



Article

TinyCapsViT: Ultra-Lightweight Hyperspectral and Multispectral Image Classification for UAV Edge Deployment Using Capsule Vision Transformers

Sagar Dalai ^{1,*} , Marco Moreno ¹ , Eoin O'Connell ¹ , Thomas Newe ¹ and Gerard Dooly ²

¹ CRIS Research Group, Department of Electronic and Computer Engineering, University of Limerick, V94 T9PX Limerick, Ireland; marco.moreno@ul.ie (M.M.); eoin.oconnell@ul.ie (E.O.); thomas.newe@ul.ie (T.N.)

² CRIS Research Group, School of Engineering, University of Limerick, The Lonsdale Building, V94 T9PX Limerick, Ireland; gerard.dooly@ul.ie

* Correspondence: sagar.dalai@ul.ie

Highlights

TinyCapsViT introduces an ultra-lightweight capsule-inspired vision transformer for hyperspectral and multispectral image classification. The model requires only 2781 trainable parameters, enabling efficient TinyML and edge deployment. Extensive experiments demonstrate a practical trade-off between classification accuracy, robustness, and computational efficiency. Deployment on Raspberry Pi and Jetson Xavier NX validates its suitability for resource-constrained spectral imaging.

What are the main findings?

- Competitive performance with low complexity: TinyCapsViT maintains competitive classification performance while substantially reducing parameters, memory, and computational requirements.
- Practical edge deployment: TFLite and hardware evaluations demonstrate efficient inference on resource-constrained platforms.

What are the implications?

- Edge environmental monitoring: TinyCapsViT enables spectral image classification on embedded and UAV-based platforms with limited resources.
- Efficient spectral AI: The study demonstrates the potential of lightweight attention and capsule-inspired feature refinement for practical TinyML-based spectral imaging.

Abstract

Hyperspectral image (HSI) classification is central to environmental monitoring, yet real-time deployment of deep learning models on resource-constrained edge platforms remains challenging due to high spectral dimensionality and computational overhead. In this paper, we propose TinyCapsViT, an ultra-lightweight hybrid architecture that integrates convolutional feature extraction, transformer-based attention, and capsule-inspired representation learning for efficient HSI classification under strict TinyML constraints. The model employs a minimal convolutional stem using pointwise and depthwise separable convolutions to capture local spectral-spatial features, followed by a compact tokenization strategy and learnable positional embeddings. A lightweight self-attention module enables global context modeling with reduced computational complexity, while a capsule-inspired refinement block with squash nonlinearity and residual scaling enhances feature discrimination. The proposed architecture contains only 2781 trainable parameters, representing up



Academic Editors: Le Sun, Jie Feng and Yang Xu

Received: 10 June 2026

Revised: 3 August 2026

Accepted: 3 August 2026

Published: 7 August 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and

conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

to a 30× reduction compared with larger architectures such as ResNet and Vision Transformer, and approximately 11× and 9.5× fewer parameters than the CNN and 3D-CNN baselines, respectively. Extensive experiments across benchmark hyperspectral datasets demonstrate that TinyCapsViT achieves up to 99.61% overall accuracy and maintains competitive classification performance despite its substantially reduced model complexity. Although TinyCapsViT does not consistently achieve the highest classification accuracy compared with larger baseline models, it provides a favourable trade-off between classification performance and computational efficiency. These results demonstrate the potential of TinyCapsViT as a practical solution for real-time hyperspectral analysis on resource-constrained edge platforms and UAV-based environmental monitoring applications.

Keywords: hyperspectral image; multispectral image; TinyML; ViT; CapsNet; CNN

1. Introduction

Hyperspectral and multispectral imaging have become essential tools in remote sensing applications such as environmental monitoring, precision agriculture, land cover mapping, mineral exploration, disaster assessment, and UAV-based sensing systems [1,2]. Unlike conventional RGB imaging, hyperspectral imaging captures hundreds of narrow and contiguous spectral bands, while multispectral imaging captures fewer but strategically selected spectral bands, enabling detailed spectral characterization of materials and objects. This spectral richness enables discrimination of land cover classes and environmental changes that are undetectable in conventional imagery, but often this is at the cost of high data dimensionality that strains pipelines [3].

