019). Spectral information is provided by optical imagery, whereas structural information is provided by SAR or LiDAR, and improved separation of challenging categories—such as discriminating between different forest types (e.g., deciduous, mixed, and coniferous) or separating spectrally similar classes like wetlands from water bodies—has been reported when these sources are combined (Ghamisi et al., 2019). When optical spectral detail is integrated with SAR structural responses, improved land-surface characterization has been reported (Ghamisi et al., 2019). The selection of an optimal fusion strategy is often guided by the characteristics of the data being combined. Data combinations are typically categorized as either homogeneous, where sources share the same



measurement modality (e.g., multi-temporal optical imagery), or heterogeneous, where sources originate from different sensor types with distinct physical principles (e.g., optical reflectance and SAR backscatter). Based on the stage at which this integration occurs, fusion architectures are commonly grouped into Early (input-level), Middle (feature-level), and Late (decision-level) Fusion (Mena et al., 2024).

### **2.5.1 Early Fusion**

Early Fusion (input-level) is performed by combining different sources at the input stage. After pre-processing, the sources are stacked along the channel dimension to create a single multi-channel input tensor, which is then provided to a single-stream network (Mena et al., 2024). For example, 10 channels from S2 imagery can be concatenated with the VV and VH channels from S1 imagery (vertical transmit/vertical receive; vertical transmit/horizontal receive) to create a 12-channel input for a U-Net-type architecture.

This strategy has been applied in LULC studies. Early Fusion was used for detailed LULC classification in a tropical landscape by stacking four S2 channels with two S1 channels to form a 6-channel input tensor for a U-Net. Both advantages and limitations of this approach were reported (Solórzano et al., 2021).

A characteristic of Early Fusion is that correlations between modalities can be learned in the earliest layers. However, the same convolutional filters are applied to all channels irrespective of source. As a result, optical reflectance values (generally within 0-1 after normalization) and SAR backscatter coefficients (often negative values in decibels) are processed jointly by a single filter, while these quantities are characterized by different statistical properties and physical meaning. This forced sharing of filters may limit modality-specific

feature extraction, particularly when the sources are highly heterogeneous in their measurement physics and statistical distributions.

### **2.5.2 Middle Fusion**

In Middle Fusion (feature-level), each source is processed through separate, parallel encoder branches, and the resulting features are combined at an intermediate stage. Modality-specific features are learned separately by the branches, and the resulting feature maps are fused, typically before a common decoder, via concatenation or element-wise summation, or through learnable mechanisms such as attention-based or gated fusion units by which each modality is adaptively weighted (Mena et al., 2025).

In this configuration, spectral and textural patterns from optical imagery are learned by one encoder branch, whereas geometric and textural information from SAR imagery is learned by another. Effective feature extraction before the integration of complementary information for classification is enabled by this separation of modality-specific processing.

### **2.5.3 Late Fusion**

Late Fusion (decision-level) is performed by training two or more models independently, one for each source. For each model, an output (for example, a class probability map) is produced, and these predictions are combined in a final step to produce the LULC map. The combination rule may be non-learnable, such as averaging probabilities or majority voting, or learnable, in which case an additional classifier is trained to merge the outputs (Mena et al., 2024).

In Late Fusion, each source is processed through independent models. During feature extraction, cross-modal representations are not formed because the branches remain separate.

Resolution of ambiguities that require joint source analysis can be prevented by integration at the prediction level (Ghamisi et al., 2019).

## 2.6 Boundary Effects in Convolutional Networks

Images that often exceed available memory for end-to-end processing are produced by modern satellite sensors. A coverage of  $100 \times 100$  km at 10 m resolution is provided by a single S2 image, and  $10,000 \times 10,000$  pixels per channel are obtained. When 10 channels are stored as 32-bit floating-point values, approximately 4 GB of memory are required for one image. An additional multiplicative increase in memory consumption is incurred by intermediate feature maps during network processing.

Large images are divided into smaller, manageable sections by patch-based processing. For an image of size  $H_{img} \times W_{img}$ , patches of size  $H_{patch} \times W_{patch}$  are extracted and processed independently, as expressed in Equation (2.16):

$$N_{patches} = \left\lceil \frac{H_{img} - H_{patch}}{H_{stride}} + 1 \right\rceil \times \left\lceil \frac{W_{img} - W_{patch}}{W_{stride}} + 1 \right\rceil \quad (2.16)$$

where the stride between adjacent patches is represented by  $H_{stride}$  and  $W_{stride}$ , and the ceiling function is denoted by  $\lceil \cdot \rceil$ . When the stride equals the patch size, the image is tiled without overlap by the patches. When smaller strides are used, overlapping patches are created and multiple predictions are provided for pixels in overlap regions.

As shown in Figure 2.5, within a  $25 \times 25$ -pixel patch, complete contextual information from surrounding pixels is available to the central region (green), whereas progressively less context is available to the edge and corner regions (pink) within the patch boundaries.

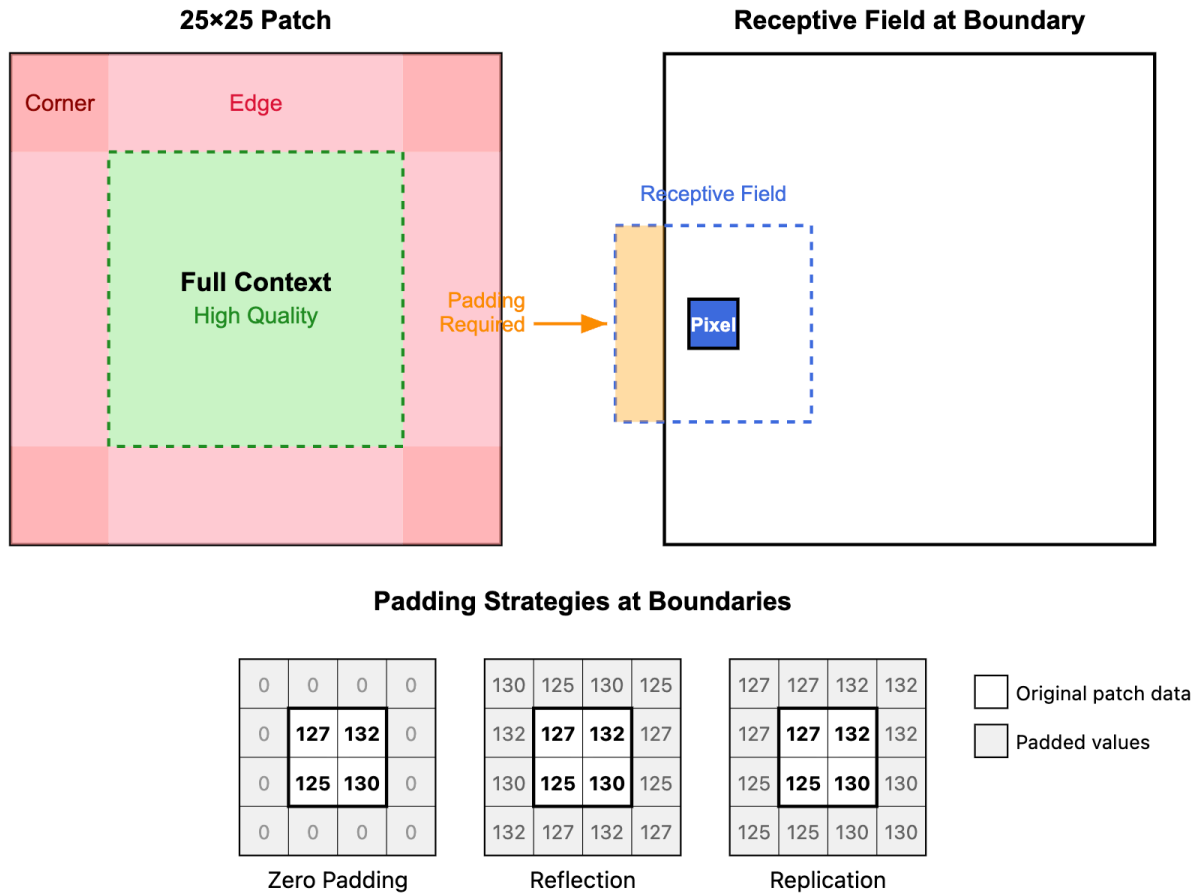


Figure 2.5. Patch-based processing challenges and padding strategies.

The effective receptive field in deep networks is extended beyond the nominal kernel size. The field of view is enlarged by successive pooling and convolution and can be extended to cover substantial areas of the input. As shown in the right panel of Figure 2.5, when a pixel near a patch edge is processed, the receptive field (blue dashed rectangle) is extended beyond the available patch. The missing context is typically approximated with padding.

Three padding strategies are commonly employed, and distinct boundary patterns are introduced by each. Missing values are filled with zeros by zero padding, as shown in the bottom-left panel of Figure 2.5, and artificial low-value borders are created. Pixel values are

mirrored across the boundary by reflection padding, and symmetric patterns are produced (center panel). Edge pixel values are extended outward by replication padding (right panel). Specific artifacts are induced by each strategy: dark borders are induced by zero padding, unrealistic symmetries are induced by reflection, and artificially homogeneous regions are induced by replication.

## **2.7 Interpretability Methods for Deep Learning Networks**

Interpretability methods for deep neural networks are reviewed in this section. Two categories are described: model-agnostic methods (which do not require knowledge of internal network structure), including Local Interpretable Model-agnostic Explanations (LIME) and SHapley Additive exPlanations (SHAP); and activation-mapping methods (which utilize internal network information), including Class Activation Mapping (CAM) and Gradient-weighted Class Activation Mapping (Grad-CAM). The contributions of optical and SAR data to predictions in the fusion experiments were examined using these methods.

### **2.7.1 Local Interpretable Model-agnostic Explanations (LIME)**

Interpretability for individual predictions is provided by LIME through approximation of complex models with simpler, interpretable models in local neighborhoods (Ribeiro et al., 2016). Because LIME is model-agnostic, application to any deep neural network is enabled without requiring access to internal architectures or gradients; interpretation of pre-trained or proprietary models is thereby supported.

It is assumed in LIME that, although global model behavior may be highly nonlinear, local behavior around individual predictions can be approximated by simpler linear models. For a given input instance  $x$  and its prediction  $f(x)$  from a complex model  $f$ , an explanation model

$g$  is sought within a class of interpretable models  $G$  (typically linear models) in LIME. The explanation is obtained by solving the optimization problem shown in Equation (2.17):

$$\xi(x) = \operatorname{argmin}_{g \in G} \mathcal{L}(f, g, \pi_x) + \Omega(g) \quad (2.17)$$

where the unfaithfulness of  $g$  in approximating  $f$  within the locality defined by the proximity measure  $\pi_x$  is measured by  $\mathcal{L}(f, g, \pi_x)$ , and model complexity is denoted by  $\Omega(g)$ . Samples are weighted by the proximity measure according to their similarity to the original instance, as defined in Equation (2.18):

$$w_i = \exp\left(-\frac{D(x, x_i)^2}{\sigma^2}\right) \quad (2.18)$$

where the distance between the original instance and perturbation  $i$  is represented by  $D(x, x_i)$ , and the locality width is controlled by  $\sigma$ .

In the context of deep learning-based pixel classification, image regions are treated as interpretable components through superpixel segmentation in LIME. The process is composed of four steps: (1) the input image is divided into superpixels (contiguous regions of similar pixels); (2) binary masks are generated to switch superpixels on or off, creating perturbed versions in which selected regions are replaced with baseline values (e.g., mean pixel values); (3) each perturbed image is classified by the target network; and (4) a weighted linear regression model is fitted, with Equation (2.18) used to weight samples by proximity. The resulting linear-model coefficients are interpreted as each superpixel's contribution to the prediction; positive

coefficients are associated with regions supporting the predicted class, whereas negative coefficients are associated with regions contradicting it.

Although interpretation without architectural constraints is enabled by LIME, unstable explanations can be induced by the stochastic perturbation process. Furthermore, the generation of numerous forward passes is computationally demanding for high-resolution remote sensing imagery.

### 2.7.2 SHapley Additive exPlanations (SHAP)

A unified framework for interpreting model predictions is provided by SHapley Additive exPlanations (SHAP) (Lundberg and Lee, 2017), which is based on Shapley values from cooperative game theory. To explain how a prediction is generated, SHAP models the prediction process as a "game" where the input features (e.g., spectral bands) are "players" cooperating to produce a "payout" (the model's output). The objective is to fairly distribute this payout among the features. An importance value for each input feature is assigned within the framework based on its average marginal contribution. This contribution is determined by systematically testing the model's prediction with the feature versus without it, across all possible combinations (or coalitions) of the other features. This rigorous averaging process ensures that each feature's influence is fairly assessed, regardless of its interaction with other features. Post-hoc application to trained deep networks is enabled by DeepSHAP, in which the method is adapted for neural architectures with millions of parameters.

A cooperative game-theoretic formulation is adopted in SHAP, in which prediction is viewed as a game and features are treated as players that contribute to the final output. For a model  $f$  with input features  $x = x_1, \dots, x_n$ , the Shapley value for feature  $i$  is computed as shown in Equation (2.19):

$$\phi_i = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|! (n - |S| - 1)!}{n!} [f(S \cup i) - f(S)] \quad (2.19)$$

where the set of all features is denoted by  $N$ , a subset excluding  $i$  is denoted by  $S$ , and the model's expected output when only features in  $S$  are known is denoted by  $f(S)$ .

For deep learning models, the computation of exact Shapley values is rendered intractable by the exponential number of feature coalitions. This challenge is addressed in DeepSHAP by leveraging the compositional structure of neural networks and by propagating contribution scores through network layers using a reference baseline. For a network with layers  $f = f_L \circ f_{L-1} \circ \dots \circ f_1$ , the contribution of input  $x_i$  to output  $y$  is decomposed recursively as shown in Equation (2.20):

$$\phi_i^{(y)} = \sum_j \phi_i^{(h_j)} \cdot \phi_j^{(y)} \quad (2.20)$$

where hidden-layer activations are denoted by  $h_j$ , the contribution of input  $i$  to hidden unit  $j$  is denoted by  $\phi_i^{(h_j)}$ , and the contribution of hidden unit  $j$  to the output is denoted by  $\phi_j^{(y)}$ .

In multi-source remote sensing classification with deep fusion networks, the contributions of different spectral channels or data modalities can be quantified by SHAP. A pixel classification based on optical channels  $c_1, \dots, c_n$  and SAR channels  $s_1, \dots, s_m$  is considered. Channel contributions to the classification decision are indicated by SHAP values  $\phi_{c_i}$  and  $\phi_{s_j}$ , and quantitative assessment of data-source importance is thereby enabled. An



additive consistency property is satisfied by the method, in which the sum of all feature contributions equals the model output relative to the chosen baseline; a theoretically grounded decomposition is thereby provided. However, high computational cost for high-dimensional image data and sensitivity to baseline selection remain as limitations.

### 2.7.3 Class Activation Mapping (CAM)

Visualization of discriminative regions in convolutional neural networks is provided by Class Activation Mapping (CAM) through the relationship between feature maps and class predictions (Zhou et al., 2016). Image regions that contribute to class predictions are localized by CAM through spatial localization maps.

A specific network architecture is required by CAM: convolutional layers followed by GAP and a final fully connected layer. The spatial average of each feature map from the last convolutional layer is computed by the GAP layer, as expressed in Equation (2.21):

$$F_k = \frac{1}{H \times W} \sum_{i=1}^H \sum_{j=1}^W f_k(i, j) \quad (2.21)$$

The spatially averaged feature for the  $k$ -th feature map is denoted by  $F_k$ , and the spatial dimensions are denoted by  $H \times W$ .

Learned weights  $w_k^c$  in the final classification layer are used to combine these averaged features for class  $c$  prediction. The class activation map for class  $c$  is computed by a weighted combination of the feature maps using these same weights, as shown in Equation (2.22):

$$M_c(i, j) = \sum_k w_k^c \cdot f_k(i, j) \quad (2.22)$$

The importance of spatial location  $(i, j)$  for predicting class  $c$  is denoted by  $M_c(i, j)$ . Regions with higher activation-map values are interpreted as contributing more to the class prediction.

The CAM method requires a specific network architecture that includes a Global Average Pooling (GAP) layer. This requirement poses a challenge for pixel-based classification networks like U-Net, which do not include a GAP layer, thereby precluding the direct application of CAM. Furthermore, if a GAP layer is added to a pre-trained network, retraining is required, and the model's performance may be altered.

#### 2.7.4 Gradient-weighted Class Activation Mapping (Grad-CAM)

CAM's architectural constraints are addressed by Grad-CAM, in which gradients are used to weight feature maps, enabling application across diverse convolutional neural network architectures without network modification (Selvaraju et al., 2017).

Gradient information flowing during backpropagation is exploited by Grad-CAM. The class score  $y^c$  is defined as the pre-softmax logit for class  $c$  at a given spatial location. Starting from this class score, gradients are backpropagated to a chosen convolutional layer at which feature maps  $A^k$  preserve spatial information. Sensitivity of the class score to the activation at position  $(i, j)$  in feature map  $k$  is quantified by the gradient  $\partial y^c / \partial A_{ij}^k$ . These spatial gradients are aggregated by GAP to compute importance weights for each feature map, as shown in Equation (2.23) (Selvaraju et al., 2017):

$$\alpha_k^c = \frac{1}{Z} \sum_{i=1}^H \sum_{j=1}^W \frac{\partial y^c}{\partial A_{ij}^k} \quad (2.23)$$

$Z = H \times W$  denotes the total number of spatial positions in the feature map.

Normalization by  $Z$  yields the average gradient across all spatial locations, producing  $\alpha_k^c$ , which denotes the importance of feature map  $k$  for predicting class  $c$ .

The final Grad-CAM visualization is obtained by combining the importance weights with the corresponding activation maps through weighted summation, as given in Equation (2.24):

$$L_{\text{Grad-CAM}}^c = \text{ReLU} \left( \sum_k \alpha_k^c A^k \right) \quad (2.24)$$

$A^k$  denotes the activation values from the  $k$ -th feature map. The weighted sum in Equation (2.24) is used to combine all feature maps according to their importance for class  $c$ . Only positive contributions are retained by the ReLU function, yielding a heatmap that highlights spatial regions contributing positively to the class prediction.

When multi-source data were processed by separate encoder branches, Grad-CAM was applied to the final convolutional layer of each branch to examine modality-specific patterns. High responses near spectral transitions and vegetation boundaries were observed in activation maps from the optical branch. Elevated responses at locations with distinct geometric structures and textures were observed in activation maps from the SAR branch. These patterns were computed using Equations (2.23) and (2.24).

Grad-CAM was selected for analysis of fusion architectures. Extensive computation for large images and dependence on baseline choice are required by LIME and SHAP, which limits their practicality across multiple test images. Application without network modification is enabled by Grad-CAM, and gradient-based weighting is used to localize class-discriminative regions. This method was used to examine how optical and SAR branches contributed to classification in the MHEnc framework.

## Chapter 3 Study Area and Datasets

### 3.1 Study Area and Source Data

The study area is located within the Great Clay Belt of Northern Ontario, Canada (Figure 3.1). This region was selected for its representative boreal landscape characteristics and complex land cover composition. The landscape is characterized as relatively flat to gently undulating, typical of the Great Clay Belt and adjacent boreal lowlands, with numerous water bodies, extensive peatlands, and forests dominated by coniferous and mixed species; challenging multi-class discrimination for LULC classification is thereby posed.

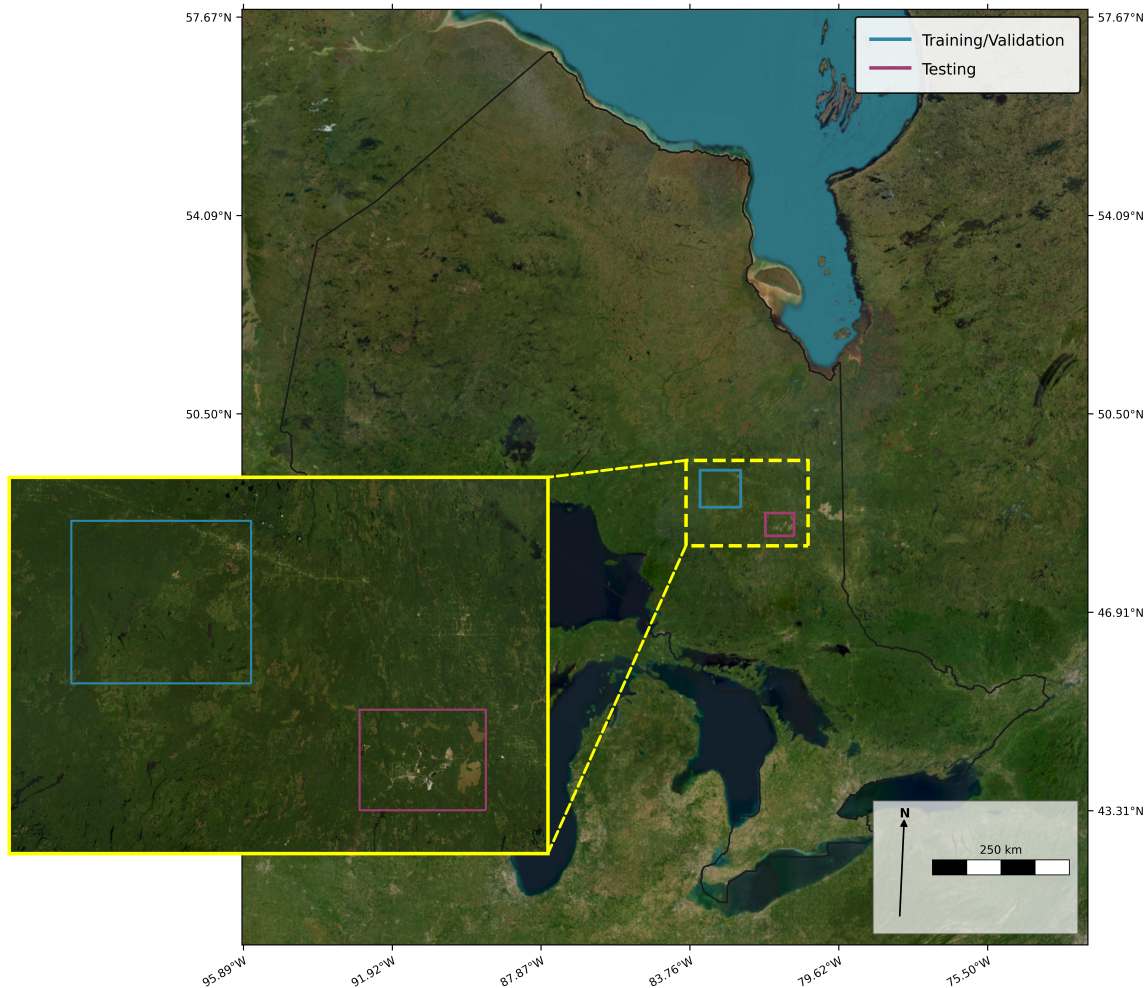


Figure 3.1. The Training/Validation area (outlined in blue) and the Testing area (outlined in purple) are shown within the Great Clay Belt region. An inset map provides a magnified view of the two spatially distinct areas.