With the rapid development of lightweight sensors and UAV platforms, hyperspectral and multispectral systems are increasingly being deployed for real-time environmental monitoring tasks [4,5]. UAV-based spectral sensing offers high spatial resolution, flexible deployment, reduced operational costs, and improved accessibility in challenging environments such as forests, coastal zones [6], and agricultural fields [1]. These advantages are particularly valuable in ecologically sensitive habitats, such as the saltmarsh environments studied in this work, where timely and accurate decision-making is critical.

However, accurate classification of hyperspectral image (HSI) and multispectral image (MSI) data remains a challenging task due to high spectral dimensionality, strong inter-band correlation, limited labeled samples, and significant computational complexity. Traditional machine learning methods often suffer from the curse of dimensionality and limited generalization capability when dealing with high-dimensional spectral data [7]. Deep learning methods, particularly convolutional neural networks (CNNs), have significantly improved classification performance by learning spatial-spectral representations automatically. Methods such as 2D-CNN, 3D-CNN, and HybridSN have demonstrated strong results in hyperspectral image classification tasks [3,8], but their parameter counts make them impractical for onboard UAV processors and TinyML platforms.

More recently, Vision Transformers (ViTs) have attracted considerable attention due to their ability to model long-range dependencies through self-attention mechanisms. Transformer-based methods such as SpectralFormer and lightweight ViTs have shown promising improvements over CNN-based methods by capturing global contextual information more effectively [9,10]. However, these models often require large parameter counts, high computational cost, and substantial memory resources, which make them difficult to deploy on edge devices and onboard UAV platforms. Capsule networks have also been explored for remote sensing classification due to their ability to preserve hierarchical

relationships between features and improve representation learning. Nevertheless, traditional capsule architectures often introduce expensive routing mechanisms and increased computational burden, limiting their practicality for real-time deployment [11].

Inspired by the effectiveness of the previously proposed CrossCapsViT framework [4], this work investigates how transformer-capsule architectures can be redesigned for ultra-low-resource TinyML deployment without sacrificing classification performance. This paper proposes TinyCapsViT, a lightweight Capsule Vision Transformer for hyperspectral and multispectral image classification under TinyML constraints. The proposed framework combines depthwise separable convolutions for efficient spatial-spectral feature extraction, learnable positional embeddings for spatial awareness, multi-head self-attention for global dependency modeling, and a capsule-inspired feature representation module with squash activation for enhanced discriminative learning. Global average pooling and a lightweight classification head further reduce model complexity and improve deployment feasibility. The complete overview of the proposed TinyCapsViT framework for efficient hyperspectral and multispectral classification in resource constrained edge devices for UAV based environmental monitoring has been depicted in the Figure 1.

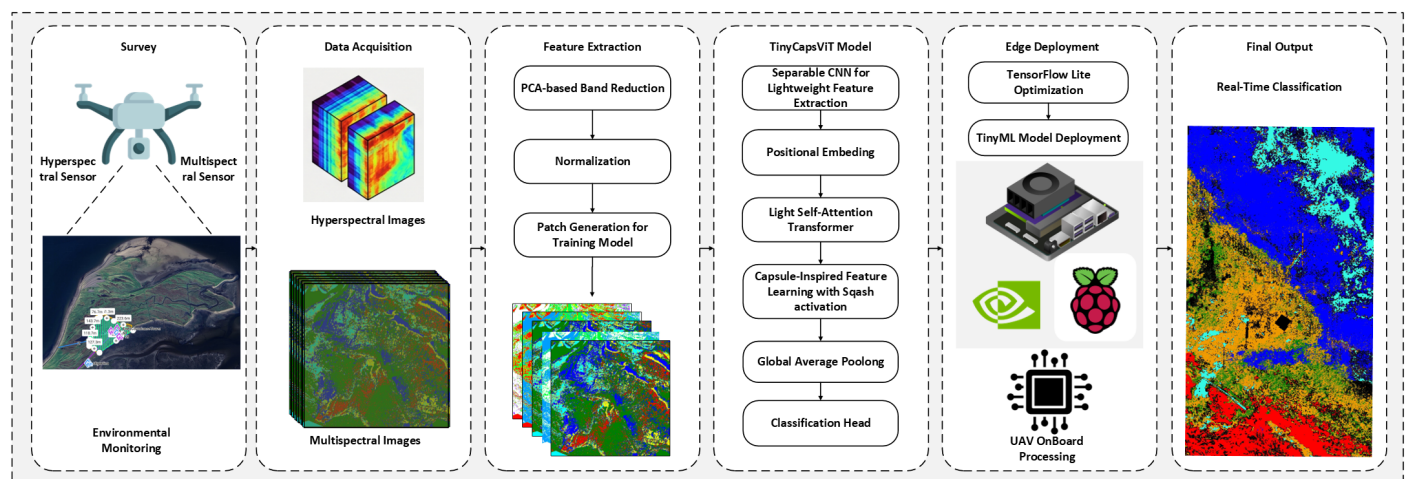


Figure 1. Overview of proposed TinyCapsViT framework for efficient Hyperspectral and Multispectral Classification in resource constrained edge devices for UAV-based Environmental Monitoring.

The main contributions of this paper are summarized as follows:

- We propose TinyCapsViT, an ultra-lightweight vision transformer architecture that integrates efficient convolutional feature extraction, Lite Self-Attention, and a capsule-inspired refinement block for hyperspectral and multispectral image classification.
- The proposed model achieves competitive classification performance using only 2781 trainable parameters, substantially reducing memory and computational requirements for TinyML deployment.
- Extensive experiments on multiple public and custom benchmark datasets demonstrate the effectiveness of TinyCapsViT, including robustness evaluation under spectral corruption.
- Real-world deployment and benchmarking on Raspberry Pi 3 and Jetson Xavier NX validate the practical suitability of the proposed model for resource-constrained edge computing applications.

The remainder of this paper is organized as follows. Section 2 reviews existing studies on HSI/MSI classification, lightweight transformer architectures, and capsule-based learning approaches. Section 3 describes the proposed TinyCapsViT framework in detail. Section 4 presents the experimental setup, benchmark datasets, and comparative

performance analysis. Section 5 discusses the effectiveness, computational efficiency, and deployment suitability of the proposed model. Finally, Section 6 concludes the paper and outlines future research directions.

2. Related Work

This section reviews the progression of classification methods from traditional machine learning through CNNs, transformers, and capsule networks, identifying the computational and architectural gaps that motivate TinyCapsViT.

2.1. Traditional Machine Learning for Spectral Image Classification

Hyperspectral image (HSI) classification has been widely studied due to its importance in remote sensing and environmental monitoring. Early approaches primarily relied on traditional machine learning techniques such as support vector machines (SVMs) and random forests, which operate on handcrafted spectral features [12,13]. Although these methods perform reasonably well with limited training data, they struggle to capture complex spectral–spatial correlations inherent in high-dimensional hyperspectral data.

2.2. Deep Learning for Spectral Image Classification

2.2.1. CNN-Based Methods

Convolutional Neural Networks (CNNs) have been extensively used for hyperspectral and multispectral image classification due to their strong capability in automatically learning spatial–spectral features. Early studies primarily employed 2D-CNNs, where spectral bands are treated as channel-wise inputs and spatial features are extracted through convolutional operations. Although effective, 2D-CNNs often fail to fully exploit the spectral continuity and inter-band correlations inherent in hyperspectral data [3].

To overcome this limitation, 3D-CNNs were introduced to jointly model spatial and spectral information by applying convolutions across both dimensions simultaneously. Models such as HybridSN and Spectral–Spatial Residual Network (SSRN) significantly improved classification performance by combining 3D spectral feature extraction with 2D spatial refinement [14,15]. Similarly, residual learning strategies such as ResNet-based architectures improved deeper feature learning by mitigating gradient degradation problems in deep CNNs [16]. However, these models usually involve large parameter counts, high memory consumption, and increased computational complexity, limiting their applicability in resource-constrained edge systems.

2.2.2. Transformer-Based Methods

Transformer-based models have recently gained significant attention for hyperspectral image classification due to their ability to model long-range dependencies using self-attention mechanisms. Unlike CNNs, Vision Transformers (ViTs) can capture global contextual relationships more effectively, which is particularly beneficial for hyperspectral data where spectral dependencies span across multiple bands [17].

SpectralFormer introduced transformer-based modeling specifically for hyperspectral image classification and demonstrated notable improvements over conventional CNN-based methods by leveraging spectral sequence modeling through self-attention [18]. Several lightweight transformer variants and hybrid CNN–transformer frameworks have also been proposed to balance accuracy and computational efficiency [8]. However, standard transformer architectures often require large embedding dimensions, multiple attention heads, and high computational resources, resulting in significant memory usage and slower inference. These limitations make them difficult to deploy on edge devices and onboard UAV platforms where real-time processing is essential.