Two distinct areas were selected to support robust model evaluation. Approximately 6,096 km<sup>2</sup> are encompassed by the training and validation area, centered at 49.18 °N, 82.90 °W. A total area of 2,650 km<sup>2</sup> is encompassed by the test area, which is located 140 km to the southeast at 48.55 °N, 81.26 °W. This test area was selected for its distinct land cover composition, including higher proportions of water bodies, larger urban settlements (City of Timmins), and more extensive agricultural activities. A challenging test of model generalization is provided by the spatial separation and compositional differences between the training and test areas.

Multi-source satellite imagery from the Copernicus Sentinel constellation was selected for its complementary information content and operational availability. S2 Multispectral Instrument (MSI) imagery was acquired in August 2019 (S2-Aug) and October 2019 (S2-Oct), representing summer leaf-on and late-senescence (leaf-off) conditions. Phenological differences critical for vegetation discrimination, particularly between deciduous and coniferous forest types, were captured by this temporal pairing. Level-2A surface reflectance products were obtained from the Copernicus Open Access Hub. Scenes with cloud cover of less than 3% were used as a preliminary filter, followed by visual inspection to ensure the study areas were cloud-free, thereby ensuring data quality.

Ten spectral channels from the S2 Multispectral Instrument (MSI) were selected for this study, as detailed in Table 3.1. The 60 m spatial resolution channels (B1, B9, B10), which are primarily intended for atmospheric correction, were excluded from the analysis.

Table 3.1. Sentinel-2 MSI bands used in this study and their properties.

| <b>Band ID</b> | <b>Description</b> | <b>Central Wavelength (nm)</b> | <b>Original Resolution (m)</b> |
|----------------|--------------------|--------------------------------|--------------------------------|
| B2             | Blue               | 490                            | 10                             |
| B3             | Green              | 560                            | 10                             |

| <b>Band ID</b> | <b>Description</b> | <b>Central Wavelength (nm)</b> | <b>Original Resolution (m)</b> |
|----------------|--------------------|--------------------------------|--------------------------------|
| B4             | Red                | 665                            | 10                             |
| B5             | Red-Edge 1         | 705                            | 20                             |
| B6             | Red-Edge 2         | 740                            | 20                             |
| B7             | Red-Edge 3         | 783                            | 20                             |
| B8             | NIR                | 842                            | 10                             |
| B8A            | NIR (narrow)       | 865                            | 20                             |
| B11            | SWIR 1             | 1610                           | 20                             |
| B12            | SWIR 2             | 2190                           | 20                             |

All S2 imagery was resampled to 10 m spatial resolution using the nearest-neighbor method to avoid radiometric mixing during resampling. The S2 imagery served as the geometric reference, and the processed S1 imagery was subsequently co-registered to align all datasets to the same 10 m grid system.

S1 imagery was acquired in August 2019 (S1-Aug), temporally aligned with the optical acquisition. Data were collected in Interferometric Wide Swath (IW) mode at C-band frequency (5.405 GHz) using Level-1 Ground Range Detected (GRD) products as input for processing.

SAR processing for this study was conducted using the Sentinel Application Platform (SNAP) software (European Space Agency, 2024). The processing chain included radiometric calibration to Sigma Nought ( $\sigma^0$ ) backscatter coefficients, speckle filtering using a Refined Lee filter with a 7×7-pixel window, and Range-Doppler terrain correction using the Shuttle Radar Topography Mission (SRTM) 1-arcsecond digital elevation model. The resulting VV and VH polarization channels were resampled to 10 m spatial resolution and retained in linear  $\sigma^0$  units prior to normalization.



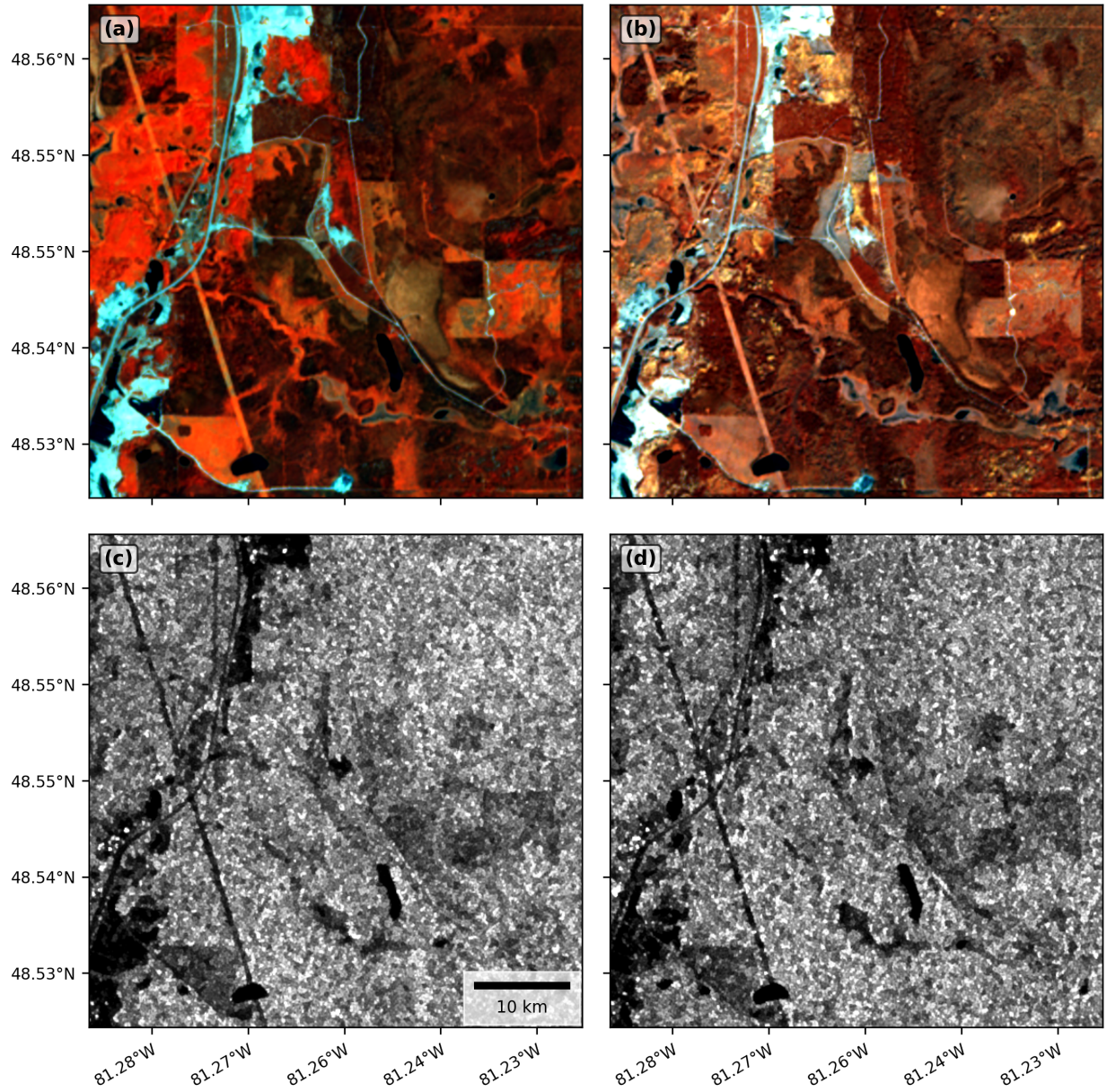


Figure 3.2. Visual representation of the remotely sensed imagery over a representative sub-area within the test area. (a) S2-Aug false color composite. (b) S2-Oct false color composite. (c) S1-Aug VV polarization backscatter. (d) S1-Aug VH polarization backscatter.

Optical and SAR imagery from a representative sub-area within the test area are shown in Figure 3.2. Bright red tones are exhibited by healthy vegetation during the peak growing season, urban/developed areas appear as cyan tones, and water bodies are exhibited as dark blue or black in the S2-Aug false-color composite (a). A shift from red to brown due to leaf senescence is



observed in the S2-Oct composite (b), and separation between deciduous and evergreen forests is thereby facilitated. Speckle noise is present in the SAR imagery (c, d). Strong returns from buildings are produced in VV polarization (c), while volume scattering from vegetation canopies is recorded by VH polarization (d).

Canopy Height Model (CHM) data were obtained from the Ontario Ministry of Natural Resources and Forestry's Forest Resources Inventory (FRI) leaf-on Light Detection and Ranging (LiDAR) program. This dataset, with a native spatial resolution of 0.5 m, is distributed as a series of geographically distinct files, or "tiles". A total of 8,100 such tiles covering the training and validation area and 2,668 tiles covering the test area were downloaded through spatial queries against the FRI tile index. These tiles were mosaicked using Geospatial Data Abstraction Library (GDAL) Virtual Raster functionality and resampled to 10 m spatial resolution to match the Sentinel grid. Bilinear interpolation was selected for this resampling process; as CHM represents continuous height data, this method provides a smoother and more realistic surface aggregation to the coarser 10 m grid compared to the blocky artifacts that would be produced by nearest-neighbour interpolation. The resulting CHM data served as an independent validation source for assessing the biophysical consistency of the reference land-cover labels.

In addition to the satellite imagery and LiDAR data, existing Canadian geospatial products were used as the basis for generating spatially explicit reference maps. These datasets were selected for complementary coverage and temporal alignment with the study period. The specific products and the methodology for their integration to create the reference labels are detailed in Section 3.2.

### 3.2 Reference Data Generation

The study area was categorized into seven LULC classes: (0) Water, (1) Urban/Developed, (2) Wetland, (3) Agriculture, (4) Deciduous Forest, (5) Mixed Forest, and (6) Coniferous Forest.

Table 3.2. Description of the seven target LULC classes used in this study.

| Class ID | Class Name        | Description                                                                                        |
|----------|-------------------|----------------------------------------------------------------------------------------------------|
| 0        | Water             | Open water bodies including lakes, rivers, and reservoirs                                          |
| 1        | Urban/Developed   | Built-up areas comprising residential, commercial, and industrial infrastructure                   |
| 2        | Wetland           | Areas of standing water with emergent vegetation, including marshes, swamps, and peatlands         |
| 3        | Agriculture       | Active croplands and pastures under cultivation                                                    |
| 4        | Deciduous Forest  | Broadleaf forest with deciduous tree crown cover exceeding 75%                                     |
| 5        | Mixed Forest      | Forest containing both coniferous and deciduous trees, with neither type exceeding 75% crown cover |
| 6        | Coniferous Forest | Evergreen forest with coniferous tree crown cover exceeding 75%                                    |

Reference labels were generated by integrating three Canadian geospatial products, selected for their complementary coverage and temporal alignment with the study period. The first dataset was the 2020 Land Cover of Canada (LCC), which provided national-scale coverage at 30 m resolution (Natural Resources Canada, 2023). This product ensured broad geographic consistency across the study region. The second dataset was the Ontario Land Cover Version 1.0 (OLC V1.0) (Ontario Ministry of Natural Resources and Forestry, 2023), which contributed provincial-scale detail at 15 m resolution and provided regional specificity for boreal forest classification. The third dataset was the 2019 Annual Crop Inventory (ACI) (Agriculture and Agri-Food Canada, 2023), which supplied agricultural land classification at 30 m resolution. The ACI was generated using decision-tree methodology applied to optical and radar imagery for the

2019 growing season, enabling accurate agricultural area delineation. Each source was reclassified to the seven target classes defined in Table 3.2, resampled using nearest-neighbor interpolation, and co-registered to a 10 m grid aligned with S2 to preserve categorical labels. The workflow comprised reclassification, resampling and co-registration, pixel-wise agreement analysis with consensus labeling, and canopy-height-based consistency checks. Pixels flagged as NoData were excluded from evaluation metrics while being retained in the input imagery to maintain spatial context; during training and validation, loss and accuracy computations were configured to ignore pixels with NoData labels.

Substantial disagreement was revealed by pixel-level comparison across the three reclassified products, with different classes frequently assigned to the same locations. This disagreement was attributed to inconsistent class definitions across products and to partially overlapping acquisition periods (OLC V1.0: 2015 and 2020; ACI: 2019; LCC: 2019-2021).

As shown in Figure 3.3, approximately 61.0 million pixels were contained within the training and validation area. Grey regions (43.6%;  $\approx 26.6$  million pixels) were identified where three-way comparison among LCC, OLC V1.0, and ACI could not be performed, either because two or more sources lacked valid labels or because one source was NoData while the other two provided different classes. The remaining 56.4% ( $\approx 34.4$  million pixels) were eligible for comparison between at least two sources. Within this comparable portion, full agreement among all three sources (green) accounted for 11.2% of the total area. Partial agreement between exactly two sources (yellow) covered 30.5%, including cases where one source was NoData and the other two provided matching labels. Complete disagreement among all three valid sources (red) accounted for 14.7%. The substantial proportions of partial agreement and complete disagreement highlighted significant label uncertainty. This high degree of discrepancy was

interpreted as reflecting differences in class definitions, mapping methodologies, and temporal coverage among the source products. Accordingly, a consensus-based approach was adopted as the primary strategy to mitigate these inconsistencies and construct reliable reference data for model training and evaluation.

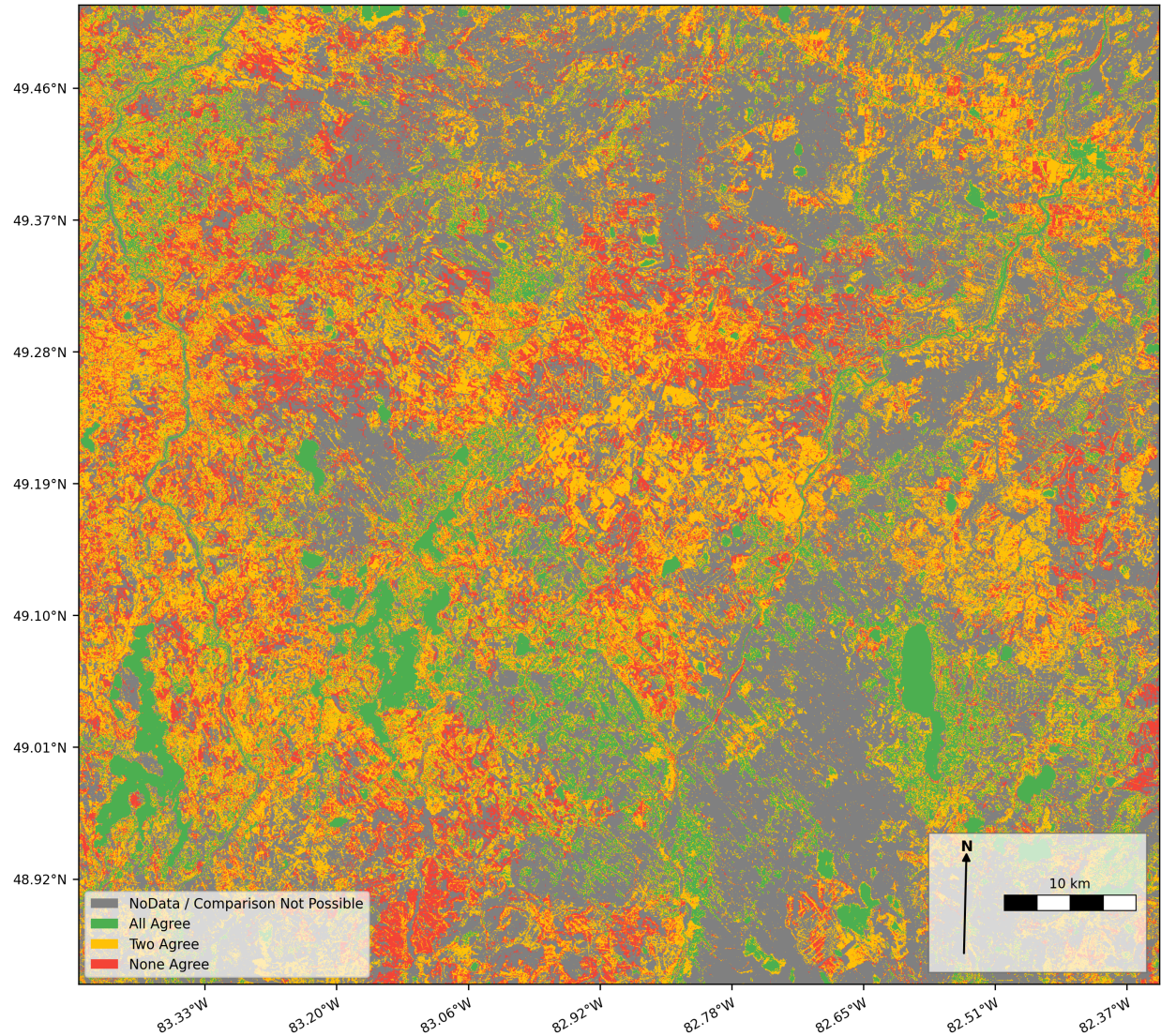


Figure 3.3. Comparison map among the three reclassified datasets in the training area. Green: all three agree; Yellow: exactly two agree; Red: all three disagree; Grey: comparison not possible.

These discrepancies were addressed using a class-tuned consensus strategy with a defined hierarchy of rules. First, cases involving pixels with incomplete data were processed. When one source was NoData and the other two matched, the matching label was accepted. When one source was NoData and the remaining two disagreed, the pixel was set to NoData. For the remaining pixels, where all three sources provided valid labels, class-specific rules were applied. For Water, Wetland, and the three forest classes, strict agreement across all three datasets (i.e.,  $A=A=A$ ) was required. For Urban/Developed and Agriculture, which were underrepresented or inconsistently mapped, a relaxed criterion of agreement between at least two datasets (i.e.,  $A=A=A$  or  $A=A=B$ ) was accepted to retain sufficient samples. Any pixels that failed to meet these criteria (e.g., complete disagreement ( $A=B=C$ ), or 2-way agreement for a strict-rule class) were set to NoData.

Following the consensus labeling, validation based on the LiDAR-derived Canopy Height Model (CHM) was applied as a final filter to further mitigate the identified label uncertainty. This step was intended to ensure biophysical consistency between the consensus labels and their expected vertical structure, thereby removing clear contradictions from the reference map. A primary threshold of 10 m was established to separate forest from non-forest classes. This value was selected based on empirical analysis of CHM statistics for pixels where all three source datasets consistently agreed on a forest class (Deciduous, Mixed, or Coniferous). The 10 m value was found to be a reliable minimum height that effectively filtered out non-forest areas (e.g., clear-cuts, young regrowth) that may have been mislabeled as forest in the source products. Accordingly, pixels labeled as any of the three forest classes but having a CHM value  $< 10$  m were flagged as inconsistent and set to NoData. Conversely, this 10 m threshold was also applied as an upper bound for the Agriculture class to identify and exclude obvious errors. Pixels

labeled as Agriculture with a CHM value  $\geq 10$  m were set to NoData. While agricultural crops do not biophysically reach this height, this high threshold was not intended to define crop characteristics. Instead, it served as a robust, temporally-independent filter to remove pixels that were biophysically inconsistent with an 'Agriculture' label, such as misclassified forest patches or isolated trees within fields. This approach ensured that the filter could be applied reliably regardless of the crop's specific growth stage at the time of data acquisition. Finally, a separate threshold of CHM  $\geq 2$  m was used to exclude pixels labeled as Water, a value selected to account for potential noise or emergent vegetation at water edges.

Table 3.3. Class distribution of the training/validation reference map after consensus labeling and CHM filtering.

| <b>Class ID</b> | <b>Class name</b> | <b>Pixels</b> | <b>Ratio (%)</b> |
|-----------------|-------------------|---------------|------------------|
| 0               | Water             | 1,931,013     | 3.17             |
| 1               | Urban/Developed   | 276,574       | 0.45             |
| 2               | Wetland           | 1,569,112     | 2.57             |
| 3               | Agriculture       | 69,395        | 0.11             |
| 4               | Deciduous         | 2,081,882     | 3.42             |
| 5               | Mixed             | 11,993,636    | 19.67            |
| 6               | Coniferous        | 10,708,867    | 17.57            |
| —               | <b>NoData</b>     | 32,329,555    | 53.03            |
| —               | <b>Total</b>      | 60,960,034    | 100.00           |

The final training and validation reference map on the 10 m Sentinel-2 grid contained the class distribution shown in Table 3.3. After consensus labeling and canopy-height consistency filtering were applied, NoData pixels comprised 53.03% of the area. The significance of this high proportion is that it reflects the rigorous, conservative data quality protocol; label certainty and biophysical consistency were explicitly prioritized over the total quantity of training pixels. This process substantially reduces label noise from inter-source disagreements and contradictory data (e.g., low-height forest labels). The primary methodological impact is the requirement to mask these pixels during optimization; as detailed in Section 4.1, loss computations were

configured to ignore all NoData locations. Although a large portion of the area was excluded from loss calculation, the remaining 46.97% (approximately 28.6 million pixels) provided a high-confidence dataset of sufficient magnitude for robust model training. Among the valid labeled classes, Mixed Forest and Coniferous Forest accounted for 19.67% and 17.57% of the area, respectively, whereas Urban/Developed and Agriculture remained underrepresented at 0.45% and 0.11%. This class imbalance within the valid data was addressed using weighted sampling and composite loss strategies, as described in Sections 4.4.2 (weighted sampling) and 4.1 (composite loss).



### 3.3 Normalization of Remotely Sensed Data

All satellite imagery was Z-score normalized at the scene level prior to patch extraction. Pre-normalization pixel-value distributions for S2-Aug are shown in Figure 3.4, in which differences between the training and validation area and the test area were evident.