2.2.3. Capsule Networks

In remote sensing and hyperspectral image (HSI) analysis, capsule-based architectures have attracted increasing attention because hyperspectral data contain highly correlated spectral–spatial information and complex nonlinear feature distributions. Traditional CNN-based methods often struggle to preserve detailed spatial relationships and may fail to effectively model subtle spectral variations among neighbouring classes. Capsule networks address this limitation by maintaining richer feature representations and capturing intrinsic spectral–spatial dependencies [19]. In particular, CapsNet-based frameworks have demonstrated improved classification performance in challenging scenarios involving mixed pixels, limited training samples, and highly heterogeneous land-cover categories.

Several studies have extended the original CapsNet architecture to hyperspectral image classification. Jia et al. proposed a spectral–spatial capsule network that combines 1D and 2D convolutions with capsule layers to better exploit joint spectral and spatial features, showing superior performance compared with conventional CNN-based methods [19]. Similarly, attention-guided capsule architectures have been introduced to further enhance feature selection and improve discriminative representation learning for remote sensing imagery [20]. These models integrate attention mechanisms with capsule routing to focus on the most informative spectral–spatial regions while preserving hierarchical relationships between features.

More recent approaches have explored multiscale and 3D capsule architectures for hyperspectral classification. Wang et al. proposed a multiscale spectral–spatial capsule network that integrates multi-head attention with capsule routing to extract finer-grained spectral–spatial information across multiple feature scales [21]. Likewise, 3D convolutional capsule networks have been developed to jointly learn spectral and spatial representations while improving generalization under limited supervision [22]. These methods demonstrate the potential of capsule-based learning for enhancing representation quality and classification robustness in remote sensing applications.

Despite these advantages, conventional CapsNet architectures suffer from significant computational limitations. The dynamic routing process between capsules is iterative and computationally expensive, leading to increased inference time, memory consumption, and parameter complexity [23]. Such overhead becomes particularly problematic for large-scale hyperspectral datasets and real-time applications deployed on edge devices or UAV platforms. To mitigate these issues, recent studies have proposed lightweight routing strategies, attention-guided capsule mechanisms, and partially connected capsule structures aimed at reducing computational burden while preserving classification accuracy [24,25]. Nevertheless, achieving an optimal balance between efficiency and representational capability remains an open research challenge in capsule-based remote sensing models.

2.3. TinyML and Edge AI for Spectral Image Classification

The growing demand for real-time environmental monitoring, precision agriculture, disaster management, and autonomous UAV systems has accelerated research in TinyML and edge artificial intelligence (Edge AI) for remote sensing applications. TinyML enables machine learning models to run on resource-constrained embedded devices such as microcontrollers, Raspberry Pi systems, NVIDIA Jetson platforms, and onboard UAV processors while maintaining low latency, low power consumption, and reduced memory usage [26]. Unlike cloud-based processing, edge AI performs data analysis directly on sensing devices, reducing communication overhead and enabling faster real-time decision-making.

In hyperspectral and multispectral remote sensing, edge intelligence is particularly important because hyperspectral sensors generate high-dimensional data with hundreds

of spectral bands, leading to high storage, transmission, and computational requirements. Traditional cloud-based processing is often unsuitable for UAV and satellite applications due to bandwidth limitations, latency, and energy consumption. Consequently, lightweight onboard AI models have emerged as a promising solution for real-time spectral analysis and autonomous operation [27].

To support deployment on constrained hardware, several optimization techniques are commonly used in TinyML systems. TensorFlow Lite (TFLite), quantization, pruning, and knowledge distillation help reduce model size, memory usage, and computational complexity while maintaining acceptable accuracy [28,29]. Lightweight neural architectures such as MobileNet-style networks and efficient attention modules are also widely adopted to minimize floating-point operations (FLOPs) and energy consumption.

Recent studies have explored the integration of transformers and capsule networks to leverage both global contextual modeling and discriminative feature representation. For instance, CrossCapsViT [4] combined cross-attention mechanisms with capsule-inspired learning to improve hyperspectral image classification performance by jointly exploiting spectral-spatial dependencies and hierarchical feature representations. While such hybrid architectures achieve strong classification accuracy, their computational complexity and parameter requirements can limit deployment on resource-constrained edge devices.