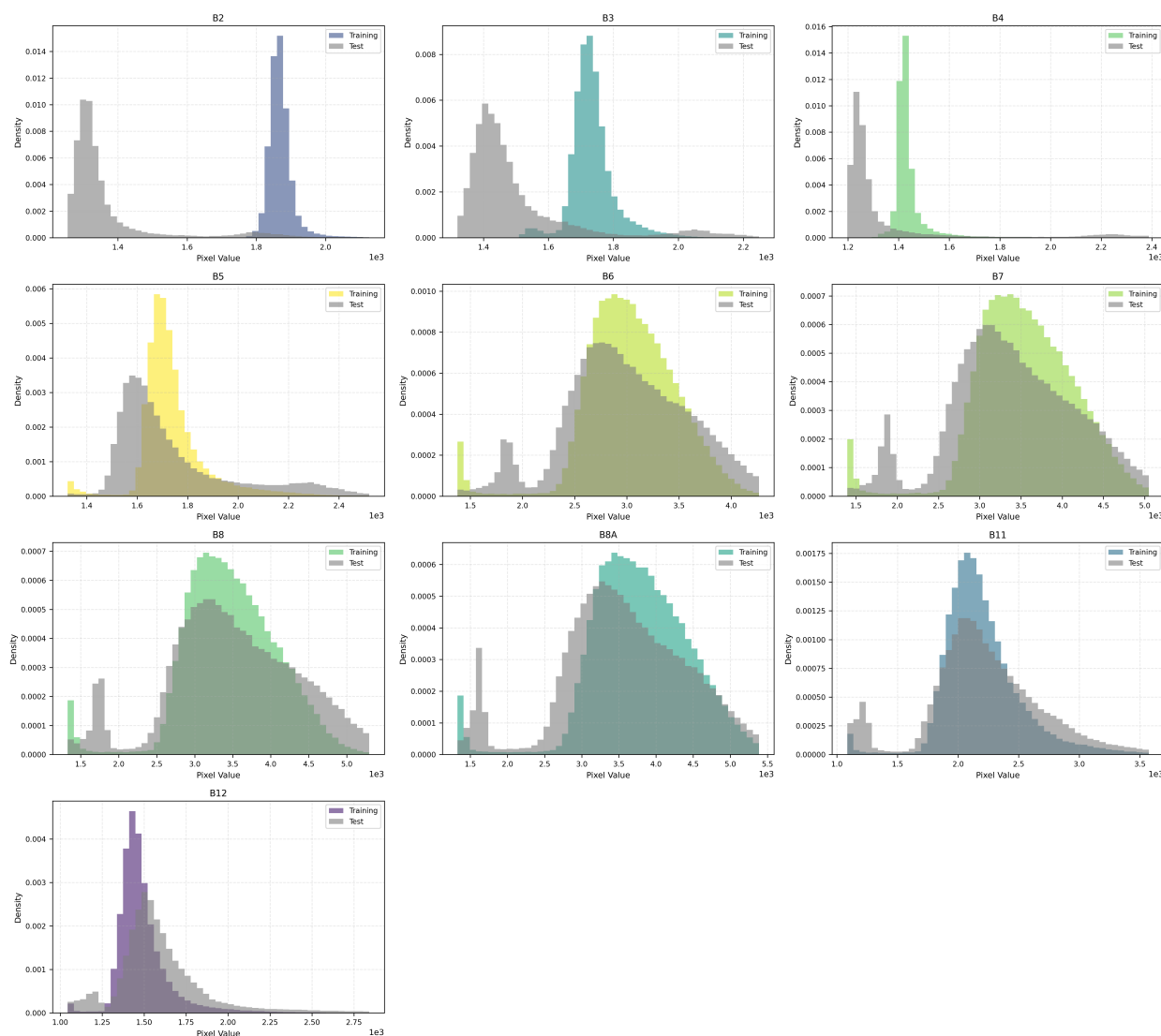


Figure 3.4. Pre-normalization pixel-value distributions for S2-Aug in the training and validation area (colored) and the test area (grey).

In the visible bands (B2-B4), a shift toward higher central tendency for the training and validation area relative to the test area was observed. In the red-edge and Near Infrared (NIR)



bands (B5-B8, B8A), distributions from the test area were characterized by greater variance and lower peak densities. Within the SWIR bands, no rightward shift of the training relative to the test distribution was observed for B11, whereas B12 was characterized by a pronounced upper tail in the test distribution. An extended upper tail with low density was also present in the B4 test distribution. These distributional differences were interpreted as being associated with variations in atmospheric conditions, acquisition geometry, and land-cover composition between the two areas. Potential reductions in generalization capability were considered given the statistical differences in pixel values between the training and validation area and the test area. To address this, per-scene, per-band Z-score normalization was applied independently to S1 and S2 scenes. For each pixel, the normalized value was computed as shown in Equation (3.1):

$$Z_{i,j,c} = \frac{x_{i,j,c} - \mu_c}{\sigma_c} \quad (3.1)$$

where the original pixel value at location  $(i, j)$  for channel  $c$  is denoted by  $x_{i,j,c}$ ; the scene-level mean for band  $c$  computed over valid pixels is denoted by  $\mu_c$ ; and the corresponding standard deviation is denoted by  $\sigma_c$ .

## Chapter 4 Deep Learning Framework and Experimental Design

### 4.1 Multi-Head Encoder (MHEnc) Framework

To address the multi-scale challenge in LULC classification, MHEnc was developed based on a shallow encoder-decoder backbone termed BasicUNet. Three parallel convolutional heads with dilation rates  $\delta = 1, 2, 3$  were applied to the decoder features, and the resulting class logits were combined via learnable attention weights.

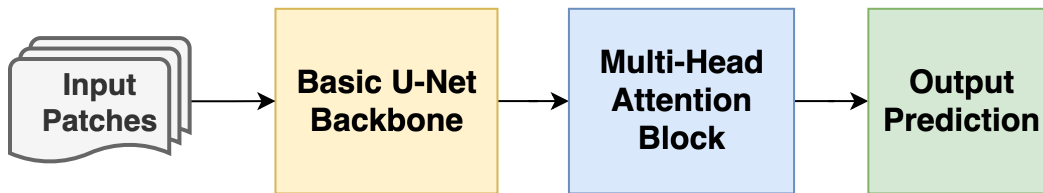


Figure 4.1. MHEnc architecture overview.

The processing pipeline is illustrated in Figure 4.1. Input patches were encoded and decoded through BasicUNet, with features at the original resolution produced. These decoded features were then processed by the Multi-Head Block, in which attention-weighted fusion was applied to generate the final logits. A shallow backbone architecture was adopted to preserve spatial detail for  $25 \times 25$ -pixel inputs.

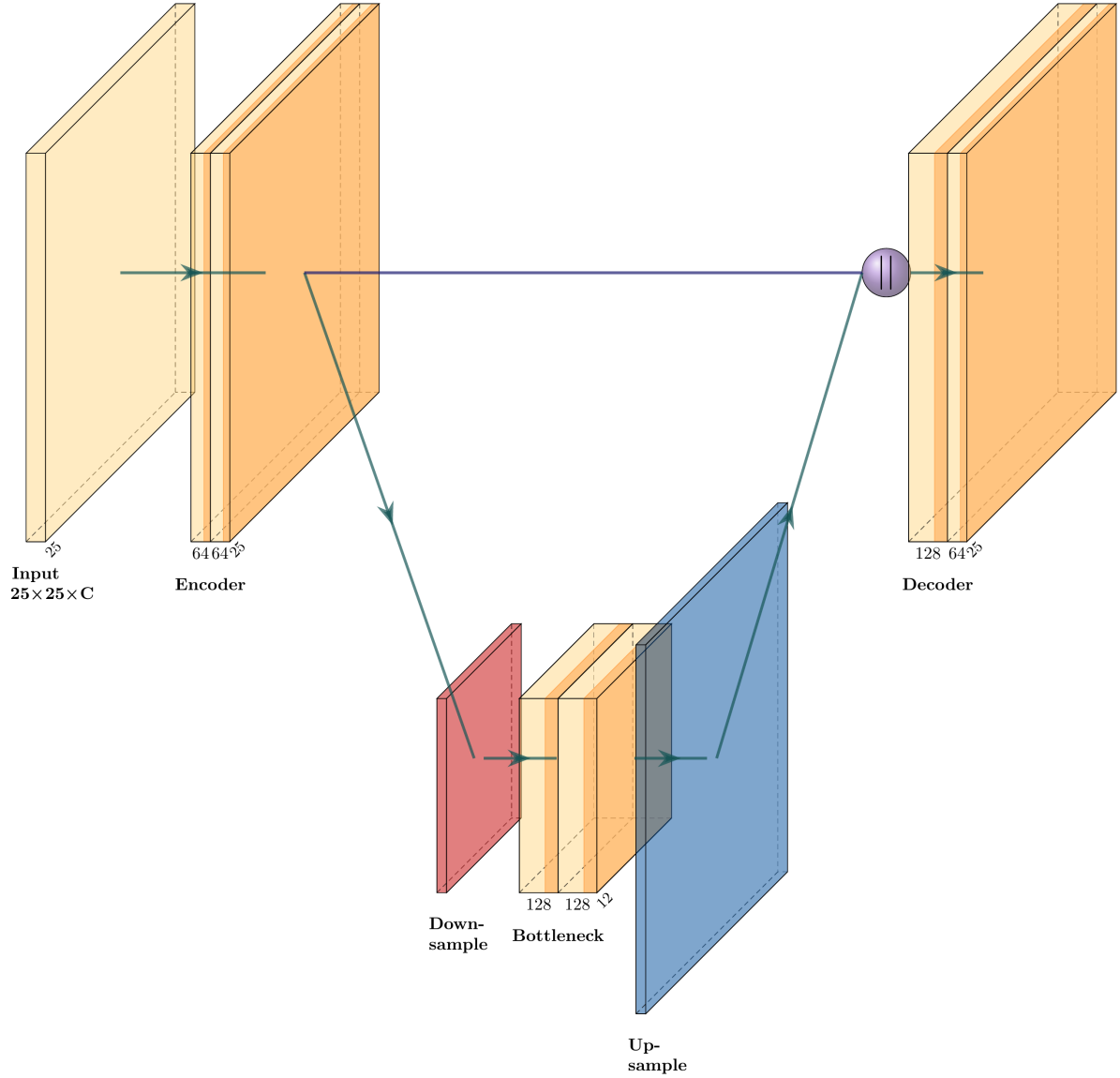


Figure 4.2. BasicUNet architecture.

The BasicUNet architecture, shown in Figure 4.2, was adopted as the shallow backbone. The encoder was composed of an initial block containing two  $3 \times 3$  convolutions with batch normalization and ReLU activation, from which 64 feature channels were produced. Spatial dimensions were reduced from  $25 \times 25$  to  $12 \times 12$  pixels through  $2 \times 2$  max pooling. These downsampled features at  $12 \times 12$  resolution were processed by the second encoder block, where the channel dimension was expanded from 64 to 128.

In the decoder, features were upsampled from  $12 \times 12$  to  $24 \times 24$  through a  $2 \times 2$  transposed convolution. Because pooling was applied to an odd-sized input, one-pixel symmetric padding was added to restore the  $25 \times 25$  dimensions. Encoder and decoder features were concatenated through skip connections, and the combined 128-channel representation was refined by a decoder block to produce 64 channels. In BasicUNet, a  $1 \times 1$  convolutional classifier was then applied to generate class predictions. In MHEnc, this 64-channel backbone output was passed to the Multi-Head Block, where attention-weighted multi-scale fusion was performed.

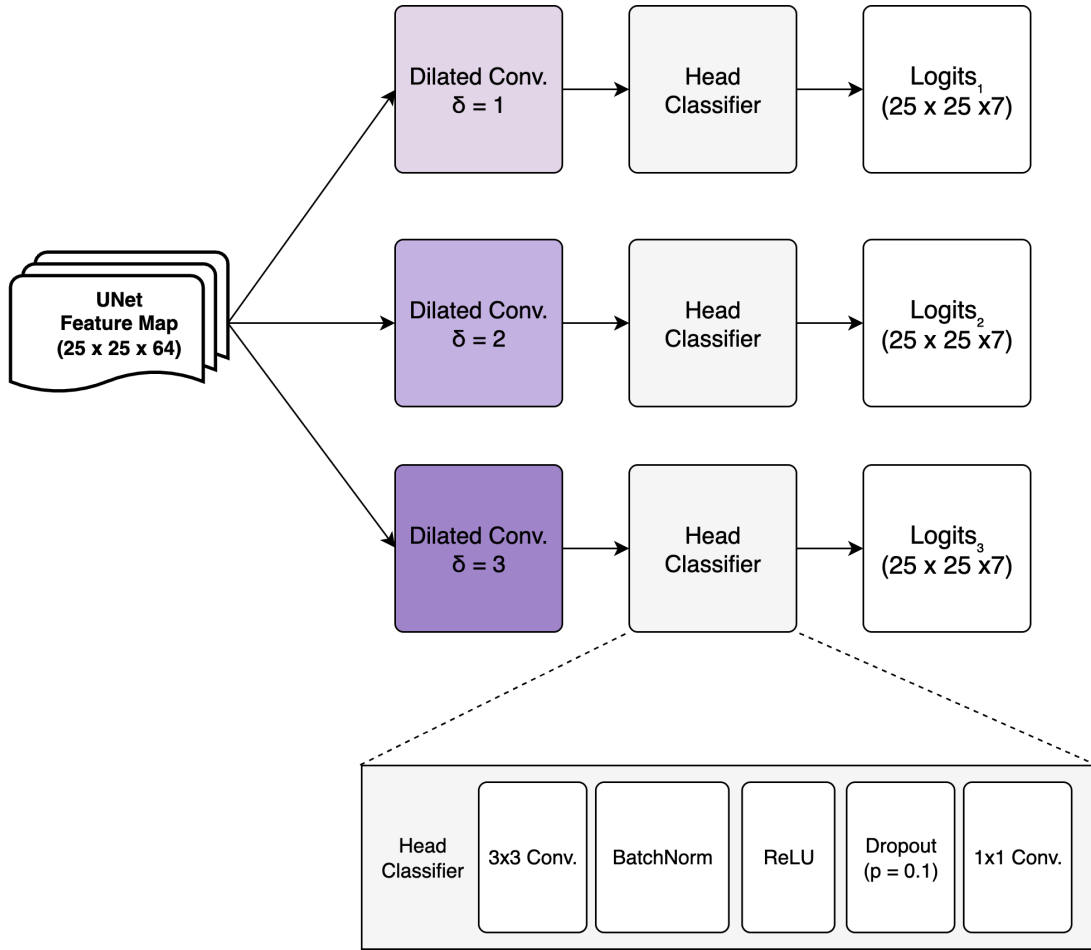


Figure 4.3. Multi-Head Block architecture.

The distinguishing component of MHEnc was the Multi-Head Block applied to the decoder output. The architecture of this block is detailed in Figure 4.3. To address receptive-field

limitations of fixed  $3 \times 3$  convolutions in the BasicUNet backbone, decoder feature maps of size  $64 \times 25 \times 25$  were processed through three parallel branches. A  $3 \times 3$  convolution with dilation rate  $\delta = 1$  was employed by the first branch to extract local detail. A dilation rate  $\delta = 2$  was used by the second branch to capture intermediate-scale patterns. A dilation rate  $\delta = 3$  was used by the third branch to aggregate broader spatial context. Identical input-output channel counts and  $3 \times 3$  kernel dimensions were maintained across all branches, resulting in equal parameter counts per branch.

Each branch output was transformed by a Head-Classfier module composed of a  $3 \times 3$  convolution, batch normalization, ReLU activation, spatial dropout (2D,  $p = 0.1$ ), and a  $1 \times 1$  convolution, producing logits with dimensions  $7 \times 25 \times 25$ . The final prediction was computed by weighted summation of the three branch logits using attention weights  $\alpha_i$ , where  $i \in 1, 2, 3$ . These weights were obtained by applying softmax normalization to three learnable scalar parameters.

Moderate computational overhead was introduced by the multi-scale design relative to alternative approaches. Nine parameters per input-output channel pair were required for each dilated convolution regardless of dilation rate, whereas 25 and 49 parameters would be needed for actual  $5 \times 5$  and  $7 \times 7$  kernels, respectively. Three scalar weights were employed for direct logit combination in the fusion mechanism, eliminating the need for channel concatenation and subsequent  $1 \times 1$  convolution required by methods such as ASPP. The absolute Floating Point Operations (FLOPs) were constrained by the  $25 \times 25$ -pixel input resolution and a single downsampling stage in the shallow backbone. Lower incremental cost was achieved through these design choices compared to architectures in which larger kernels or concatenation-based fusion strategies were employed.

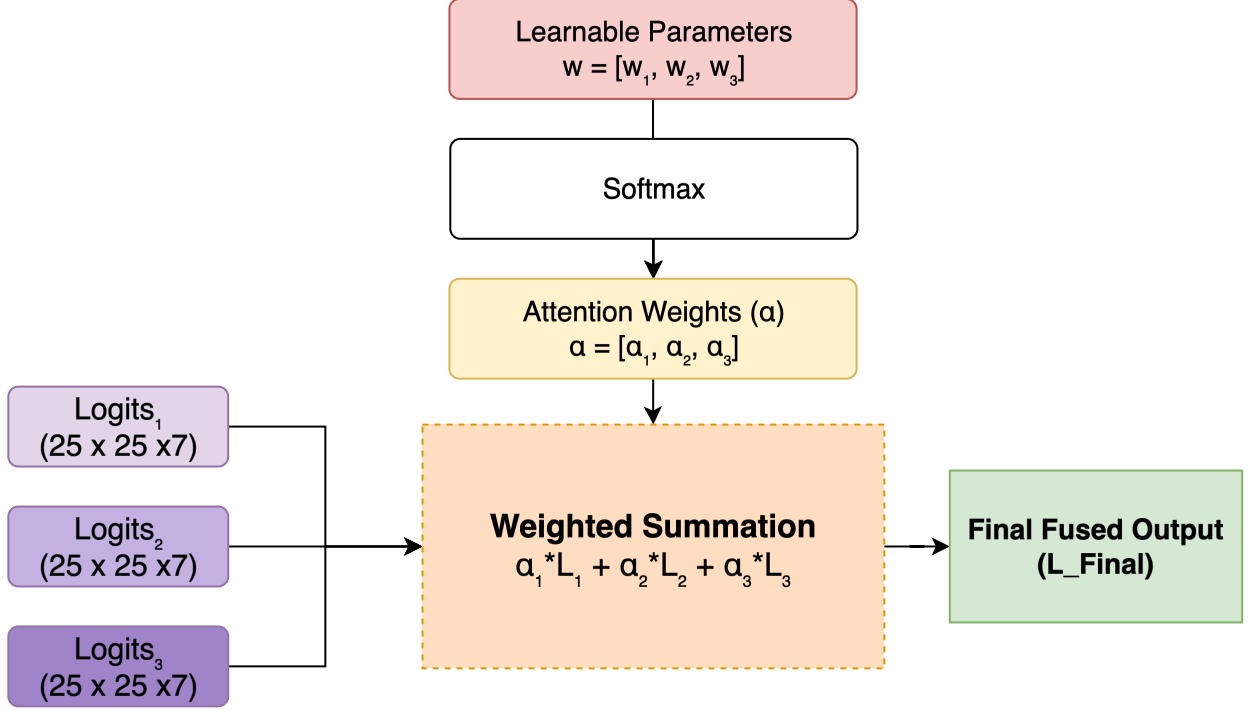


Figure 4.4. Attention-weighted fusion mechanism.

Multi-scale predictions were combined through attention-weighted fusion. This mechanism is illustrated in Figure 4.4. Three learnable scalars were denoted by  $\mathbf{w} = [w_1, w_2, w_3]$ . By applying a softmax, normalized attention weights  $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \alpha_3]$  were obtained as shown in Equation (4.1):

$$\alpha_i = \frac{\exp(w_i)}{\sum_{j=1}^3 \exp(w_j)} \quad (4.1)$$

Weights were initialized uniformly such that  $\alpha_i = 1/3$  at the start of training. This global, input-agnostic weighting scheme was adopted to make the relative contribution of each receptive-field scale explicit and stable. Interpretability was also enhanced because the learned  $\alpha_i$  values directly indicated scale preferences across the dataset.

Per-head logits were denoted by  $\mathbf{Z}_i \in \mathbb{R}^{7 \times 25 \times 25}$ . The fused logits were computed as shown in Equation (4.2):

$$\mathbf{Z}_{\text{final}} = \sum_{i=1}^3 \alpha_i \mathbf{Z}_i \quad (4.2)$$

Fusion at the logit level was chosen to preserve each head’s calibrated class evidence without requiring additional feature-alignment layers. This approach was consistent with maintaining a shallow backbone for  $25 \times 25$  inputs while multi-scale information was leveraged.

The training loss combined pixel-wise and region-overlap terms as shown in Equation (4.3):

$$\mathcal{L}_{\text{comp}} = 0.5 \mathcal{L}_{\text{focal}} + 0.5 \mathcal{L}_{\text{dice}} \quad (4.3)$$

Focal loss was configured with  $\gamma = 2.0$  and a class-weight factor  $\alpha_t = 0.25$  to reduce the influence of abundant easy pixels and to up-weight under-represented classes (Lin et al., 2017).

Dice loss was used to optimize set overlap, which is sensitive to boundary errors and class extent (Milletari et al., 2016). To ensure optimization was based only on verified labels, pixels corresponding to NoData locations in the reference map were masked and excluded from all loss computations. This masking strategy was applied consistently across all training and validation procedures.

To prevent collapse to a single head while allowing data-driven scale preference, an attention regularizer was applied as shown in Equation (4.4):

$$\mathcal{L}_{\text{reg\_attn}} = \lambda_{L2} \|\mathbf{w}\|_2^2 + \lambda_{KL} D_{KL}(\boldsymbol{\alpha} \parallel \mathbf{U}) \quad (4.4)$$

The constants were set to  $\mathbf{U} = [1/3, 1/3, 1/3]$ ,  $\lambda_{L2} = 0.01$ , and  $\lambda_{KL} = 0.1$ . The  $\ell_2$  term was used to bound the magnitude of  $\mathbf{w}$  and to stabilize optimization. The Kullback–Leibler (KL)-divergence term was used as a soft prior toward balanced utilization of the three heads, while deviation was permitted when supported by the data. During inference, the learned  $\boldsymbol{\alpha}$  values were held fixed to maintain interpretability of the scale mixture.

The total loss was defined as shown in Equation (4.5):

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{comp}}(\mathbf{Z}_{\text{final}}) + \alpha_{\text{aux}} \frac{1}{3} \sum_{i=1}^3 \mathcal{L}_{\text{comp}}(\mathbf{Z}_i) + \alpha_{\text{reg}} \mathcal{L}_{\text{reg\_attn}} \quad (4.5)$$

The auxiliary term was designed to ensure that each head remained predictive early in training, which facilitated gradient flow and prevented dominance of a single branch before attention weights had stabilized. The auxiliary weight  $\alpha_{\text{aux}}$  was scheduled to increase linearly from 0 to  $w_{\text{init}}$  over a warm-up phase and then from  $w_{\text{init}}$  to  $w_{\text{final}}$  over the remaining epochs (with defaults  $w_{\text{init}} = 0.2$  and  $w_{\text{final}} = 0.4$ ). The regularization weight was fixed at  $\alpha_{\text{reg}} = 0.01$ . This schedule was adopted for training the shallow backbone on 25×25-pixel patches.

## 4.2 Exploration of Fusion Strategies

The fusion stage was selected based on the characteristics of the input imagery. Two types of heterogeneity were present: sensor differences between S1-Aug and S2-Aug, and



temporal differences between S2-Aug and S2-Oct. Two scenarios were designed to examine these cases using a common backbone.

**Case 1: Heterogeneous Sensor Fusion.** This scenario involved the combination of S1-Aug (VV and VH polarizations, calibrated to  $\sigma^0$  and Z-score normalized) with S2-Aug (10 bands). Both datasets, having been normalized as described in Section 3.3, were used as inputs. The primary challenge was identified as the reconciliation of differing measurement physics. Responses to surface geometry and roughness with characteristic speckle texture were produced in SAR backscatter, whereas spectral contrasts from materials and vegetation were captured by optical reflectance. Numerical scaling was addressed by per-band normalization; however, reconciliation of the fundamental difference in what each sensor measures was not achieved. Accordingly, modality-specific processing before integration was required.

**Case 2: Homogeneous Temporal Fusion.** In this scenario, S2-Aug was combined with S2-Oct. Identical spectral bands and measurement principles were shared by both, whereas different phenological states, peak vegetation versus senescence and post-harvest conditions, were captured. The informative signal was considered to reside in temporal changes between corresponding bands. The fusion strategy was designed to enable direct learning of these temporal gradients while maintaining consistent representations across dates.

For both scenarios, the Early, Middle, and Late Fusion strategies, as defined in Section 2.5, were implemented on top of the MHEnc backbone. By using the same encoder-decoder capacity, training schedule, and evaluation protocol, differences in performance were attributable to the fusion stage rather than to model size or optimization.

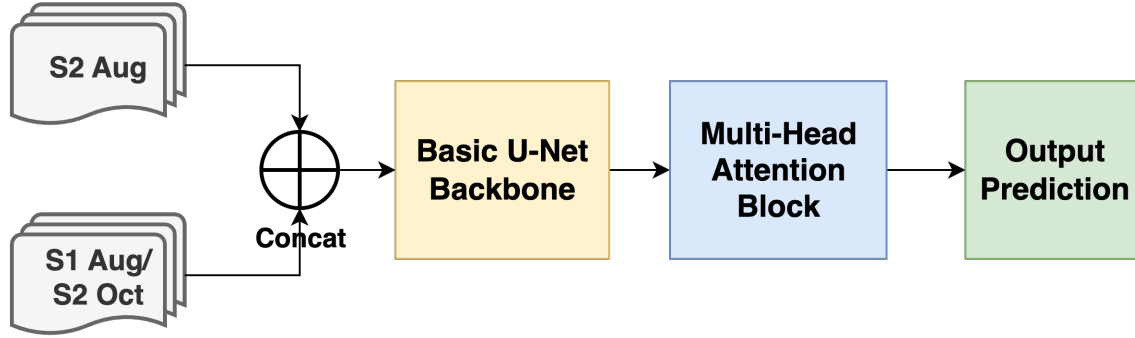


Figure 4.5. Diagram of Early Fusion.

As illustrated in Figure 4.5, all channels were concatenated at the input in Early Fusion. A single multi-channel input was formed and was processed by the MHEnc backbone. In Case 1 (S1-S2, August 2019), the input contained 12 channels (10 optical and 2 SAR). In Case 2 (S2-Aug and S2-Oct, 2019), the input contained 20 channels (10 + 10). While dimensionality reduction techniques, such as Principal Component Analysis (PCA), could be applied to these concatenated inputs to reduce computational load, this study employed direct concatenation. This decision was based on two factors. First, it adheres to an end-to-end learning philosophy, allowing the initial convolutional layers to learn optimal feature representations directly rather than relying on a fixed statistical transformation. Second, for homogeneous temporal fusion (Case 2), direct concatenation was specifically chosen to preserve the channel-wise relationships between dates, enabling the network to explicitly model temporal gradients and phenological changes, which would be obscured by a PCA transformation. The first layers were optimized on the joint input during training so that features were aligned to information from all channels from the outset.

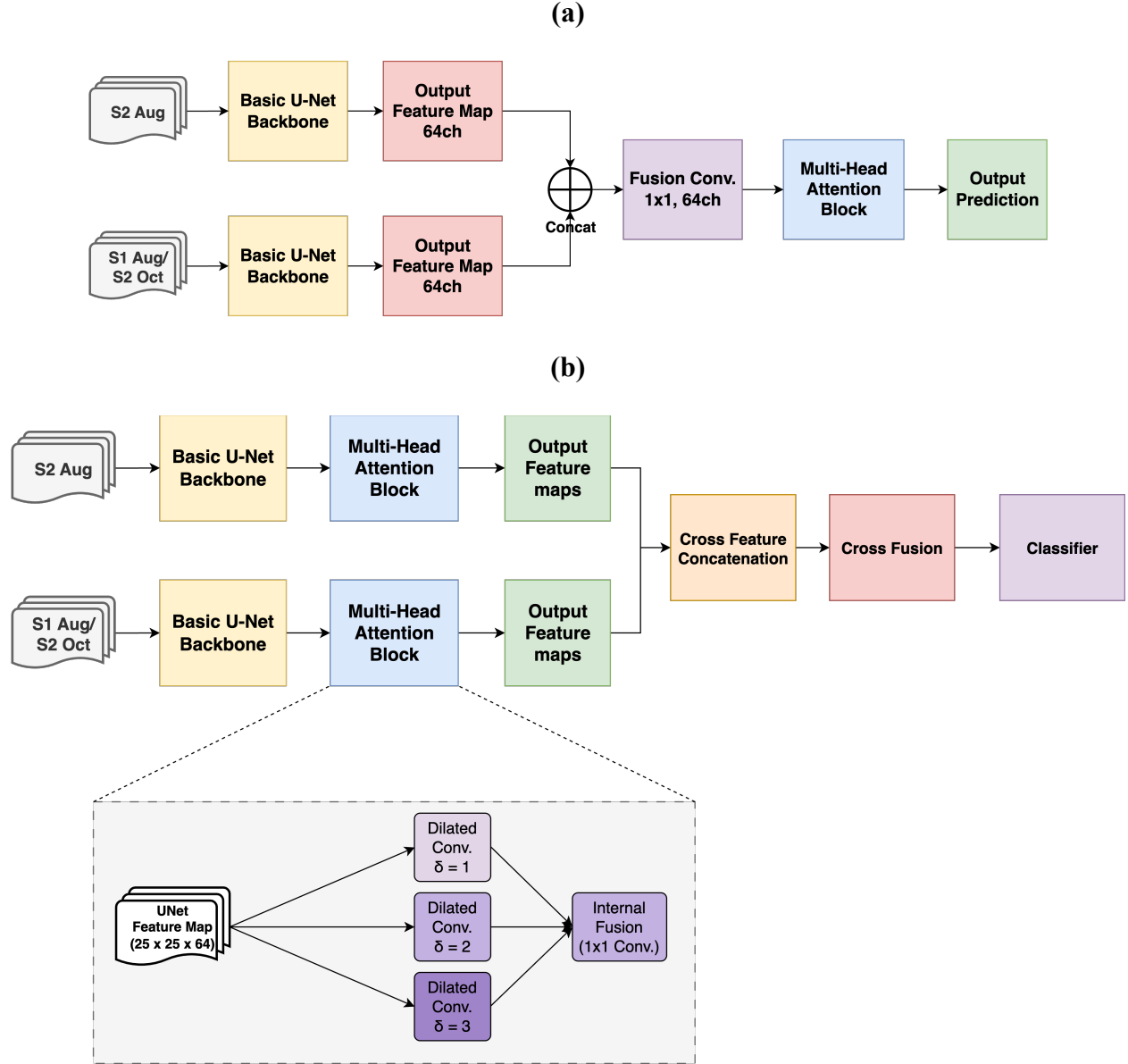


Figure 4.6. Diagrams of Middle Fusion strategies: (a) Symmetric Middle Fusion; (b) Convolutional Middle Fusion.

Sources were combined after separate feature extraction in Middle Fusion. Two variants were implemented.

Symmetric Middle Fusion (Figure 4.6(a)). Inputs A and B were processed independently by two identical BasicUNet backbones, each producing a decoder feature map of size  $64 \times 25 \times 25$ . The maps were concatenated along channels to  $128 \times 25 \times 25$ , reduced to  $64 \times 25 \times 25$  by a  $1 \times 1$  convolution with batch normalization and ReLU, and then passed to a shared Multi-Head

Block (Section 4.1) for multi-scale processing and classification. Three heads with dilation  $\delta = 1, 2, 3$  and per-head classifiers were used in the Multi-Head Block, producing per-head logits of size  $7 \times 25 \times 25$ . Attention-weighted logit fusion was performed using softmax-normalized scalar weights with L2 and KL regularization. Convolutional weights were initialized with Kaiming initialization, and the batch-normalization scale was initialized to 1.

Convolutional Middle Fusion (Figure 4.6(b)). A BasicUNet backbone followed by a branch-local Multi-Head Block was applied within each branch. Within each branch, the three head features were concatenated and compacted by an internal  $1 \times 1$  fusion to  $64 \times 25 \times 25$ . The two branch outputs were concatenated to  $128 \times 25 \times 25$  and integrated by a Cross-Fusion module composed of a  $3 \times 3$  convolution that modeled spatial interactions at  $128 \times 25 \times 25$  and a  $1 \times 1$  convolution that reduced the channels to  $64 \times 25 \times 25$ . The final classifier applied a  $3 \times 3$  layer to 32 channels with dropout 0.1, followed by a  $1 \times 1$  layer to seven classes.

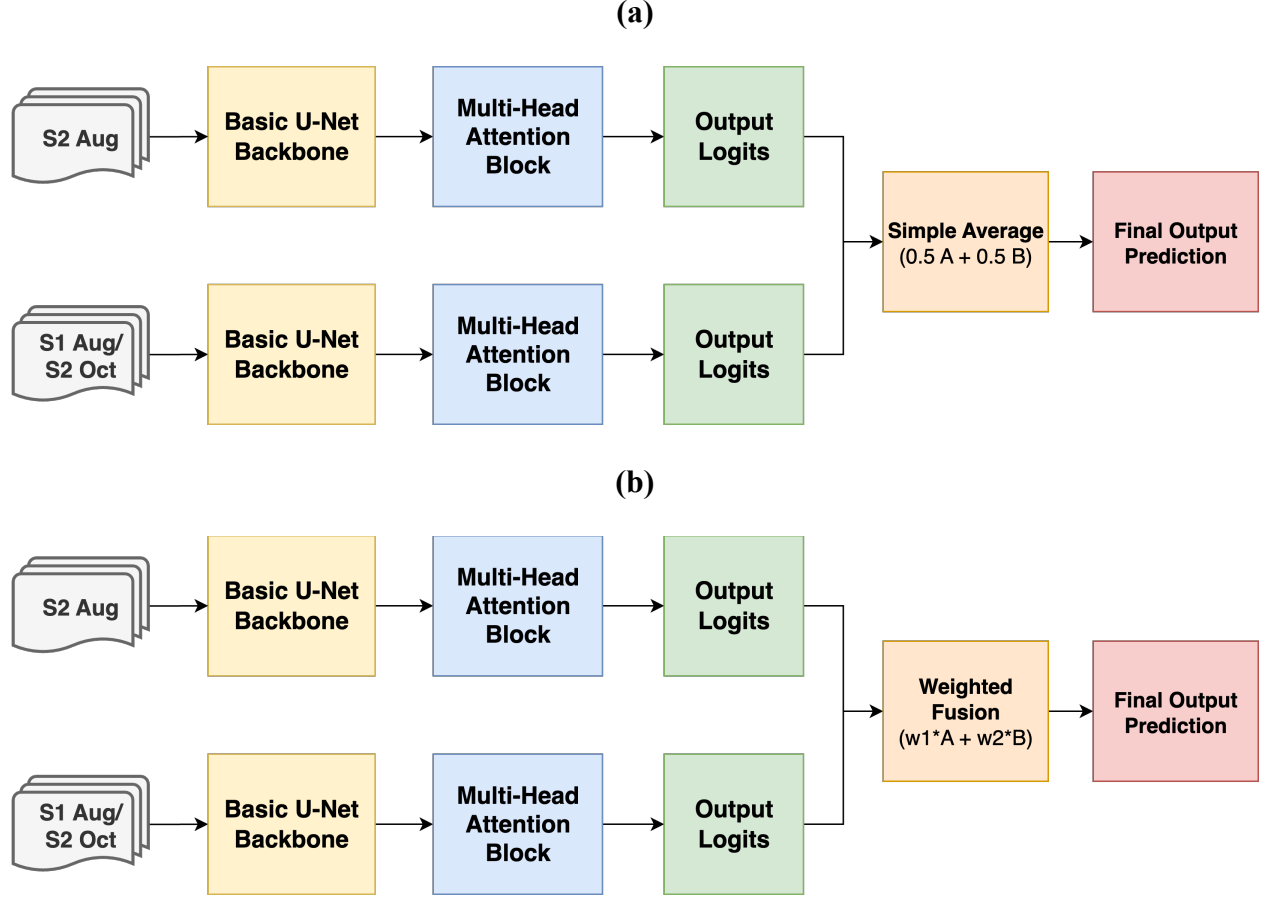


Figure 4.7. Diagrams of Late Fusion variants: (a) Simple-Average Late Fusion, where output logits are averaged; (b) Learnable-Weighted Late Fusion, where output logits are combined using learnable weights.

Late Fusion variants are summarized in Figure 4.7. In this strategy, sources were kept separate throughout training, and only the predictions (logits) were merged at the final stage. Two variants were implemented. The first, Simple Average (Figure 4.7(a)), produced per-pixel logits from two independent MHEnc branches and averaged them as  $1/2 (Z_A + Z_B)$ . The loss was computed on the averaged logits, and branch-internal attention-fusion regularization terms were accumulated. The second, Learnable Weighted (Figure 4.7(b)), also produced per-pixel logits  $\mathbf{Z}_A$  and  $\mathbf{Z}_B$  from two MHEnc branches. Learnable raw weights  $\mathbf{w} = [w_A, w_B]$  were mapped by softmax to  $\alpha = [\alpha_A, \alpha_B]$ , and fused logits were computed as  $\mathbf{Z}_{\text{fused}} = \alpha_A \mathbf{Z}_A + \alpha_B \mathbf{Z}_B$ . The loss

was defined as  $\mathcal{L} = \mathcal{L}_{\text{comp}}(\mathbf{Z}_{\text{fused}}) + \alpha_{\text{reg}} R(\mathbf{w})$ , where  $R(\mathbf{w})$  included an L2 term and a KL term toward the uniform weight. Both branches were updated through gradients from  $\mathcal{L}$  via  $\mathbf{Z}_A$  and  $\mathbf{Z}_B$ , but no features were exchanged between branches.

### 4.3 Design of the Hybrid Fusion Strategy

The hybrid fusion architecture for combining temporal S2 imagery with S1 SAR is described in this section.

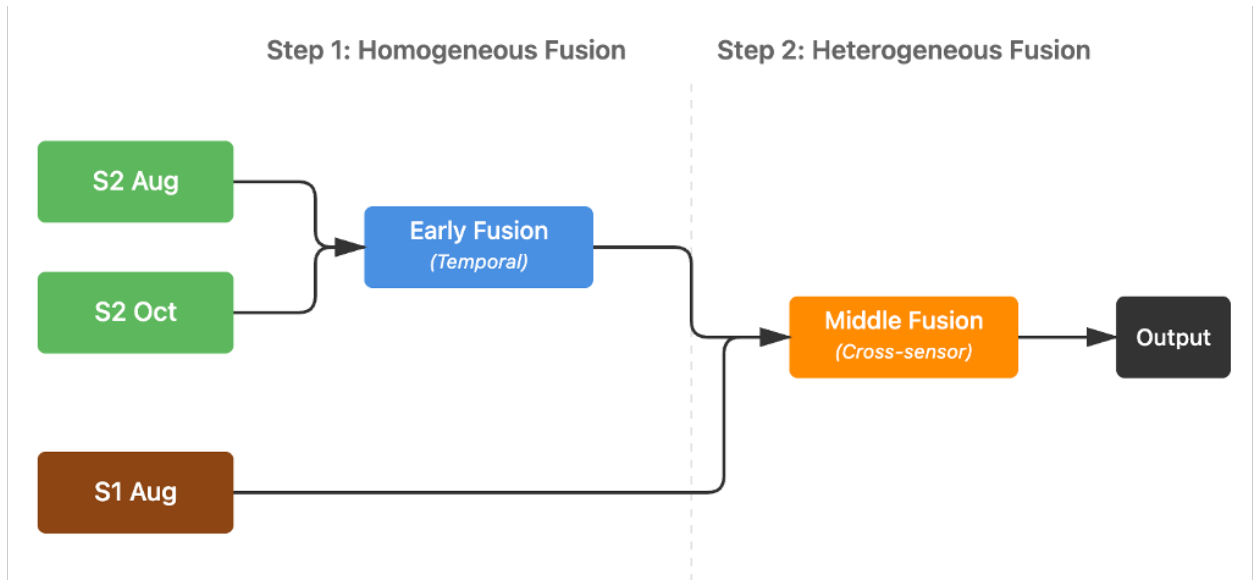


Figure 4.8. Hybrid Fusion architecture.

Two inputs were defined (Figure 4.8). Input A was formed by concatenating S2-Aug and S2-Oct, producing a 20-channel optical input ( $10 \text{ bands} \times 2 \text{ dates}$ ). Input B was composed of S1-Aug (VV and VH), producing a 2-channel SAR input.

Each input was processed by an independent shallow BasicUNet backbone. The decoder output of each backbone had dimensions  $64 \times 25 \times 25$ . Within each branch, a branch-local Multi-Head Block was applied to perform three parallel  $3 \times 3$  convolutions with dilation rates  $\delta =$

1, 2, 3 to extract local, intermediate-scale, and broader-context features. The three outputs were concatenated and compacted by an internal  $1 \times 1$  fusion to  $64 \times 25 \times 25$ .

The two branch features were concatenated to  $128 \times 25 \times 25$  and passed to a Cross-Fusion module. A  $3 \times 3$  convolution was applied to capture spatial interactions between optical and SAR features while preserving 128 channels, followed by a  $1 \times 1$  convolution that reduced the representation to  $64 \times 25 \times 25$ . The classifier was composed of a  $3 \times 3$  convolution that reduced channels from 64 to 32 with dropout 0.1, followed by a  $1 \times 1$  convolution that mapped to seven classes.

## 4.4 Experimental Setup

### 4.4.1 Dataset Preparation and Splitting

To limit spatial autocorrelation, the training and validation area was partitioned into a checkerboard pattern of  $100 \times 100$ -pixel blocks. Alternating blocks were assigned to the training or validation sets based on their center coordinates. This block-level assignment ensures that  $25 \times 25$ -pixel training patches are drawn from spatially distinct regions from validation patches, preventing them from crossing the split boundary. The 100-pixel block dimension was selected as a conservative measure, representing four times the linear dimension of the input patches (25 pixels). This separation scale was judged to be substantially larger than the model's potential effective receptive field—particularly given the use of dilated convolutions up to  $\delta = 3$  in the MHEnc framework—thereby minimizing information leakage from shared local context and ensuring a more robust evaluation of model generalization.

Within each set,  $25 \times 25$ -pixel patches were extracted with a stride of 20 pixels, and a 5-pixel overlap was produced to ensure dense coverage. Patches containing at least 50% NoData pixels were excluded.

After the initial checkerboard assignment, selected blocks were reassigned to achieve class balance while preserving spatial independence. The target was 70% of each class in the training set, with  $\pm 5$  percentage points tolerance. Reassignment was constrained as follows: a maximum of 20% of blocks were eligible for reassignment; each block was moved at most twice; and a maximum of 10 iterations was permitted to avoid an infinite loop. Candidate blocks were selected to minimize deviation from target proportions, with priority given to underrepresented classes. The process was terminated when all classes were within tolerance or when constraints had been reached.

Table 4.1. Class distribution between the training and validation sets.

| <b>Class</b>      | <b>Training Set (%)</b> | <b>Validation Set (%)</b> |
|-------------------|-------------------------|---------------------------|
| 0 Water           | 71.2                    | 28.8                      |
| 1 Urban/Developed | 68.5                    | 31.5                      |
| 2 Wetland         | 72.1                    | 27.9                      |
| 3 Agriculture     | 69.8                    | 30.2                      |
| 4 Deciduous       | 70.3                    | 29.7                      |
| 5 Mixed           | 71.6                    | 28.4                      |
| 6 Coniferous      | 69.2                    | 30.8                      |

The final split contained 46,209 training patches (64.5%) and 25,454 validation patches (35.5%). The class distribution between the training and validation sets is reported in Table 4.1. The 70% training target was achieved by all classes within  $\pm 5$  percentage points.

#### 4.4.2 Training Procedures

Models were implemented in PyTorch. Training was performed on an NVIDIA RTX 6000 Ada GPU with 48 GB of memory.

Adam with decoupled weight decay (AdamW) was used for optimization with an initial learning rate of  $1 \times 10^{-4}$  and a weight decay of  $1 \times 10^{-4}$ . In AdamW, weight decay is applied separately from gradient-based updates, and improvements in regularization stability have been



reported (Loshchilov and Hutter, 2019). The momentum parameters were set to  $\beta_1 = 0.9$  and  $\beta_2 = 0.999$ , following standard practice.

Class imbalance was addressed through weighted sampling at the patch level. For each class, weights inversely proportional to pixel counts were computed and were normalized by the mean across classes. Each patch was assigned the maximum weight among the classes present within it. Training samples were drawn with replacement according to these patch weights, maintaining  $N$  samples per epoch while increasing the sampling frequency of patches containing rare classes. Patches dominated by common classes were sampled less frequently or were omitted in individual epochs.

Data augmentation was applied during training with random transformations for each mini-batch. Geometric transformations included horizontal flips, vertical flips, 90 ° rotations, elastic deformations, and grid distortions. Radiometric transformations consisted of brightness and contrast adjustments within  $\pm 20\%$  and additive Gaussian noise with variance in the range 10-50. All augmentations were applied after normalization. These transformations were designed to simulate plausible geometric and radiometric variations while reducing overfitting to specific acquisition conditions.

#### **4.4.3 Edge Effect Investigation**

To investigate the patch boundary effects detailed in Section 2.6, the impact of restricting loss computation to patch interiors was assessed. Two training configurations were compared. In the central-region training, loss was computed only on the central 20x20 pixels of each 25x25 patch. The full patch was retained as input to preserve contextual information, but boundary pixels were excluded from gradient computation. In the full-patch training, loss was computed

on all 25x25 pixels. In both configurations, NoData pixels were masked from the loss calculation as described in Section 4.1.

Inference employed a sliding-window procedure matched to the training configuration. For central-region models, only the central 20x20 predictions from each window were retained, with a 10-pixel stride producing four-fold coverage for interior pixels. For full-patch models, the entire 25x25 window was used with a 12-pixel stride. Overlapping predictions were averaged uniformly. Reflection padding was applied at image boundaries to prevent boundary artifacts.

Model capacity, optimizer settings, learning schedule, data augmentation, and patch extraction remained constant across configurations. Through this controlled design, performance differences were attributable to the boundary-handling strategy.