Despite recent progress, many hyperspectral image classification models primarily focus on improving accuracy while overlooking deployment constraints. Deep CNN and transformer-based architectures typically require substantial computational power and memory resources, limiting their applicability for real-time edge deployment on UAVs and TinyML platforms [10,11]. Consequently, there is an increasing demand for compact and computationally efficient models that simultaneously achieve high classification performance, low parameter complexity, fast inference, and edge-device compatibility.

2.4. Research Gap

Although CNNs, transformers, and capsule networks have individually demonstrated strong performance in hyperspectral and multispectral image classification, existing methods often suffer from one of two major limitations: either high computational complexity or insufficient global contextual modeling. CNN-based methods struggle to capture long-range dependencies, transformer models are often too computationally expensive for edge deployment, and capsule networks introduce costly routing operations.

Very few studies have focused on designing a unified lightweight architecture that combines the efficiency of separable convolutions, the global modeling capability of transformers, and the discriminative power of capsule learning under strict TinyML constraints.

To address these limitations, this paper proposes TinyCapsViT, a lightweight Capsule Vision Transformer that combines separable convolutions, efficient self-attention, and capsule-inspired feature refinement within a compact architecture tailored for hyperspectral and multispectral image classification. The proposed framework is specifically designed to enable accurate and real-time deployment on resource-constrained edge and TinyML platforms.

3. Proposed Methodology

In this section, we present TinyCapsViT, an ultra-lightweight hybrid architecture designed for efficient hyperspectral image (HSI) or multispectral image classification. The proposed framework combines convolutional feature extraction, lightweight self-attention, and capsule-inspired representation learning to effectively model both local spectral-spatial patterns and global contextual dependencies. The architecture is specifically designed to

minimize computational complexity while maintaining high classification accuracy, making it suitable for TinyML and edge deployment scenarios.

The overall pipeline consists of five major stages: spatial-spectral patch extraction, dimensionality reduction using Principal Component Analysis (PCA), data normalization, lightweight feature learning using the proposed TinyCapsViT architecture, and TensorFlow Lite (TFLite) deployment for real-time inference. Figure 2 illustrates the complete framework of the proposed method.