## **4.5 Model Evaluation**

### **4.5.1 Comparison with existing models**

Under controlled conditions, the performance of the MHEnc framework was assessed by implementing two baseline architectures with identical preprocessing, training, and inference settings.

BasicUNet was configured to follow the U-Net paradigm (Ronneberger et al., 2015) using the same shallow backbone described in Section 4.1. The decoder output was mapped to class logits by a single  $1\times 1$  convolution, without multi-scale processing or attention. The contribution of the multi-head component in MHEnc was isolated by this baseline.

CBAMUNet was constructed by extending BasicUNet with Convolutional Block Attention Module components (Woo et al., 2018) at three locations while keeping the same shallow U-Net backbone: after the first encoder block at  $25\times 25$  resolution, at the  $12\times 12$  bottleneck, and after the decoder block before the  $1\times 1$  classifier. Within each CBAM, channel

attention was computed from global average pooling and global max pooling followed by a shared two-layer perceptron (MLP) with reduction ratio 16, and spatial attention was computed by a  $7 \times 7$  convolution over pooled channel statistics. Skip connections were used to carry pre-CBAM encoder features, so attention-refined features were delivered to the decoder along the main path while unmodified high-resolution details were provided through the shortcuts.

#### 4.5.2 Evaluation Metrics

Model performance was assessed using a suite of standard metrics to provide both global and class-specific insights. All metrics were computed excluding NoData pixels from the reference map.

Overall Accuracy (OA) was used to summarize global classification performance as shown in Equation (4.6):

$$OA = \frac{\text{Number of Correctly Classified Pixels}}{\text{Total Number of Valid Pixels}} \quad (4.6)$$

While intuitive, OA can be dominated by frequent classes and therefore may obscure errors on rare classes.

Chance agreement was accounted for by Cohen's Kappa ( $\kappa$ ):

$$\kappa = \frac{p_o - p_e}{1 - p_e} \quad (4.7)$$

where  $p_o$  denotes the observed agreement (equivalent to OA) and  $p_e = \sum_{i=1}^7 p_{i1} \times p_{i2}$ , with  $p_{i1}$  and  $p_{i2}$  denoting the proportions of class  $i$  in the reference and prediction maps, respectively.

Mean Intersection-over-Union (mIoU) was used as a stringent overlap-based measure.

For class  $c$ ,

$$\text{IoU}_c = \frac{\text{TP}_c}{\text{TP}_c + \text{FP}_c + \text{FN}_c} \quad (4.8)$$

where  $\text{TP}_c$ ,  $\text{FP}_c$ , and  $\text{FN}_c$  are true positives, false positives, and false negatives for class  $c$ .

The mean across the seven classes yields mIoU.

Class-wise metrics were used to provide additional detail. Precision and Recall for class  $c$  were defined as

$$\text{Precision}_c = \frac{\text{TP}_c}{\text{TP}_c + \text{FP}_c} \quad (4.9)$$

$$\text{Recall}_c = \frac{\text{TP}_c}{\text{TP}_c + \text{FN}_c} \quad (4.10)$$

The class-wise F1-Score was defined as the harmonic mean of Precision and Recall:

$$\text{F1}_c = 2 \times \frac{\text{Precision}_c \times \text{Recall}_c}{\text{Precision}_c + \text{Recall}_c} \quad (4.11)$$

Validation performance was monitored after each epoch using these metrics to track training progress and to identify overfitting. The checkpoint with the highest validation mIoU was selected for final evaluation. The final validation performance for each major experimental configuration, which formed the basis for this selection, is reported in Appendix A.5. Final testing was conducted on the test area using the sliding-window protocol described in Section 4.4.3. Together, these metrics were used to provide global and class-specific assessments that support interpretation of results and guidance for subsequent analyses.

## Chapter 5 Experimental Results

Quantitative and qualitative results are reported for the test area. The MHEnc framework is evaluated and fusion design principles are assessed.

### 5.1 Single-Source Performance of MHEnc and Baselines

Unless noted, all experiments in this section were conducted using the central-region training configuration described in Section 4.4.3, in which loss was computed on the central  $20 \times 20$  pixels of each  $25 \times 25$  patch and inference followed the matched sliding-window protocol. Metrics were computed on the test area with NoData pixels excluded. OA was rounded to two decimals, and mIoU and  $\kappa$  were rounded to four decimals. Performance differences are reported in percentage points (pp). When comparing single-source configurations, mIoU was taken as the primary metric.

Table 5.1. Single-source performance on the test area

| <b>Model Architecture</b> | <b>Data Source</b> | <b>OA (%)</b> | <b>Kappa (<math>\kappa</math>)</b> | <b>mIoU</b> |
|---------------------------|--------------------|---------------|------------------------------------|-------------|
| BasicUNet                 | S2-Aug             | 87.72         | 0.8349                             | 0.6053      |
| CBAMUNet                  | S2-Aug             | 90.57         | 0.8724                             | 0.6403      |
| MHEnc                     | S2-Aug             | 90.88         | 0.8768                             | 0.6507      |
| BasicUNet                 | S2-Oct             | 89.80         | 0.8634                             | 0.7232      |
| CBAMUNet                  | S2-Oct             | 73.43         | 0.6657                             | 0.6082      |
| MHEnc                     | S2-Oct             | 92.59         | 0.9003                             | 0.7723      |
| BasicUNet                 | S1-Aug             | 36.74         | 0.2676                             | 0.2073      |
| CBAMUNet                  | S1-Aug             | 67.88         | 0.5854                             | 0.3816      |
| MHEnc                     | S1-Aug             | 64.88         | 0.5518                             | 0.3782      |

Performance for single-source imagery is reported in Table 5.1. With S2-Oct, MHEnc achieved 0.7723 mIoU and 92.59% OA. These results represent a clear improvement over CBAMUNet (0.6082 mIoU, 73.43% OA) and BasicUNet (0.7232 mIoU, 89.80% OA), yielding

an increase of +0.0491 mIoU and +2.79 pp in OA over the BasicUNet. The performance of each architecture across the datasets is visually summarized in Figure 5.1.

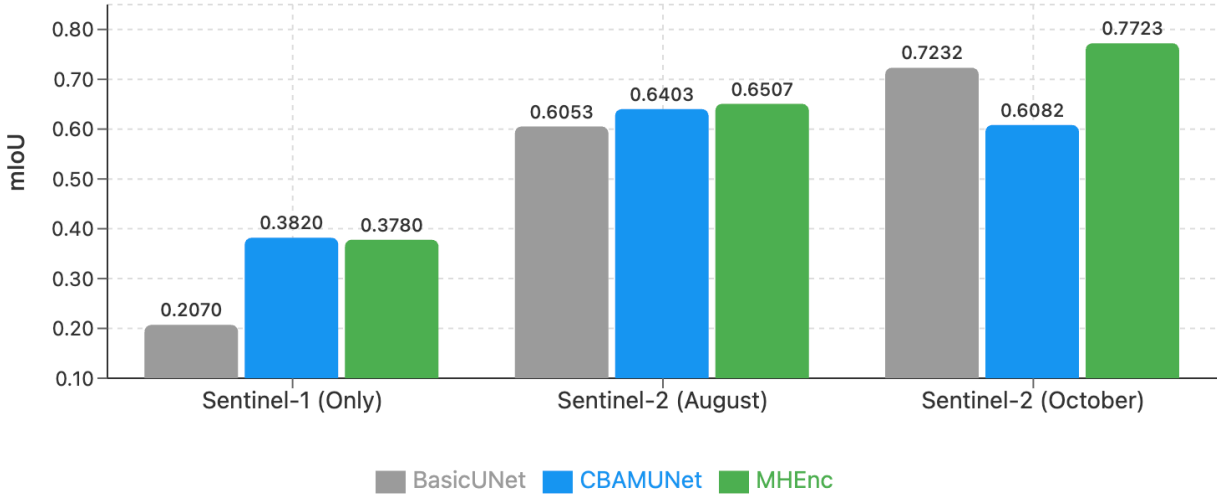


Figure 5.1. Comparison of single-source performance (mIoU) on the test area

For S2-Aug, performance differences among architectures were smaller. MHEnc (0.6507 mIoU, 90.88% OA) again achieved the highest mIoU, representing a marginal gain of +0.0104 mIoU over CBAMUNet (0.6403 mIoU, 90.57% OA) and a more substantial gain of +0.0454 mIoU over BasicUNet (0.6053 mIoU, 87.72% OA). In terms of Overall Accuracy, MHEnc also outperformed CBAMUNet by +0.31 pp and BasicUNet by +3.16 pp. The improved performance with S2-Oct across models was interpreted as being consistent with post-senescence spectral characteristics, which capture increased separability for seasonally responsive classes.

With S1-Aug, 0.2073 mIoU and 36.74% OA were obtained by BasicUNet, whereas higher accuracies were recorded by attention-based architectures. CBAMUNet (0.3816 mIoU) and MHEnc (0.3782 mIoU) achieved substantially higher mIoU scores (gains of +0.1743 and

+0.1709, respectively) compared to BasicUNet. For OA, MHEnc's 64.88% was also a substantial +28.14 pp higher than BasicUNet's 36.74%. These outcomes were consistent with single-date S1 SAR, where classification is challenging due to limited sensitivity, speckle noise, and the absence of multi-band spectral information.

Notably, while MHEnc maintained stable performance across both S2-Aug and S2-Oct imagery, the performance of CBAMUNet was observed to be unstable, showing a sharp decline on the S2-Oct dataset (OA −17.14 pp). This aspect is explored further in the Discussion (Section 6.1). Within the constraints of S1 data, utilization of informative textural features may be enhanced by attention mechanisms, though fundamental sensor limitations were not overcome.

Class-wise F1-scores, presented in Table 5.2, provide further insight into these performance differences.

Table 5.2. Class-wise F1-scores (%) on the test area for single-source imagery.

| Data Source | Model     | Water        | Urban/Dev    | Wetland      | Agriculture  | Deciduous    | Mixed        | Coniferous   |
|-------------|-----------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| S2-Aug      | BasicUNet | 95.90        | 82.69        | 82.45        | 11.58        | 49.17        | 91.91        | 80.37        |
| S2-Aug      | CBAMUNet  | 98.17        | 81.37        | 88.13        | 14.19        | 52.02        | 93.42        | 85.06        |
| S2-Aug      | MHEnc     | 98.41        | 79.41        | <b>88.84</b> | 18.95        | 53.69        | <b>93.99</b> | 87.03        |
| S2-Oct      | BasicUNet | 99.66        | 91.37        | 77.71        | 85.92        | 47.14        | 82.61        | 90.54        |
| S2-Oct      | CBAMUNet  | 66.75        | 90.84        | 88.66        | 80.35        | 46.07        | 90.67        | 49.41        |
| S2-Oct      | MHEnc     | <b>99.77</b> | <b>93.87</b> | 85.76        | <b>88.05</b> | <b>54.61</b> | 87.99        | <b>90.58</b> |
| S1-Aug      | BasicUNet | 0.18         | 6.77         | 77.81        | 7.17         | 6.06         | 72.67        | 24.40        |
| S1-Aug      | CBAMUNet  | 90.47        | 40.72        | 81.63        | 35.71        | 3.09         | 51.33        | 48.72        |
| S1-Aug      | MHEnc     | 78.18        | 23.79        | 86.26        | 42.90        | 4.42         | 69.57        | 44.11        |

For the Urban/Developed class, enhanced optical contrast in the leaf-off season appeared beneficial. MHEnc's F1-Score improved by +14.46 pp, from 79.41% on S2-Aug to 93.87% on S2-Oct, potentially due to reduced foliage obscuring roads and roofs.

Strong seasonal effects for Agriculture were observed. In S2-Aug, all three models achieved F1-Scores below 20%. These values increased dramatically in S2-Oct. For MHEnc, the



F1-Score rose from 18.95% to 88.05%, a gain of 69.10 pp. This large improvement was consistent with post-harvest conditions that increase separability.

For the Wetland class, the structural information captured by SAR proved particularly effective. MHEnc achieved a high F1-Score of 86.26% using only S1-Aug data. The difference between this SAR-only score and the best optical score (S2-Aug, 88.84%) was merely 2.58 pp, demonstrating radar's robust discrimination for this class. Compared to BasicUNet, MHEnc showed a notable gain of +8.45 pp on S1-Aug data for Wetland detection.

Forest classification showed different patterns. For Mixed Forest with S2-Aug, the highest score was achieved by MHEnc at 93.99%. However, with S2-Oct, 90.67% was achieved by CBAMUNet, which exceeded the MHEnc value (87.99%) by 2.68 pp. Deciduous Forest remained challenging, with MHEnc achieving the highest (though modest) optical F1-Scores (53.69% and 54.61%).

Significant date-dependent variability was observed for CBAMUNet. As noted, its OA dropped sharply on S2-Oct, driven by F1-Score declines of −31.42 pp for Water and −35.65 pp for Coniferous Forest. This instability contrasts sharply with MHEnc, where performance changes between dates were marginal (e.g., +1.36 pp for Water, +3.55 pp for Coniferous Forest). BasicUNet also exhibited changes, though less extreme, with increases of +3.76 pp for Water and +10.17 pp for Coniferous Forest. This variability resulted in major performance discrepancies on S2-Oct data, where the gap between the best (MHEnc) and worst (CBAMUNet) F1-Scores reached 33.02 pp for Water and 41.17 pp for Coniferous Forest. This is attributed to the seasonal instability of CBAMUNet compared to the relative stability of MHEnc and BasicUNet.

Considering the best performing date for each class, MHEnc achieved the highest F1-Scores across all seven target categories. While some advantages were modest, MHEnc consistently demonstrated robust performance, ranking first for six classes on S2-Aug and five on S2-Oct. For example, on S2-Oct, its Urban/Developed score was +2.50 pp higher than BasicUNet (93.87% vs 91.37%), and on S2-Aug, its Mixed Forest score was +0.57 pp higher than CBAMUNet (93.99% vs 93.42%).

Table 5.3. Learned attention weights ( $\alpha$ ) for the parallel multi-scale heads. Head-k uses dilation rate  $\delta = k$ .

| <b>Multi-scale Head</b> | <b>S1-Aug</b> | <b>S2-Aug</b> | <b>S2-Oct</b> | <b>All Sources</b> |
|-------------------------|---------------|---------------|---------------|--------------------|
| Head-1 ( $\delta=1$ )   | 0.2627        | 0.2976        | 0.2678        | 0.3062             |
| Head-2 ( $\delta=2$ )   | 0.2798        | 0.2927        | 0.2944        | 0.3031             |
| Head-3 ( $\delta=3$ )   | 0.4575        | 0.4097        | 0.4378        | 0.3907             |

A consistent and dominant pattern was observed across all single-source configurations in Table 5.3: the head with the largest dilation rate (Head-3,  $\delta = 3$ ) was assigned a significantly higher attention weight (ranging from 0.4097 to 0.4575) compared to the other heads. The recorded weights were 0.4575 for S1-Aug, 0.4097 for S2-Aug, and 0.4378 for S2-Oct.

A hierarchy among scales was observed in the weight distribution: Head-1 recorded the lowest weights, Head-2 recorded intermediate weights, and Head-3 recorded the highest weights. Notably, the variation in these learned weights across the different data sources was minimal. This suggests that the observed preference for broader spatial context (Head-3) is likely an intrinsic characteristic of the LULC patterns in the study area, rather than being specific to a particular sensor modality or acquisition time.

A more balanced distribution was observed for S2-Aug. The range between the highest and lowest weights was 0.1121 (0.4097 vs 0.2976), compared with 0.1948 for S1-Aug and 0.1700 for S2-Oct.

Evidence of temporal complementarity between S2-Aug and S2-Oct was observed. While limited overall performance was recorded for S1-Aug, useful information for specific classes such as Wetland was still provided. Stable performance across sources was observed for MHEnc, supporting its use as the backbone.

## 5.2 Results from Different Fusion Strategies

After the single-source experiments, fusion strategies were tested using the same MHEnc backbone. Two scenarios were examined: heterogeneous sensor fusion (S1-Aug + S2-Aug) and homogeneous temporal fusion (S2-Aug + S2-Oct). To identify image regions associated with each data source in the final predictions, visualizations were generated using the Grad-CAM methodology detailed in Section 2.7.4.

Table 5.4. Performance of fusion strategies on the test area.

| Data Combination                                 | Fusion Strategy                | OA (%)       | Kappa ( $\kappa$ ) | mIoU          |
|--------------------------------------------------|--------------------------------|--------------|--------------------|---------------|
| <b>Case 1: Sensor Fusion (S2 Aug + S1 Aug)</b>   |                                |              |                    |               |
| S2 Aug + S1 Aug                                  | Early Fusion                   | 91.64        | 0.8865             | 0.6669        |
| S2 Aug + S1 Aug                                  | <b>Symmetric Middle Fusion</b> | <b>93.81</b> | <b>0.9163</b>      | <b>0.7426</b> |
| S2 Aug + S1 Aug                                  | Convolutional Middle Fusion    | 92.74        | 0.9017             | 0.6932        |
| S2 Aug + S1 Aug                                  | Late Fusion (Simple Average)   | 92.60        | 0.8998             | 0.6805        |
| S2 Aug + S1 Aug                                  | Late Fusion (Weighted)         | 92.00        | 0.8917             | 0.6899        |
| <b>Case 2: Temporal Fusion (S2 Aug + S2 Oct)</b> |                                |              |                    |               |
| S2 Aug + S2 Oct                                  | <b>Early Fusion</b>            | <b>96.44</b> | <b>0.9518</b>      | <b>0.8425</b> |
| S2 Aug + S2 Oct                                  | Symmetric Middle Fusion        | 95.29        | 0.9361             | 0.8089        |
| S2 Aug + S2 Oct                                  | Convolutional Middle Fusion    | 95.89        | 0.9444             | 0.8117        |
| S2 Aug + S2 Oct                                  | Late Fusion (Simple Average)   | 95.69        | 0.9417             | 0.8130        |
| S2 Aug + S2 Oct                                  | Late Fusion (Weighted)         | 94.86        | 0.9304             | 0.7671        |

The optimal fusion stage was dependent on the data combination (Table 5.4). For heterogeneous fusion (S2-Aug + S1-Aug), the highest mIoU of 0.7426 with 93.81% OA was obtained by Symmetric Middle Fusion, compared with Early Fusion at 0.6669 mIoU and 91.64% OA. For temporal fusion (S2-Aug + S2-Oct), the best performance was obtained by Early Fusion with 0.8425 mIoU and 96.44% OA. Convolutional Middle Fusion yielded intermediate results in

both scenarios. Late Fusion strategies exhibited lower mIoU values: for sensor fusion, the Weighted variant reached 0.6899 mIoU while Simple-Average reached 0.6805 mIoU; for temporal fusion, both variants yielded lower mIoU and OA than Early Fusion. These outcomes are illustrated in Figure 5.2 and Figure 5.3, where it is shown that Symmetric Middle Fusion yielded the highest mIoU for the heterogeneous data (Figure 5.2), while Early Fusion produced the highest mIoU for the homogeneous temporal combination (Figure 5.3).

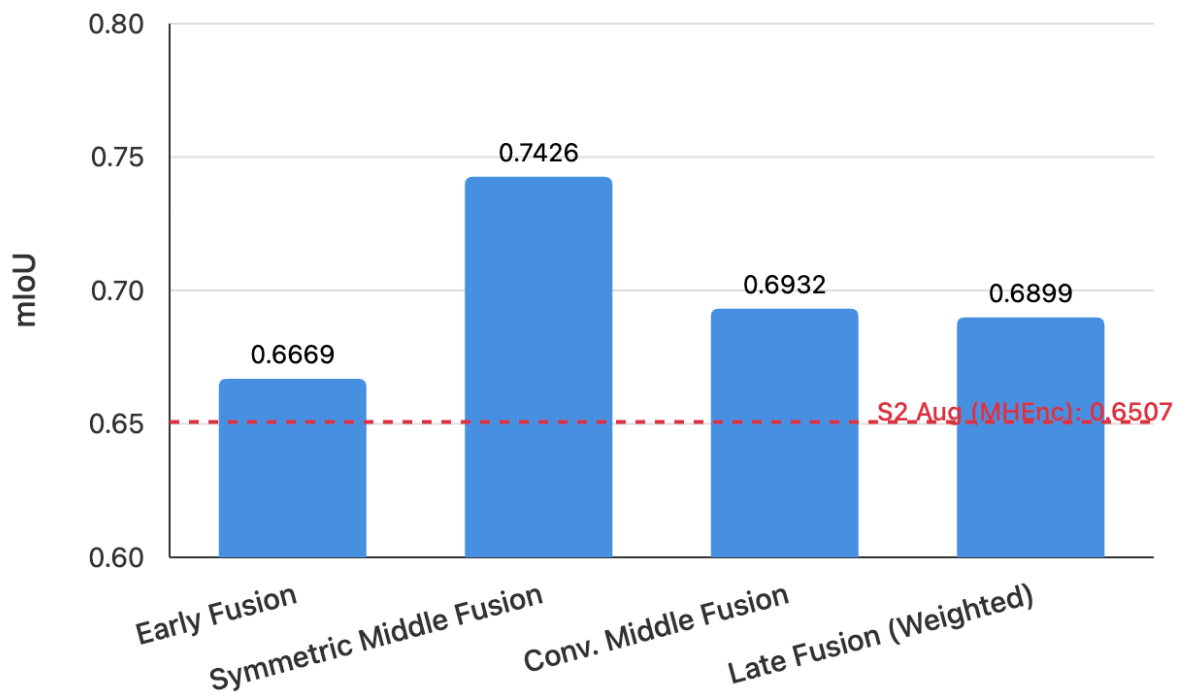


Figure 5.2. Performance of heterogeneous fusion strategies

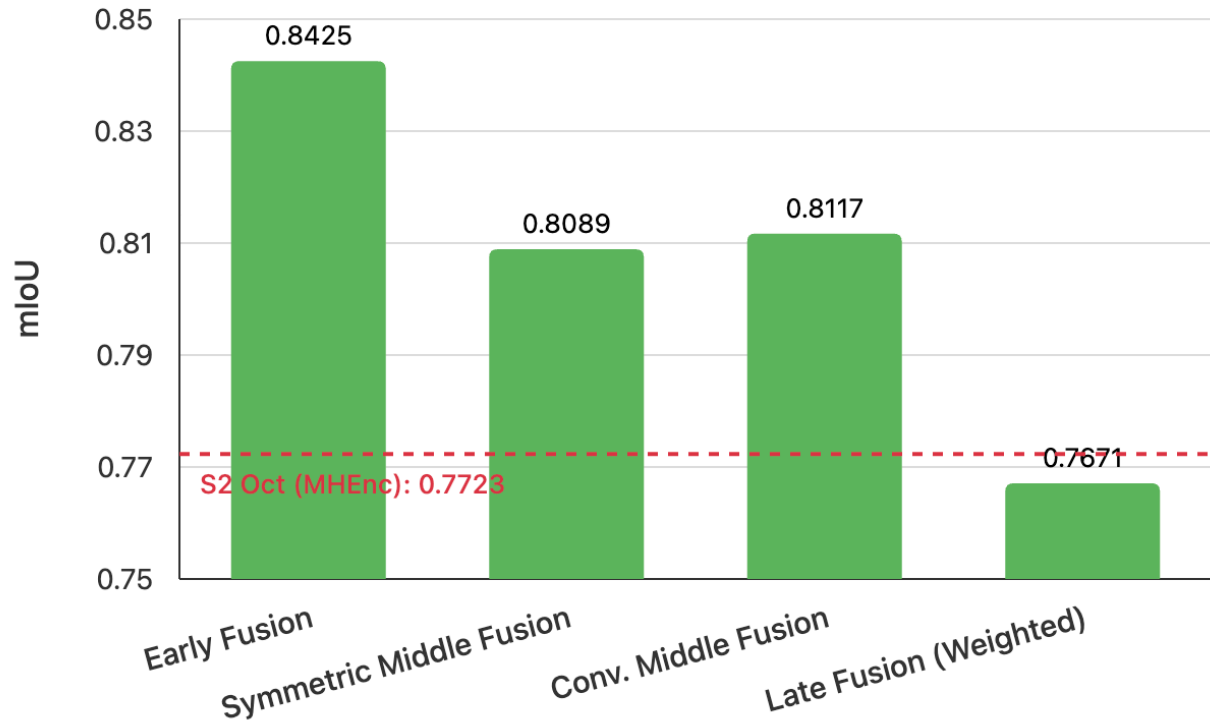


Figure 5.3. Performance of homogeneous fusion strategies

Table 5.5. Class-wise F1-Scores (%) for all fusion strategies on the test area.

| Data Combination                          | Fusion Strategy              | Water        | Urban/Dev    | Wetland      | Agriculture  | Deciduous    | Mixed        | Coniferous   |
|-------------------------------------------|------------------------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| Case 1: Sensor Fusion (S2 Aug + S1 Aug)   |                              |              |              |              |              |              |              |              |
| S2 Aug + S1 Aug                           | Early Fusion                 | 99.13        | 79.65        | 91.83        | 30.34        | 55.67        | 93.65        | 84.36        |
| S2 Aug + S1 Aug                           | Symmetric Middle Fusion      | <b>99.32</b> | 87.21        | 92.82        | <b>59.72</b> | 58.59        | 94.78        | <b>90.81</b> |
| S2 Aug + S1 Aug                           | Convolutional Middle Fusion  | 99.27        | 72.55        | <b>93.63</b> | 48.85        | <b>59.12</b> | <b>95.47</b> | 86.42        |
| S2 Aug + S1 Aug                           | Late Fusion (Simple Average) | 99.12        | 83.97        | 90.89        | 23.49        | 55.78        | 95.16        | 88.69        |
| S2 Aug + S1 Aug                           | Late Fusion (Weighted)       | 99.08        | <b>87.91</b> | 86.53        | 36.47        | 55.07        | 93.13        | 90.48        |
| Case 2: Temporal Fusion (S2 Aug + S2 Oct) |                              |              |              |              |              |              |              |              |
| S2 Aug + S2 Oct                           | Early Fusion                 | 99.52        | <b>93.39</b> | 97.94        | <b>85.96</b> | 68.27        | <b>96.20</b> | <b>93.19</b> |
| S2 Aug + S2 Oct                           | Symmetric Middle Fusion      | <b>99.64</b> | 92.63        | 96.85        | 78.61        | 67.19        | 94.48        | 90.05        |
| S2 Aug + S2 Oct                           | Convolutional Middle Fusion  | 99.54        | 93.24        | <b>98.14</b> | 75.22        | 64.65        | 95.89        | 92.03        |

|                 |                                 |       |       |       |       |              |       |       |
|-----------------|---------------------------------|-------|-------|-------|-------|--------------|-------|-------|
| S2 Aug + S2 Oct | Late Fusion<br>(Simple Average) | 99.57 | 92.94 | 97.06 | 76.40 | 67.91        | 95.45 | 91.48 |
| S2 Aug + S2 Oct | Late Fusion<br>(Weighted)       | 99.52 | 86.13 | 95.35 | 56.29 | <b>70.22</b> | 96.02 | 92.23 |

Class-specific results in Table 5.5 provide an explanation for the overall mIoU trends observed in Table 5.4. For heterogeneous sensor fusion (Case 1), Symmetric Middle Fusion's superior mIoU (0.7426) was driven by substantial performance gains in classes that were poorly resolved by Early Fusion. Specifically, the F1-Score for Agriculture increased significantly from 30.34% (Early) to 59.72% (Symmetric Middle), and Urban/Developed improved from 79.65% to 87.21%. This suggests that separate modality-specific encoding, as used in Middle Fusion, was critical for preserving the distinct features of these classes.

For homogeneous temporal fusion (Case 2), Early Fusion's highest overall mIoU (0.8425) was correlated with its consistent, high F1-Scores across most phenologically-sensitive classes, such as Agriculture (85.96%) and Mixed Forest (96.20%). While other strategies achieved isolated high scores—for instance, Late Fusion (Weighted) for Deciduous Forest (70.22%)—they failed to match the overall performance balance of Early Fusion. These findings support the interpretation that joint processing from the initial layer, as employed in Early Fusion, is the most effective strategy for integrating homogeneous temporal data.

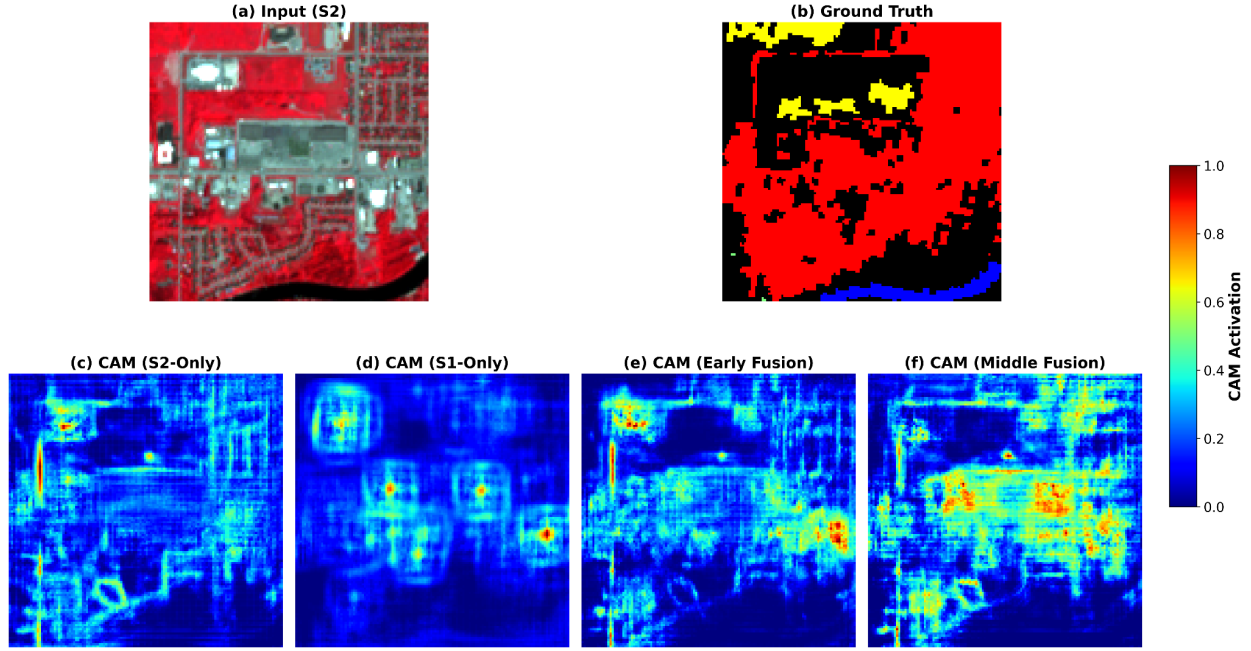


Figure 5.4. Heterogeneous sensor fusion analysis for the Urban/Developed class. (a) S2 false-color composite. (b) Ground-truth map where Urban/Developed areas are shown in red, Agriculture in yellow, and Water in blue. (c-f) Grad-CAM visualizations for S2-only, S1-only, Early Fusion, and Middle Fusion models, respectively.

Complementary cues for Urban/Developed were observed at the block-scale in Figure 5.4. All Grad-CAM maps were generated with the same normalization. In the S2-only map (c), high responses were aligned with the orthogonal road grid, whereas responses over large roofs in the central block remained weak. In the S1-only map (d), responses formed bright spots on rectangular building blocks in the central and right-central areas, while linear roads remained weak. Early Fusion (e) produced activations similar to the S2-only map, with strong roads and weak roof clusters. Middle Fusion (f) produced high responses over both the road network and the building clusters near the image center and right-center. This combined coverage was consistent with the Urban/Developed F1-Score increase from 79.65% (Early) to 87.21% (Symmetric Middle) and 87.91% (Late Fusion (Weighted)) in Table 5.5. Separate encoders followed by feature-level integration were found to preserve both roof-scale backscatter and road-scale reflectance prior to fusion.

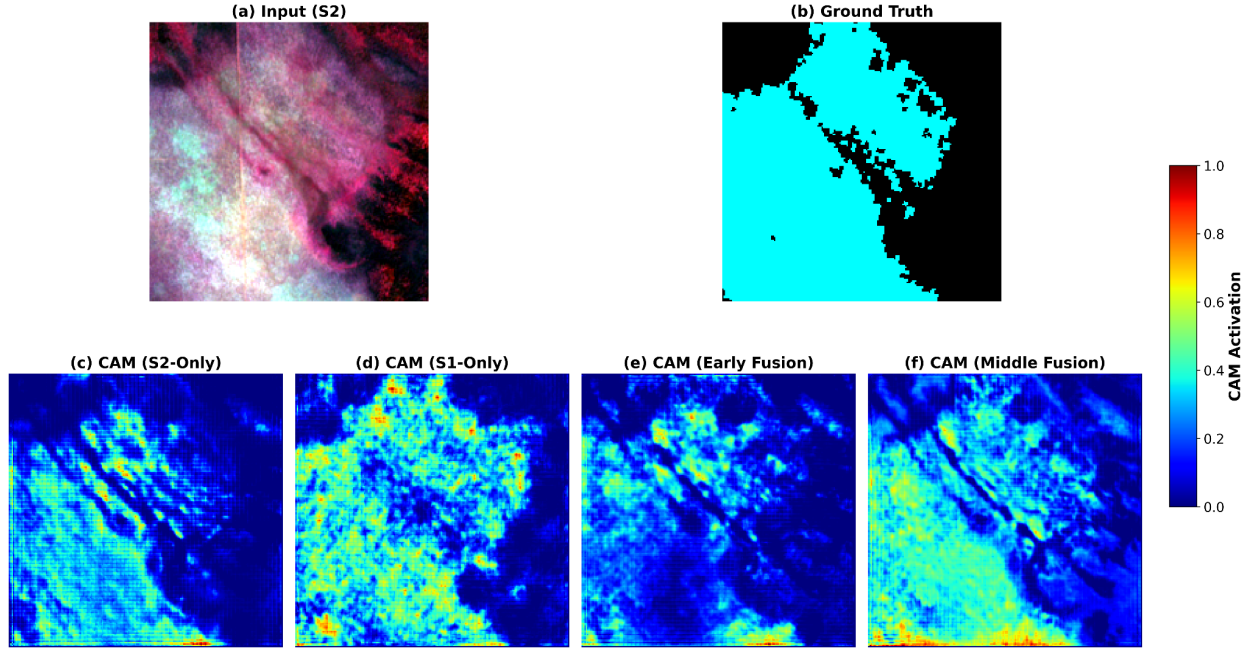


Figure 5.5. Heterogeneous sensor fusion analysis for the Wetland class. (a) S2 false-color composite. (b) Ground-truth map where Wetland is shown in cyan. (c-f) Grad-CAM visualizations for S2-only, S1-only, Early Fusion, and Middle Fusion models, respectively.

Wetland localization patterns differed across fusion strategies in Figure 5.5. In the S2-only map (c), responses were weak and scattered. The S1-only map (d) showed continuous strong lines along Wetland boundaries and strong hotspots inside the boundaries that matched the ground-truth map (b). Early Fusion (e) produced stronger Wetland signals than the S2-only map but with uneven interior coverage. Middle Fusion (f) showed continuous signals along the mapped edge with reduced background noise and more uniform interior coverage. The class F1-Scores reached 93.63% for Convolutional Middle Fusion in the sensor case and 98.14% for Convolutional Middle Fusion versus 97.94% for Early Fusion in the temporal case (Table 5.5).



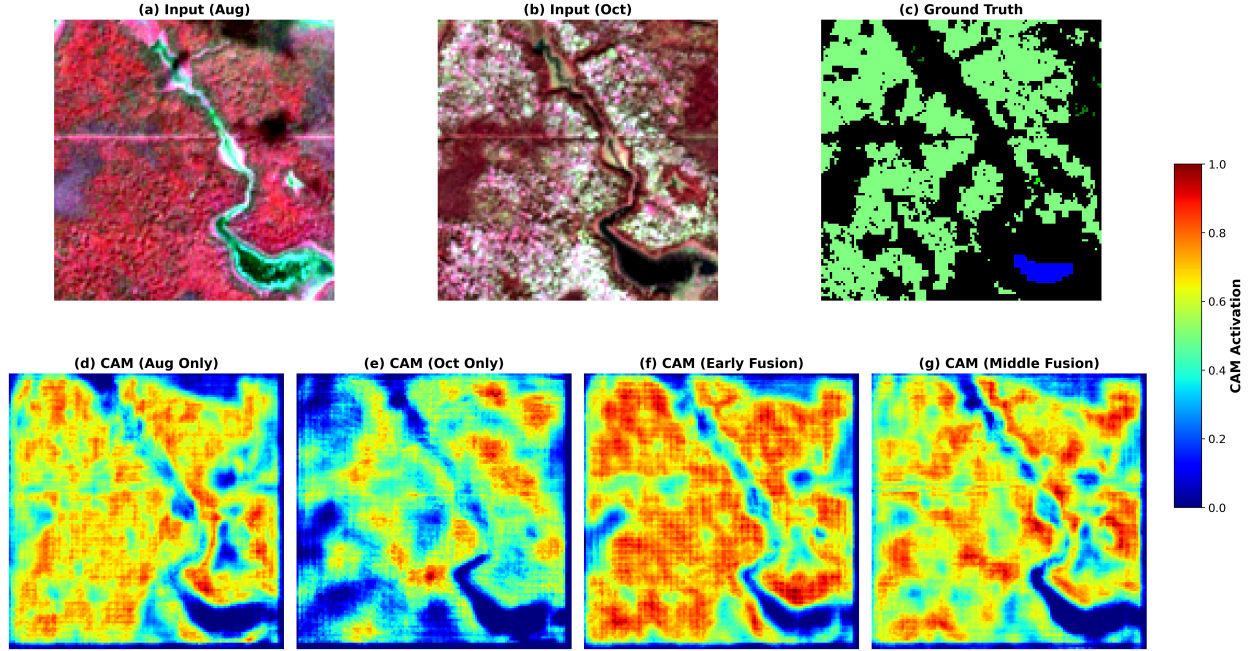


Figure 5.6. Homogeneous temporal fusion analysis for the Mixed Forest class. (a) S2-Aug false-color composite. (b) S2-Oct false-color composite. (c) Ground-truth map where Mixed Forest is shown in light green and Water in blue. (d-g) Grad-CAM visualizations for August-only, October-only, Early Fusion, and Middle Fusion models, respectively.

Temporal cues were observed to improve Mixed Forest discrimination in Figure 5.6. The Grad-CAM visualizations (d-g) exhibit a faint grid-like pattern. This is a visual artifact resulting from the sliding-window averaging technique required to generate large-scale heatmaps from a patch-based (25x25) model. The analysis, therefore, focuses on the relative, large-scale activation patterns rather than these low-level visualization artifacts. In the August-only map (d), activations were moderate with unclear boundaries at Mixed-Coniferous interfaces. The October-only map (e) leveraged senescence differences but still exhibited ambiguous responses in mixed stands. Early Fusion of the 20-channel input (f) produced stronger activations aligned with boundaries. Middle Fusion (g) merged features after separate encoding and produced weaker, more scattered responses. These patterns were consistent with the results in Table 5.5,

where Mixed Forest reached 96.20% F1 with Early Fusion, compared with 94.48% for Symmetric Middle Fusion and 95.89% for Convolutional Middle Fusion.

Taken together, a consistent pattern was indicated by the experiments. For heterogeneous sensor integration, better performance than Early Fusion was observed for Middle Fusion, which was consistent with the interpretation that separate encoders retained modality-specific structure prior to integration. For homogeneous temporal integration, better performance than Middle Fusion was observed for Early Fusion, which was consistent with the interpretation that direct concatenation enabled early layers to capture temporal differences. These principles were used in designing the Hybrid Fusion Strategy in Section 5.3. The Grad-CAM analysis was consistent with the quantitative findings, highlighting where each strategy drew information and how complementary cues were combined.

### **5.3 Performance of the Proposed Hybrid Fusion Strategy**

The Hybrid Fusion Strategy was implemented as a two-stage process: S2-Aug and S2-Oct were concatenated using Early Fusion, after which the resulting optical features were merged with S1-Aug SAR features using Middle Fusion.

The results were summarized in Table 5.6.  $OA = 96.83\%$ ,  $\kappa = 0.9572$ , and  $mIoU = 0.8665$  were achieved by the Hybrid Fusion Strategy. These metrics were computed from the confusion matrix using Eqs. (4.6)-(4.8); class-wise Precision and Recall in Table 5.6 were computed using Eqs. (4.9)-(4.10).

Table 5.6. Confusion matrix of the proposed Hybrid Fusion Strategy on the test area (pixel counts). (OA = 96.83%,  $\kappa$  = 0.9572, mIoU = 0.8665)

| <i>Reference</i>            | <b>Predicted</b> |                |                |                |               |                  |                | <b>Recall (%)</b> |
|-----------------------------|------------------|----------------|----------------|----------------|---------------|------------------|----------------|-------------------|
|                             | Water            | Urban/Dev      | Wetland        | Agriculture    | Deciduous     | Mixed            | Coniferous     |                   |
| <i>Water</i>                | <b>1,830,423</b> | 171            | 1,420          | 18             | 29            | 598              | 15,709         | <b>99.0</b>       |
| <i>Urban/Dev</i>            | 454              | <b>231,924</b> | 1,634          | 5,726          | 1,745         | 5,919            | 5,078          | <b>91.9</b>       |
| <i>Wetland</i>              | 29               | 20             | <b>544,664</b> | 33             | 28            | 874              | 2,240          | <b>99.4</b>       |
| <i>Agriculture</i>          | 1                | 4,590          | 10,595         | <b>128,048</b> | 10,004        | 2,141            | 120            | <b>82.3</b>       |
| <i>Deciduous</i>            | 0                | 14             | 110            | 34             | <b>51,544</b> | 3,951            | 22             | <b>92.6</b>       |
| <i>Mixed</i>                | 2                | 126            | 2,072          | 26             | 16,456        | <b>1,164,119</b> | 8,751          | <b>97.7</b>       |
| <i>Coniferous</i>           | 0                | 93             | 627            | 46             | 4             | 44,889           | <b>524,013</b> | <b>92.0</b>       |
| <b><i>Precision (%)</i></b> | <b>100.0</b>     | <b>97.8</b>    | <b>97.1</b>    | <b>95.6</b>    | <b>64.6</b>   | <b>95.2</b>      | <b>94.3</b>    |                   |

Diagonal entries were dominant in Table 5.6, with high per-class accuracy recorded.

Water achieved 100.0% precision, while Urban/Developed and Wetland reached 97.8% and 97.1% precision, respectively. Limited confusion occurred between Water and Wetland: 1,420 Water pixels (0.08%) were classified as Wetland and 29 Wetland pixels (0.005%) were classified as Water. This reduced cross-class error was consistent with the interpretation that the combination of S1 backscatter and S2 reflectance improved separation of spectrally similar water-wetland surfaces.

Agriculture exhibited notable confusions: 10,595 pixels (6.8%) were classified as Wetland and 10,004 pixels (6.4%) were classified as Deciduous Forest. These errors were interpreted as potentially arising from post-harvest soil moisture producing wetland-like SAR responses and from mixed signatures at field-forest transitions. Agriculture achieved 82.3%

recall and 95.6% precision, producing an F1-Score of approximately 88%, consistent with temporal gains observed in Section 5.2.

Asymmetric patterns were observed in forest discrimination. For Coniferous Forest, 44,889 pixels (7.9%) were classified as Mixed Forest, whereas only 8,751 Mixed Forest pixels (0.7%) were classified as Coniferous. This asymmetry was interpreted as being associated with spectral and structural mixing near stand boundaries, with 10 m resolution limiting the capture of pure stand signals. Additionally, 16,456 Mixed Forest pixels (1.4%) were classified as Deciduous Forest.

Deciduous Forest remained the most challenging class, with 64.6% precision and 92.6% recall. The low precision was attributed to many non-deciduous pixels, particularly Mixed Forest, being classified as Deciduous. This pattern was interpreted as being associated with overlapping spectral and structural signatures of transitional forest types in the boreal environment. The F1-Score of approximately 76% represented an improvement over single-source cases but remained lower than that of other forest classes.

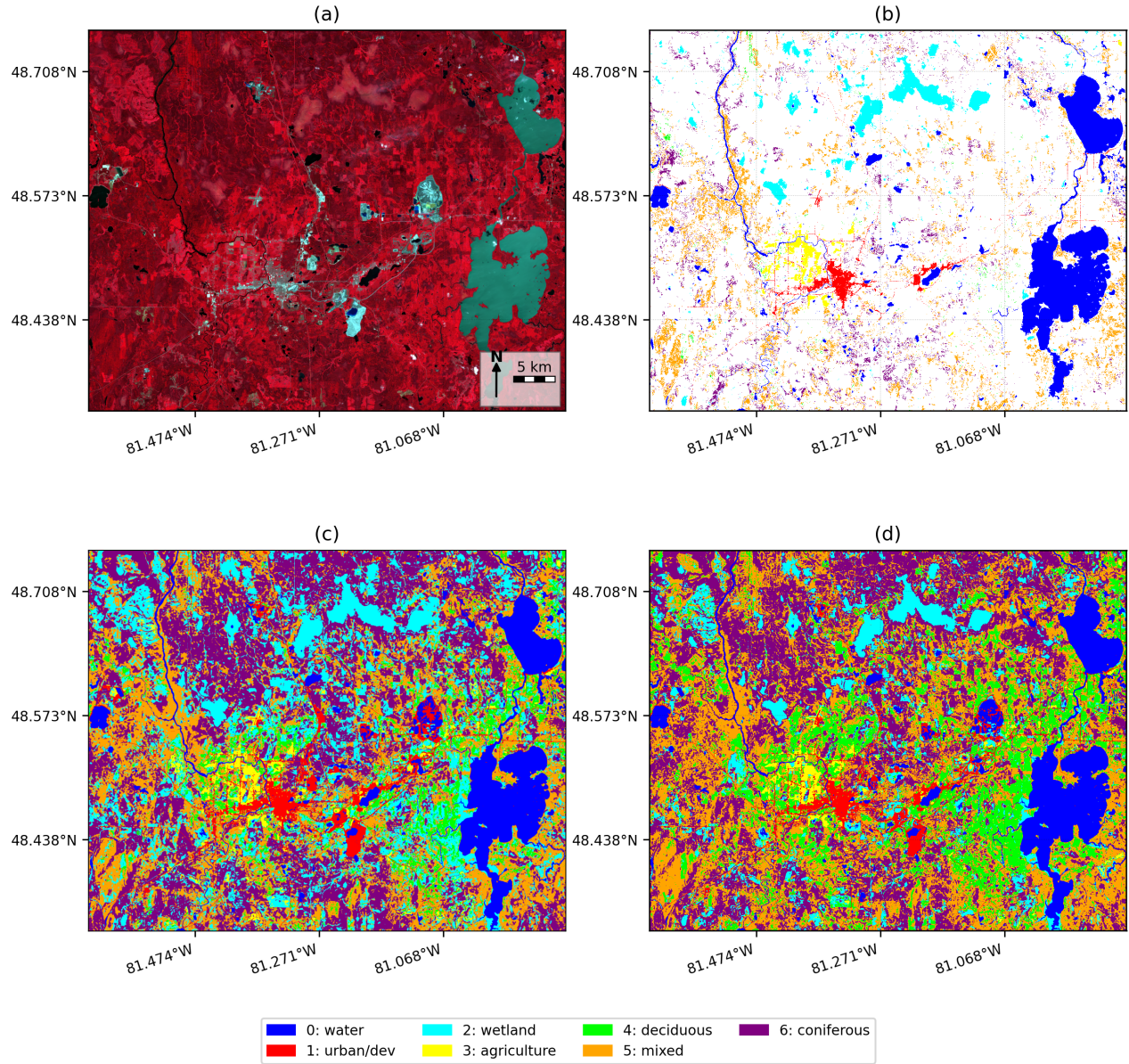


Figure 5.7. Visual comparison across the test area. (a) S2-Aug false-color composite. (b) Ground-truth map. (c) Single-source classification using MHEnc with S2-Oct. (d) Classification using the proposed Hybrid Fusion Strategy.

The classification maps in Figure 5.7 were compared across the test area. Wetland boundaries were sharper in the hybrid result (d) than in the single-source MHEnc with S2-Oct (c). Reduced salt-and-pepper noise was observed in homogeneous regions, particularly within coniferous stands. Class boundaries between Wetland and forest classes were more clearly

defined. Urban areas maintained clearer road networks, and agricultural fields showed reduced fragmentation compared with single-source results.

The proposed Hybrid Fusion Strategy was used to combine temporal information from S2 with structural information from S1. The highest performance among tested configurations was achieved with 0.8665 mIoU, 96.83% OA, and 0.9572  $\kappa$  (Table 5.6). Map-quality improvements were observed. Classification errors remained concentrated at class boundaries and between similar classes.

## Chapter 6 Discussion

### 6.1 On the MHEnc Framework

The MHEnc framework demonstrated improved and, critically, more stable LULC classification performance compared to the baselines. The architectural design (detailed in Section 4.1) was intended to capture multi-scale context, and the learned attention weights provide insight into its operation. Head-3 ( $\delta=3$ ) was assigned the highest weights across all single-source configurations in Table 5.3, ranging from 0.41 to 0.46. Given the current patch size and single downsampling stage, this distribution was interpreted as indicating a consistent utility for broader spatial context, while the contributions from  $\delta = 1$  and  $\delta = 2$  were aligned with fine and intermediate features. The cross-sensor consistency of this weighting pattern was interpreted as reflecting dataset characteristics rather than sensor-specific requirements.

With optical imagery, higher accuracy than both baseline models was achieved by MHEnc under identical training and evaluation conditions (Table 5.1). The improvement was more pronounced with S2-Oct, which was associated with increased class separability and more heterogeneous boundaries due to phenological changes. The multi-scale processing appeared to reduce boundary classification errors by combining broad context with local detail, consistent with the class-specific improvements in Table 5.2.

A primary contribution of the MHEnc framework was its performance stability across datasets with different phenological characteristics. This stability was most evident when comparing S2-Aug and S2-Oct results (Table 5.1), where a pronounced decrease in CBAMUNet performance was observed (from 90.57% OA to 73.43% OA). Class-wise metrics confirmed this instability (Table 5.2). In contrast, MHEnc maintained high accuracy (OA above 90%) and achieved the highest mIoU (0.6507 and 0.7723) across both dates.

This finding suggests that the architectural complexity of MHEnc provides essential robustness. It was hypothesized that the sequential channel-then-spatial attention in CBAM may have over-fitted to date-specific statistics (S2-Aug), limiting its generalization to the leaf-off (S2-Oct) conditions. By contrast, the parallel, logit-level fusion design of MHEnc (detailed in Section 4.1) demonstrated greater resilience to these spectral and textural shifts. This design avoids early commitment to specific features and ensures multi-scale information is consistently leveraged, justifying its complexity through improved model stability.

Furthermore, the utility of the architecture was most pronounced on challenging data. With S1-Aug imagery, MHEnc (0.3782 mIoU) substantially outperformed the BasicUNet (0.2073 mIoU), demonstrating that for low-information sources, such multi-scale processing is critical. The performance gains were also concentrated in challenging classes, such as the 5–7 pp F1-Score improvement for Deciduous Forest over BasicUNet (Table 5.2).

Two limitations were identified. First, the attention weights  $\alpha_i$  were implemented as global scalars, simplifying interpretation but preventing adaptation to local context. Second, the parallel processing incurred increased memory consumption and computation time compared with single-path architectures.

Classification errors in Table 5.6 were concentrated primarily among forest classes and between Agriculture and Wetland, likely due to spectral similarities. Overall, MHEnc achieved higher accuracy than the baselines, maintained performance across acquisition dates, and showed a consistent preference for broader spatial context (Head-3,  $\delta=3$ ) for this dataset.

## 6.2 Design Principles for Multi-Source Data Fusion

Fusion effectiveness was determined by data characteristics. Two patterns were observed across controlled comparisons. For heterogeneous sources (S2 optical and S1 SAR), higher



accuracy than Early Fusion was achieved by Middle Fusion. For homogeneous temporal S2 imagery, higher accuracy than Middle Fusion was achieved by Early Fusion. These patterns were found in both two-source (Table 5.4) and three-source (Table 6.1) evaluations, providing guidance for fusion-stage selection.

The integration of optical and SAR imagery presents challenges owing to differences in physical measurement. Optical channels record reflectance related to material composition, while SAR channels record microwave backscatter ( $\sigma^0$ ), which is determined by geometry, roughness, and dielectric properties (and thus highly sensitive to moisture). These collective differences were found to affect processing in early network layers.

For S1 and S2 combinations, Middle Fusion outperformed Early Fusion (Table 5.4). Limited feature extraction under Early Fusion was indicated by Grad-CAM visualizations, likely because the shared initial filters (a defining feature of Early Fusion, cf. Section 2.5.1) must process statistically distinct reflectance and backscatter values. This effectiveness was likely reduced by statistical differences, numerical imbalance (10 optical vs. 2 SAR channels), and differing textures, despite normalization.

The separate encoders characteristic of Middle Fusion (cf. Section 2.5.2) appeared to mitigate these issues by allowing for modality-specific feature extraction prior to integration. This effect was demonstrated by the Urban/Developed F1-Score improvement: the separate processing preserved both optical-derived road networks and SAR-derived building backscatter before integration.

Different requirements are imposed by temporal imagery from the same sensor. In multi-temporal S2, channels and measurement principles are consistent; useful information resides in temporal changes. For homogeneous temporal inputs, the best results were obtained

with Early Fusion (Table 5.4). This strategy's input-level concatenation (cf. Section 2.5.1) appeared to be optimal for learning temporal gradients between corresponding bands directly. This was consistent with improvements for phenology-sensitive classes like Agriculture and Mixed Forest (Table 5.5).

Conversely, the lower performance of Middle Fusion for temporal imagery was likely related to the architectural separation of encoders (cf. Section 2.5.2), which precludes the modeling of temporal patterns until the feature-level integration stage. Grad-CAM visualizations showed stronger, more focused activations for temporally dynamic classes under Early Fusion.

Performance within each fusion stage was affected by implementation details (Table 5.4), indicating that results depend on the specific integration layers beyond the choice of fusion stage. Feature-level integration (Middle Fusion) also consistently outperformed decision-level (Late Fusion) combination. When branches were trained independently (Late Fusion), internal representations could not be co-adapted based on complementary information.

The following guidelines were suggested by these results. For heterogeneous combinations (e.g., S2 with S1), separate initial processing with feature-level integration (Middle Fusion) is recommended. For homogeneous combinations (e.g., temporal S2), input-level concatenation (Early Fusion) produced the best results by enabling direct learning of temporal gradients.

### **6.3 Hybrid Fusion Design**

The development of the Hybrid Fusion Strategy (cf. Section 4.3, Fig. 4.8) was directly guided by the design principles established in Section 6.2. The architecture operationalizes these principles by hierarchically applying fusion strategies matched to data characteristics: an input-level (Early) fusion for the homogeneous temporal S2 data, followed by a feature-level (Middle)

fusion to integrate the resulting optical stream with the heterogeneous S1 SAR data. This two-date configuration was a deliberate choice to isolate a distinct temporal gradient (peak-growing vs. post-senescence) for assessing fusion strategies.

The effectiveness of this hierarchical approach is demonstrated in Table 6.1. The hybrid strategy achieved the highest overall performance, with an mIoU improvement of +0.0140 over applying Early Fusion to all 22 channels and +0.0325 over a symmetric three-source Middle Fusion. As confirmed by class-wise F1-Scores (Table 6.2), the hybrid strategy particularly benefited challenging classes, with significant gains for Deciduous Forest (+13.76 pp over Late Fusion) and Agriculture (+8.73 pp over the same baseline).

Beyond the choice of fusion stage, performance was affected by the implementation of the feature-level integration module. The hybrid strategy adopted the Convolutional Middle Fusion variant (cf. Section 4.2), which employs  $3 \times 3$  convolutions to model spatial interactions between the fused feature maps. This design achieved higher accuracy than the Symmetric Middle Fusion baseline (Table 6.1), which uses a simpler  $1 \times 1$  convolution post-concatenation. This suggests that the  $3 \times 3$  convolutions effectively capture cross-modal spatial relationships (e.g., high SAR returns adjacent to low optical values) that are missed by a simple  $1 \times 1$  fusion.

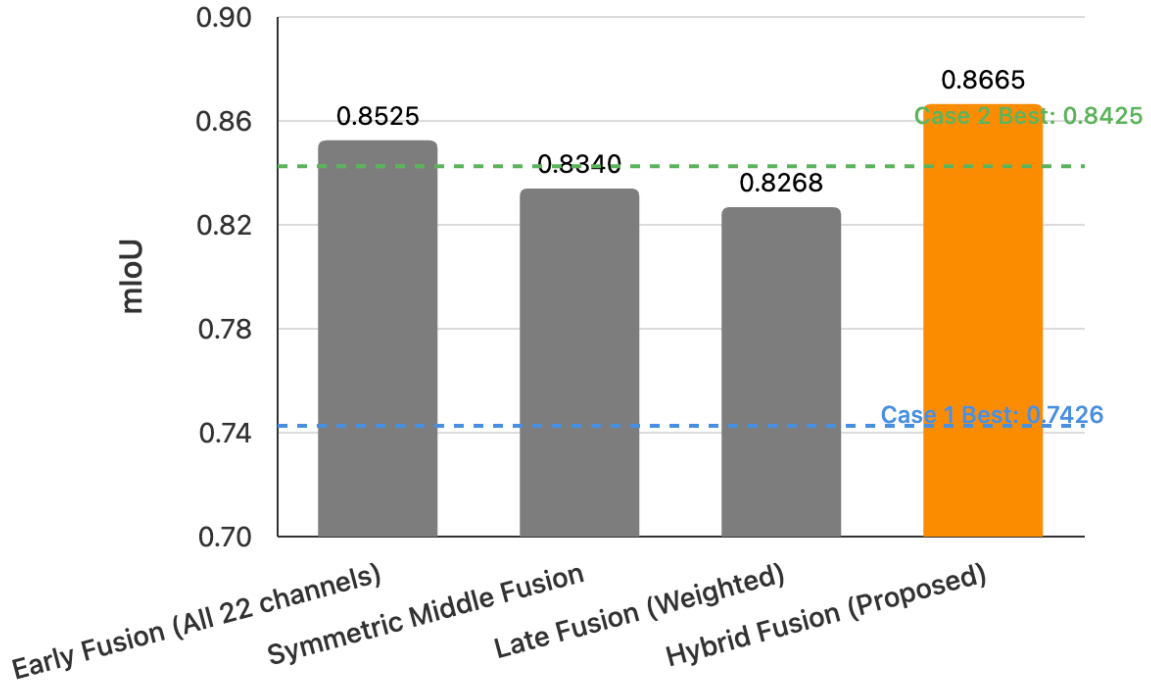


Figure 6.1. Performance comparison of three-source fusion strategies. The proposed Hybrid Fusion strategy (orange) is shown to outperform other configurations and achieves a higher mIoU than the best-performing two-source strategies from Case 1 (blue dashed line) and Case 2 (green dashed line).

Table 6.1. Performance comparison of fusion strategies combining three data sources (S2-Aug + S2-Oct + S1-Aug) on the test area.

| Fusion Strategy                          | OA (%)       | Kappa ( $\kappa$ ) | mIoU          |
|------------------------------------------|--------------|--------------------|---------------|
| <b>Hybrid Fusion Strategy (Proposed)</b> | <b>96.83</b> | <b>0.9572</b>      | <b>0.8665</b> |
| Early Fusion (All Channels)              | 96.46        | 0.9522             | 0.8525        |
| Symmetric Middle Fusion                  | 96.06        | 0.9468             | 0.8340        |
| Late Fusion (Weighted)                   | 96.20        | 0.9486             | 0.8268        |
| Late Fusion (Simple Average)             | 95.39        | 0.9377             | 0.8098        |

Table 6.2. Class-wise F1-Scores (%) for three-source fusion strategies.

| <b>Fusion Strategy</b>       | <b>Water</b> | <b>Urban/Dev</b> | <b>Wetland</b> | <b>Agriculture</b> | <b>Deciduous</b> | <b>Mixed</b> | <b>Coniferous</b> |
|------------------------------|--------------|------------------|----------------|--------------------|------------------|--------------|-------------------|
| Hybrid Fusion Strategy       | 99.50        | 94.78            | 98.23          | 88.48              | 76.09            | 96.45        | 93.11             |
| Early Fusion                 | 99.46        | 94.77            | 97.88          | 86.54              | 73.04            | 96.09        | 92.28             |
| Symmetric Middle Fusion      | 99.65        | 93.33            | 97.17          | 83.13              | 70.63            | 95.58        | 92.02             |
| Late Fusion (Weighted)       | 99.69        | 92.38            | 97.38          | 80.96              | 67.30            | 95.93        | 93.44             |
| Late Fusion (Simple Average) | 99.24        | 93.47            | 97.43          | 79.75              | 62.33            | 95.14        | 90.89             |

Lower performance was observed for Late Fusion strategies (Table 6.1, 6.2). When branches were trained independently and combined only at the prediction stage, adaptation based on complementary information was prevented, and correlated errors remained after averaging. By contrast, the joint-loss training used in feature-level fusion allowed branches to develop complementary representations.

Higher OA and mIoU were achieved when similar sources were processed together first (Early Fusion), followed by feature-level integration of heterogeneous sources (Middle Fusion). This principle could be applied to other sensor combinations, although effectiveness would depend on the specific information content and class distribution.

## 6.4 Edge Effect Investigation

The investigation into patch boundary effects (detailed in Section 4.4.3) revealed that the impact of excluding edge pixels from loss computation was dependent on both architecture and data source (Table 6.3). Detailed metrics for all boundary configurations are provided in Appendices A.1 through A.4.

Table 6.3. Effect of edge-pixel exclusion on model performance (mIoU, %). (Central-Region Only = central  $20 \times 20$ ; Full Patch = full  $25 \times 25$ ; Difference = Central – Full)

| Model                                  | Data Source        | Central-Region Only | Full Patch   | Difference   |
|----------------------------------------|--------------------|---------------------|--------------|--------------|
| BasicUNet                              | S2 Aug             | 60.53               | 62.01        | -1.48        |
| CBAMUNet                               | S2 Aug             | 64.03               | 60.78        | +3.25        |
| MHEnc                                  | S2 Aug             | 65.07               | 59.73        | +5.34        |
| BasicUNet                              | S2 Oct             | 72.32               | 77.70        | -5.38        |
| CBAMUNet                               | S2 Oct             | 60.82               | 65.63        | -4.81        |
| MHEnc                                  | S2 Oct             | 77.23               | 80.20        | -2.97        |
| BasicUNet                              | S1 Aug             | 20.73               | 26.40        | -5.67        |
| CBAMUNet                               | S1 Aug             | 38.16               | 25.16        | +13.00       |
| MHEnc                                  | S1 Aug             | 37.82               | 24.60        | +13.22       |
| <b>Proposed Hybrid Fusion Strategy</b> | <b>All sources</b> | <b>86.65</b>        | <b>82.53</b> | <b>+4.12</b> |

For attention-based architectures with S1-Aug, excluding edge pixels (central  $20 \times 20$  training) significantly improved performance, with absolute mIoU gains of 0.1300 for CBAMUNet and 0.1322 for MHEnc. This improvement was interpreted as being associated with interactions among SAR speckle, boundary padding, and attention mechanisms. When loss was

restricted to central pixels, complete spatial context was provided without the influence of padding artifacts on the attention-weight estimates.

In contrast, BasicUNet with optical imagery showed decreased mIoU when edges were excluded: 0.0148 lower for S2-Aug and 0.0538 lower for S2-Oct. The simpler architecture was interpreted as being less affected by boundary artifacts while benefiting from the additional training pixels (36% of the patch) provided by the full 25x25 area, especially in homogeneous regions.

The Hybrid Fusion Strategy showed intermediate behavior, with mIoU improving by 0.0412 when trained on central pixels (0.8665 versus 0.8253). The improvement in SAR-feature learning was interpreted as outweighing the reduction in optical training pixels.

To isolate training effects from inference, standardized prediction procedures were applied (cf. Section 4.4.3), substantially reducing aggregation differences as a confounding factor. These findings were interpreted as indicating that boundary handling should be matched to architecture and data type. Attention-based models processing SAR benefited from central-region training, whereas BasicUNet processing optical imagery performed better with full patches.

## Chapter 7 Conclusion

An approach to multi-source LULC classification was developed to address two connected challenges: multi-scale feature processing and the integration of heterogeneous remote sensing sources. Through experiments in Northern Ontario, Canada, the MHEnc framework was introduced, and fusion guidelines were established through empirical testing.

Multi-scale processing was implemented through parallel convolutional heads with dilation rates  $\delta = 1, 2$ , and  $3$ . Simultaneous feature extraction at multiple spatial scales was enabled by this design without increasing parameters per kernel. Across single-source configurations, the largest dilation rate was assigned the highest learned attention weights (0.4097-0.4575). With S2-Oct, 92.59% OA was achieved by MHEnc, compared with 89.80% for BasicUNet. The consistent preference for broader spatial context was interpreted as indicating that landscape-level patterns provided discriminative information for LULC classification in this study.

Dependence on data characteristics was indicated by the fusion experiments. For heterogeneous S1-S2, 0.7426 mIoU was achieved by Symmetric Middle Fusion, compared with 0.6669 by Early Fusion. Support for this difference was provided by Grad-CAM visualizations, in which reduced optical dominance was observed when branches were encoded separately. For homogeneous temporal S2, 0.8425 mIoU was reached by Early Fusion, consistent with modeling cross-date contrasts from the first layer. Selection of the fusion stage to match modality similarity was thereby supported by these observations.

Based on these principles, the Hybrid Fusion Strategy was developed. This approach yielded the highest overall performance on the test area (OA = 96.83%, mIoU = 0.8665), as detailed in Section 5.3, demonstrating the effectiveness of the hierarchical design. S2-Aug and



S2-Oct were concatenated at the input to capture phenological differences directly, after which these optical features were merged with S1 through Middle Fusion. Among Middle Fusion variants tested, simple concatenation was outperformed by designs with dedicated cross-modal integration layers.

The impact of patch boundaries was examined by comparing two training configurations: computing loss on the central  $20 \times 20$  pixels versus the full  $25 \times 25$  patch. For S1-Aug with attention-based models, mIoU was improved by 13.00 for CBAMUNet and by 13.22 for MHEnc when loss was restricted to the central region. For S2 imagery with BasicUNet, higher mIoU was produced by full-patch training, with increases of 1.48 on S2-Aug and 5.38 on S2-Oct relative to the central-region configuration. The effects of boundary handling were therefore dependent on both data source and architecture.

Several limitations were recognized in this study, framing the context of the results and guiding future research. First, the reference data, while rigorously constructed through multi-source consensus and validated with independent LiDAR data, were not derived from in-situ field surveys. Consequently, the reported performance metrics represent a high degree of agreement with this robust benchmark rather than absolute ground-truth accuracy. Second, the transferability of the developed models and fusion principles remains to be validated, as the framework was trained and evaluated exclusively on the boreal landscape of Northern Ontario. The specific pre-processing choices, such as the bilinear interpolation used for resampling the Canopy Height Model, also represent a potential source of influence on the final data representation that was not systematically investigated. Furthermore, the temporal scope was intentionally constrained to two dates for Sentinel-2 and a single date for Sentinel-1 to isolate architectural variables. Finally, the proposed MHEnc framework itself has architectural

limitations; its attention weights are global and spatially invariant, and the investigation was confined to CNN-based backbones, leaving the potential of other architectures like Vision Transformers unexplored.

Future work could be directed to address these limitations systematically. To mitigate the reliance on extensive labeled datasets, future research could explore semi-supervised or weakly-supervised learning techniques that can leverage vast amounts of unlabeled satellite data. Correspondingly, the geographic transferability of the MHEnc framework should be rigorously tested by applying it to diverse ecosystems, such as tropical forests or arid regions, to assess its robustness. A comprehensive sensitivity analysis could also be conducted to quantify the impact of different pre-processing methodologies on model performance. To expand the temporal analysis, future efforts should focus on integrating denser Sentinel-2 and Sentinel-1 time-series to model full phenological and structural dynamics throughout the year. Architecturally, the global attention mechanism in MHEnc could be evolved into a spatially adaptive system that adjusts to local contexts, and the fusion principles established here could be tested with more advanced backbones, including hybrid CNN-Transformer models, to potentially capture longer-range spatial dependencies more effectively.

## References

- Agriculture and Agri-Food Canada. (2023). Annual Space-Based Crop Inventory for Canada, 2019. Agroclimate, Geomatics and Earth Observation Division, Science and Technology Branch. <https://open.canada.ca/data/en/dataset/ba2645d5-4458-414d-b196-6303ac06c1c9>
- Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. (2018). DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4), 834-848. <https://doi.org/10.1109/TPAMI.2017.2699184>
- Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2021). An image is worth 16×16 words: Transformers for image recognition at scale. In *Proceedings of ICLR 2021*. <https://arxiv.org/abs/2010.11929>
- European Space Agency (ESA). (2024). SNAP — Sentinel Application Platform (software; v10 release announcement, Apr 18, 2024). <https://step.esa.int/main/snap-download/>
- Ghamisi, P., Rasti, B., Yokoya, N., Wang, Q., Höfle, B., Bruzzone, L., Bovolo, F., Chi, M., Anders, K., Gloaguen, R., Atkinson, P. M., & Benediktsson, J. A. (2019). Multisource and multitemporal data fusion in remote sensing: A comprehensive review of the state of the art. *IEEE Geoscience and Remote Sensing Magazine*, 7(1), 6-39. <https://doi.org/10.1109/MGRS.2018.2890023>

- Ghassemian, H. (2016). A review of remote sensing image fusion methods. *Information Fusion*, 32, 75–89. <https://doi.org/10.1016/j.inffus.2016.03.003>
- Hermosilla, T., Wulder, M. A., White, J. C., & Coops, N. C. (2022). Land cover classification in an era of big and open data: Optimizing localized implementation and training data selection to improve mapping outcomes. *Remote Sensing of Environment*, 268, 112780. <https://doi.org/10.1016/j.rse.2021.112780>
- Hu, B., Xu, Y., Huang, X., Cheng, Q., Ding, Q., Bai, L., & Li, Y. (2021). Improving urban land cover classification with combined use of Sentinel-2 and Sentinel-1 imagery. *ISPRS International Journal of Geo-Information*, 10(8), 533. <https://doi.org/10.3390/ijgi10080533>
- Jin, Y., Guan, X., Ge, Y., Jia, Y., & Li, W. (2022). Improved spatiotemporal information fusion based on Bayesian decision theory for land cover classification. *Remote Sensing*, 14(23), 6003. <https://doi.org/10.3390/rs14236003>
- Khan, M., Hanan, A., Kenzhebay, M., Gazzea, M., & Arghandeh, R. (2024). Transformer-based land use and land cover classification with explainability using satellite imagery. *Scientific Reports*, 14(1), 1-22. <https://doi.org/10.1038/s41598-024-67186-4>
- Li, J., Hong, D., Gao, L., Yao, J., Zheng, K., Zhang, B., & Chanussot, J. (2022). Deep learning in multimodal remote sensing data fusion: A comprehensive review. *International Journal of Applied Earth Observation and Geoinformation*, 112, 102926. <https://doi.org/10.1016/j.jag.2022.102926>

- Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (pp. 2980-2988). IEEE. <https://doi.org/10.1109/ICCV.2017.324>
- Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 3431-3440). IEEE. <https://doi.org/10.1109/CVPR.2015.7298965>
- Loshchilov, I., & Hutter, F. (2019). Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR 2019)*.
- Luo, W., Li, Y., Urtasun, R., & Zemel, R. (2016). Understanding the effective receptive field in deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NeurIPS 2016)*, 29, 4898-4906. <https://arxiv.org/abs/1701.04128>
- Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, 30. [https://proceedings.neurips.cc/paper\\_files/paper/2017/hash/8a20a8621978632d76c43dfd28b67767-Abstract.html](https://proceedings.neurips.cc/paper_files/paper/2017/hash/8a20a8621978632d76c43dfd28b67767-Abstract.html)
- Ma, L., Liu, Y., Zhang, X., Ye, Y., Yin, G., & Johnson, B. A. (2019). Deep learning in remote sensing applications: A meta-analysis and review. *ISPRS Journal of Photogrammetry and Remote Sensing*, 152, 166-177. <https://doi.org/10.1016/j.isprsjprs.2019.04.015>

Mahdianpari, M., Salehi, B., Mohammadimanesh, F., Homayouni, S., & Gill, E. (2019).

The first wetland inventory map of Newfoundland at a spatial resolution of 10 m using Sentinel-1 and Sentinel-2 data on the Google Earth Engine cloud computing platform. *Remote Sensing*, 11(1), 43. <https://doi.org/10.3390/rs11010043>

Mena, F., Arenas, D., Nuske, M., & Dengel, A. (2024). Common practices and taxonomy in

deep multiview fusion for remote sensing applications. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 17, 4797-4818.

<https://doi.org/10.1109/JSTARS.2024.3361556>

Mena, F., Pathak, D., Najjar, H., Sanchez, C., Helber, P., Bischke, B., & Dengel, A. (2025).

Adaptive fusion of multi-modal remote sensing data for optimal sub-field crop yield prediction. *Remote Sensing of Environment*, 318, 114547.

<https://doi.org/10.1016/j.rse.2024.114547>

Minaee, S., Boykov, Y. Y., Porikli, F., Plaza, A. J., Kehtarnavaz, N., & Terzopoulos, D.

(2022). Image segmentation using deep learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(7), 3523-3542.

<https://doi.org/10.1109/TPAMI.2021.3059968>

Milletari, F., Navab, N., & Ahmadi, S.-A. (2016). V-Net: Fully Convolutional Neural

Networks for Volumetric Medical Image Segmentation. *arXiv preprint*

arXiv:1606.04797.

Natural Resources Canada. (2023). 2020 Land Cover of Canada. Canada Centre for Remote Sensing, Natural Resources Canada.

<https://open.canada.ca/data/en/dataset/ee1580ab-a23d-4f86-a09b-79763677eb47>

Ontario Ministry of Natural Resources and Forestry. (2023). Ontario Land Cover Version 1.0. Science and Research Branch, Ontario Ministry of Natural Resources and Forestry.

<https://geohub.lio.gov.on.ca/documents/667367a759214a089917adccdbae7cb2/about>

Ontario Ministry of Natural Resources and Forestry. (2023). Forest Resources Inventory (FRI) leaf-on LiDAR. Ontario GeoHub.

<https://geohub.lio.gov.on.ca/maps/lio%3A%3Aforest-resources-inventory-leaf-on-lidar/about>

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). “Why should I trust you?”: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)* (pp. 1135-1144). ACM. <https://doi.org/10.1145/2939672.2939778>

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2015* (Lecture Notes in Computer Science, Vol. 9351, pp. 234-241). Springer. [https://doi.org/10.1007/978-3-319-24574-4\\_28](https://doi.org/10.1007/978-3-319-24574-4_28)

- Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., & Batra, D. (2017). Grad-CAM: Visual explanations from deep networks via gradient-based localization. In Proceedings of the *IEEE International Conference on Computer Vision (ICCV)* (pp. 618-626). IEEE. <https://doi.org/10.1109/ICCV.2017.74>
- Solórzano, J. V., Mas, J. F., Gao, Y., & Gallardo-Cruz, J. A. (2021). Land use land cover classification with U-Net: Advantages of combining Sentinel-1 and Sentinel-2 imagery. *Remote Sensing*, 13(18), 3600. <https://doi.org/10.3390/rs13183600>
- Steinhausen, M. J., Wagner, P. D., Narasimhan, B., & Waske, B. (2018). Combining Sentinel-1 and Sentinel-2 data for improved land use and land cover mapping of monsoon regions. *International Journal of Applied Earth Observation and Geoinformation*, 73, 595-604. <https://doi.org/10.1016/j.jag.2018.08.011>
- Tavares, P. A., Beltrão, N. E. S., Guimarães, U. S., & Teodoro, A. C. (2019). Integration of Sentinel-1 and Sentinel-2 for classification and LULC mapping in the urban area of Belém, eastern Brazilian Amazon. *Sensors*, 19(5), 1140. <https://doi.org/10.3390/s19051140>
- Vivone, G., Alparone, L., Chanussot, J., Dalla Mura, M., Garzelli, A., Licciardi, G. A., Restaino, R., & Wald, L. (2015). A critical comparison among pansharpening algorithms. *IEEE Transactions on Geoscience and Remote Sensing*, 53(5), 2565–2586. <https://doi.org/10.1109/TGRS.2014.2361734>



- Woo, S., Park, J., Lee, J.-Y., & Kweon, I. S. (2018). CBAM: Convolutional block attention module. In *Computer Vision - ECCV 2018* (Lecture Notes in Computer Science, Vol. 11211, pp. 3-19). Springer. [https://doi.org/10.1007/978-3-030-01234-2\\_1](https://doi.org/10.1007/978-3-030-01234-2_1)
- Yu, F., & Koltun, V. (2016). Multi-scale context aggregation by dilated convolutions. In *International Conference on Learning Representations (ICLR 2016)*. <https://arxiv.org/abs/1511.07122>
- Yuan, X., Shi, J., & Gu, L. (2021). A review of deep learning methods for semantic segmentation of remote sensing imagery. *Expert Systems with Applications*, 169, 114417. <https://doi.org/10.1016/j.eswa.2020.114417>
- Zhang, C., & Li, X. (2022). Land use and land cover mapping in the era of big data. *Land*, 11(10), 1692. <https://doi.org/10.3390/land11101692>
- Zhang, H., & Xu, R. (2018). Exploring the optimal integration levels between SAR and optical data for better urban land cover mapping in the Pearl River Delta. *International Journal of Applied Earth Observation and Geoinformation*, 64, 87-95. <https://doi.org/10.1016/j.jag.2017.08.013>
- Zhao, H., Shi, J., Qi, X., Wang, X., & Jia, J. (2017). Pyramid scene parsing network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 6230-6239). IEEE. <https://doi.org/10.1109/CVPR.2017.660>
- Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., & Torralba, A. (2016). Learning deep features for discriminative localization. In *Proceedings of the IEEE Conference on*

*Computer Vision and Pattern Recognition (CVPR)* (pp. 2921-2929). IEEE.

<https://doi.org/10.1109/CVPR.2016.31>

## Appendix A

A 1. Effect of patch boundary exclusion on baseline model performance.

| Model     | Data Source | Metric             | Central 18×18<br>pixels | Central 20×20<br>pixels | Central 22×22<br>pixels | Full 25×25<br>pixels |
|-----------|-------------|--------------------|-------------------------|-------------------------|-------------------------|----------------------|
| BasicUNet | S2 Aug      | OA (%)             | 70.10                   | 87.72                   | 87.23                   | 89.68                |
|           |             | Kappa ( $\kappa$ ) | 0.6307                  | 0.8349                  | 0.8287                  | 0.8615               |
|           |             | mIoU               | 0.4860                  | 0.6053                  | 0.6099                  | 0.6201               |
| CBAMUNet  | S2 Aug      | OA (%)             | 88.11                   | 90.57                   | 91.26                   | 87.82                |
|           |             | Kappa ( $\kappa$ ) | 0.8407                  | 0.8724                  | 0.8822                  | 0.8387               |
|           |             | mIoU               | 0.6107                  | 0.6403                  | 0.6591                  | 0.6078               |
| MHEnc     | S2 Aug      | OA (%)             | 90.93                   | 90.88                   | 91.81                   | 87.45                |
|           |             | Kappa ( $\kappa$ ) | 0.8780                  | 0.8768                  | 0.8895                  | 0.8323               |
|           |             | mIoU               | 0.6553                  | 0.6507                  | 0.6663                  | 0.5973               |
| BasicUNet | S2 Oct      | OA (%)             | 93.38                   | 89.80                   | 93.66                   | 93.87                |
|           |             | Kappa ( $\kappa$ ) | 0.9105                  | 0.8634                  | 0.9145                  | 0.9172               |
|           |             | mIoU               | 0.7597                  | 0.7232                  | 0.7841                  | 0.7770               |
| CBAMUNet  | S2 Oct      | OA (%)             | 93.44                   | 73.43                   | 94.33                   | 76.08                |
|           |             | Kappa ( $\kappa$ ) | 0.9118                  | 0.6657                  | 0.9234                  | 0.6978               |
|           |             | mIoU               | 0.7681                  | 0.6082                  | 0.7839                  | 0.6563               |
| MHEnc     | S2 Oct      | OA (%)             | 90.07                   | 92.59                   | 94.10                   | 94.55                |
|           |             | Kappa ( $\kappa$ ) | 0.8668                  | 0.9003                  | 0.9205                  | 0.9264               |
|           |             | mIoU               | 0.7234                  | 0.7723                  | 0.7920                  | 0.8020               |
| BasicUNet | S1 Aug      | OA (%)             | 42.06                   | 36.74                   | 37.59                   | 52.14                |
|           |             | Kappa ( $\kappa$ ) | 0.3210                  | 0.2676                  | 0.2793                  | 0.4101               |
|           |             | mIoU               | 0.2623                  | 0.2073                  | 0.2211                  | 0.2640               |
| CBAMUNet  | S1 Aug      | OA (%)             | 38.61                   | 67.88                   | 59.40                   | 37.50                |
|           |             | Kappa ( $\kappa$ ) | 0.3006                  | 0.5854                  | 0.4881                  | 0.2910               |
|           |             | mIoU               | 0.2562                  | 0.3816                  | 0.2898                  | 0.2516               |
| MHEnc     | S1 Aug      | OA (%)             | 75.87                   | 64.88                   | 76.58                   | 40.85                |
|           |             | Kappa ( $\kappa$ ) | 0.6758                  | 0.5518                  | 0.6833                  | 0.3123               |
|           |             | mIoU               | 0.4228                  | 0.3782                  | 0.4271                  | 0.2460               |

**A 2.** Performance metrics for sensor fusion strategies (S2 Aug + S1 Aug) across different patch boundary exclusion configurations.

| <b>Fusion Strategy</b>       | <b>Metric</b>      | <b>Central 18×18<br/>pixels</b> | <b>Central 20×20<br/>pixels</b> | <b>Central 22×22<br/>pixels</b> | <b>Full 25×25<br/>pixels</b> |
|------------------------------|--------------------|---------------------------------|---------------------------------|---------------------------------|------------------------------|
| Early Fusion                 | OA (%)             | 91.22                           | 91.64                           | 92.34                           | 90.61                        |
|                              | Kappa ( $\kappa$ ) | 0.8810                          | 0.8865                          | 0.8965                          | 0.8730                       |
|                              | mIoU               | 0.6488                          | 0.6669                          | 0.6993                          | 0.6433                       |
| Symmetric Middle Fusion      | OA (%)             | 90.67                           | 93.81                           | 93.41                           | 90.50                        |
|                              | Kappa ( $\kappa$ ) | 0.8733                          | 0.9163                          | 0.9108                          | 0.8708                       |
|                              | mIoU               | 0.6484                          | 0.7426                          | 0.7121                          | 0.6529                       |
| Convolutional Middle Fusion  | OA (%)             | 93.40                           | 92.74                           | 92.89                           | 90.25                        |
|                              | Kappa ( $\kappa$ ) | 0.9108                          | 0.9017                          | 0.9038                          | 0.8693                       |
|                              | mIoU               | 0.7230                          | 0.6932                          | 0.6912                          | 0.6465                       |
| Late Fusion (Simple Average) | OA (%)             | 92.18                           | 92.60                           | 92.56                           | 92.07                        |
|                              | Kappa ( $\kappa$ ) | 0.8942                          | 0.8998                          | 0.8993                          | 0.8926                       |
|                              | mIoU               | 0.6732                          | 0.6805                          | 0.6895                          | 0.6651                       |
| Late Fusion (Weighted)       | OA (%)             | 91.77                           | 92.00                           | 89.09                           | 90.76                        |
|                              | Kappa ( $\kappa$ ) | 0.8884                          | 0.8917                          | 0.8521                          | 0.8748                       |
|                              | mIoU               | 0.6580                          | 0.6899                          | 0.5796                          | 0.6377                       |

**A 3.** Performance metrics for temporal fusion strategies (S2 Aug + S2 Oct) across different patch boundary exclusion configurations.

| <b>Fusion Strategy</b>      | <b>Metric</b>      | <b>Central 18×18<br/>pixels</b> | <b>Central 20×20<br/>pixels</b> | <b>Central 22×22<br/>pixels</b> | <b>Full 25×25<br/>pixels</b> |
|-----------------------------|--------------------|---------------------------------|---------------------------------|---------------------------------|------------------------------|
| Early Fusion                | OA (%)             | 96.05                           | 96.44                           | 96.06                           | 96.59                        |
|                             | Kappa ( $\kappa$ ) | 0.9467                          | 0.9518                          | 0.9468                          | 0.9539                       |
|                             | mIoU               | 0.8147                          | 0.8425                          | 0.8262                          | 0.8381                       |
| Symmetric Middle Fusion     | OA (%)             | 94.70                           | 95.29                           | 94.88                           | 93.96                        |
|                             | Kappa ( $\kappa$ ) | 0.9283                          | 0.9361                          | 0.9308                          | 0.9181                       |
|                             | mIoU               | 0.7525                          | 0.8089                          | 0.7698                          | 0.7485                       |
| Convolutional Middle Fusion | OA (%)             | 95.37                           | 95.89                           | 95.62                           | 94.70                        |

|                              |                    |        |        |        |        |
|------------------------------|--------------------|--------|--------|--------|--------|
| Late Fusion (Simple Average) | Kappa ( $\kappa$ ) | 0.9374 | 0.9444 | 0.9407 | 0.9282 |
|                              | mIoU               | 0.7798 | 0.8117 | 0.7934 | 0.7638 |
|                              | OA (%)             | 94.98  | 95.69  | 95.18  | 95.72  |
| Late Fusion (Weighted)       | Kappa ( $\kappa$ ) | 0.9320 | 0.9417 | 0.9348 | 0.9421 |
|                              | mIoU               | 0.7584 | 0.8130 | 0.7804 | 0.8020 |
|                              | OA (%)             | 94.60  | 94.86  | 95.86  | 95.36  |
|                              | Kappa ( $\kappa$ ) | 0.9269 | 0.9304 | 0.9440 | 0.9371 |
|                              | mIoU               | 0.7511 | 0.7671 | 0.8079 | 0.7800 |

**A 4.** Performance metrics for combined sensor and temporal fusion strategies (S2 Aug + S2 Oct + S1 Aug) across different patch boundary exclusion configurations.

| <b>Fusion Strategy</b>       | <b>Metric</b>      | <b>Central 18×18<br/>pixels</b> | <b>Central 20×20<br/>pixels</b> | <b>Central 22×22<br/>pixels</b> | <b>Full 25×25<br/>pixels</b> |
|------------------------------|--------------------|---------------------------------|---------------------------------|---------------------------------|------------------------------|
| Early Fusion                 | OA (%)             | 85.81                           | 96.46                           | 96.27                           | 96.23                        |
|                              | Kappa ( $\kappa$ ) | 0.8175                          | 0.9522                          | 0.9496                          | 0.9491                       |
|                              | mIoU               | 0.7329                          | 0.8525                          | 0.8270                          | 0.8199                       |
| Symmetric Middle Fusion      | OA (%)             | 96.32                           | 96.06                           | 96.23                           | 96.00                        |
|                              | Kappa ( $\kappa$ ) | 0.9502                          | 0.9468                          | 0.9490                          | 0.9460                       |
|                              | mIoU               | 0.8356                          | 0.8340                          | 0.8222                          | 0.8226                       |
| Convolutional Middle Fusion  | OA (%)             | 96.64                           | 96.83                           | 95.95                           | 96.10                        |
|                              | Kappa ( $\kappa$ ) | 0.9546                          | 0.9572                          | 0.9453                          | 0.9473                       |
|                              | mIoU               | 0.8529                          | 0.8665                          | 0.8205                          | 0.8253                       |
| Late Fusion (Simple Average) | OA (%)             | 96.65                           | 95.39                           | 96.51                           | 96.39                        |
|                              | Kappa ( $\kappa$ ) | 0.9547                          | 0.9377                          | 0.9528                          | 0.9512                       |
|                              | mIoU               | 0.8584                          | 0.8098                          | 0.8517                          | 0.8601                       |
| Late Fusion (Weighted)       | OA (%)             | 96.78                           | 96.20                           | 96.01                           | 96.24                        |
|                              | Kappa ( $\kappa$ ) | 0.9564                          | 0.9486                          | 0.9461                          | 0.9491                       |
|                              | mIoU               | 0.8601                          | 0.8268                          | 0.8373                          | 0.8492                       |

**A 5.** Final performance on the validation set for all major experimental configurations.

| <b>Scenario</b>                   | <b>Model / Strategy</b>                      | <b>OA (%)</b> | <b>Kappa (<math>\kappa</math>)</b> | <b>mIoU</b>  |
|-----------------------------------|----------------------------------------------|---------------|------------------------------------|--------------|
| <b>Single-Source<br/>(S2-Aug)</b> | BasicUNet                                    | 89.65         | 0.842                              | 0.799        |
|                                   | CBAMUNet                                     | 89.01         | 0.833                              | 0.786        |
|                                   | <b>MHEnc (Proposed)</b>                      | <b>89.63</b>  | <b>0.842</b>                       | <b>0.812</b> |
| <b>Single-Source<br/>(S2-Oct)</b> | BasicUNet                                    | 85.72         | 0.783                              | 0.755        |
|                                   | CBAMUNet                                     | 86.24         | 0.790                              | 0.779        |
|                                   | <b>MHEnc (Proposed)</b>                      | <b>85.82</b>  | <b>0.783</b>                       | <b>0.750</b> |
| <b>Single-Source<br/>(S1-Aug)</b> | BasicUNet                                    | 52.12         | 0.244                              | 0.321        |
|                                   | CBAMUNet                                     | 53.56         | 0.283                              | 0.367        |
|                                   | <b>MHEnc (Proposed)</b>                      | <b>55.05</b>  | <b>0.276</b>                       | <b>0.383</b> |
| <b>Two-Source<br/>Fusion</b>      | Early Fusion (S2-Aug + S1-Aug)               | 89.62         | 0.841                              | 0.816        |
|                                   | Symmetric Middle Fusion<br>(S2-Aug + S1-Aug) | 90.27         | 0.852                              | 0.815        |
|                                   | Early Fusion (S2-Aug + S2-Oct)               | 90.18         | 0.851                              | 0.828        |
|                                   | Symmetric Middle Fusion<br>(S2-Aug + S2-Oct) | 90.49         | 0.855                              | 0.836        |
| <b>Three-Source<br/>Fusion</b>    | Early Fusion (All Channels)                  | 89.90         | 0.847                              | 0.826        |
|                                   | <b>Hybrid Fusion Strategy (Proposed)</b>     | <b>90.29</b>  | <b>0.852</b>                       | <b>0.831</b> |