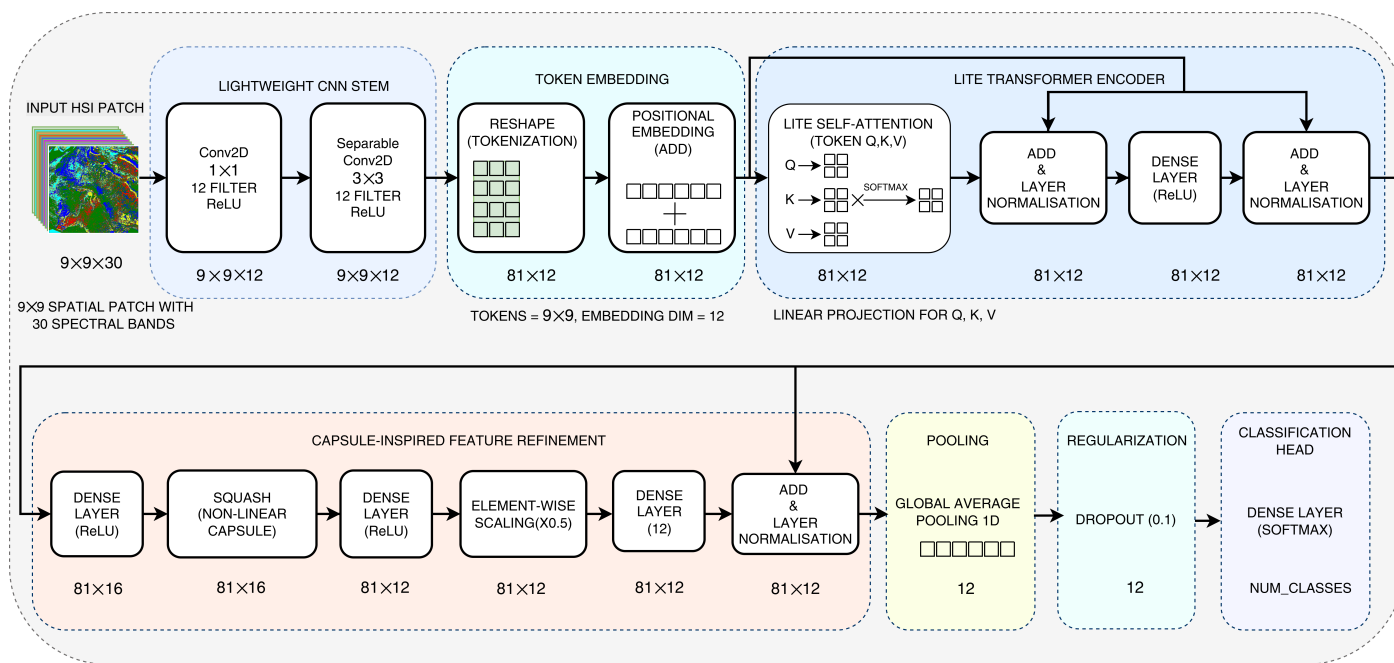


Figure 2. Architecture of proposed TinyCapsViT model integrating lightweight separable convolution, transformer attention and capsule-inspired feature representation for efficient hyperspectral/multispectral image classification.

3.1. Data Preprocessing

Hyperspectral and multispectral images contain rich spectral information distributed across multiple bands. To effectively utilize both spatial and spectral context, a patch-based learning strategy combined with dimensionality reduction is adopted.

Let the input data cube be represented as:

$$\mathbf{X} \in \mathbb{R}^{H \times W \times B} \quad (1)$$

where H and W denote the spatial dimensions, and B represents the number of spectral bands.

For each labeled pixel location (i, j) , a fixed-size spatial-spectral patch centered at the target pixel is extracted:

$$\mathbf{P} \in \mathbb{R}^{S \times S \times B} \quad (2)$$

where S denotes the patch size. In this work, a patch size of 9×9 is used, which provides a good balance between capturing local contextual information and maintaining computational efficiency.

Due to the high dimensionality and strong inter-band correlation in hyperspectral data, Principal Component Analysis (PCA) was employed as a dimensionality reduction technique to reduce the computational burden associated with high-dimensional hyperspectral data. Rather than selecting individual spectral bands, PCA projects the original spectral information into a lower-dimensional subspace while preserving the majority of

data variance. This preprocessing step has been widely adopted in hyperspectral image classification because it significantly reduces memory requirements and inference cost, making it particularly suitable for resource-constrained TinyML deployments. Although dedicated band-selection methods preserve physically interpretable spectral bands, they typically require additional optimization procedures and are often dataset-specific. Since the primary objective of this work is the development of an efficient deployment-oriented classification architecture rather than optimal band selection, PCA provides a computationally efficient and widely accepted preprocessing strategy. The original spectral space is projected into a lower-dimensional subspace as:

$$X_{PCA} = \mathbf{X}\mathbf{W} \quad (3)$$

where $\mathbf{W}$ represents the projection matrix formed by the principal eigenvectors.

This transformation preserves the most informative spectral variance while significantly reducing computational complexity, memory usage, and model training time, making it suitable for resource-constrained TinyML deployment.

3.2. Data Normalization

After PCA, feature normalization is performed using standardization to stabilize training and improve convergence.

Each feature is normalized as:

$$X_{norm} = \frac{X - \mu}{\sigma} \quad (4)$$

where μ and σ represent the mean and standard deviation of each spectral feature, respectively.

This step ensures that all spectral features contribute equally during optimization.

3.3. TinyCapsViT Architecture

The proposed TinyCapsViT model combines lightweight convolutional feature extraction, efficient transformer-based global context modeling, and capsule-inspired discriminative representation learning within a compact architecture optimized for TinyML and edge deployment.

3.3.1. Lightweight CNN Stem

Let the input hyperspectral patch as $\mathbf{X} \in \mathbb{R}^{H \times W \times B}$. Due to the high spectral redundancy in HSI data, directly processing all bands is computationally expensive. To address this, we first apply a pointwise convolution:

$$\mathbf{F}_1 = \sigma(\text{Conv}_{1 \times 1}(\mathbf{X})) \quad (5)$$

which performs channel-wise projection and reduces spectral redundancy while preserving essential information. Next, a depthwise separable convolution is employed:

$$\mathbf{F}_2 = \sigma(\text{SepConv}_{3 \times 3}(\mathbf{F}_1)) \quad (6)$$

to efficiently capture local spatial context with significantly fewer parameters compared to standard convolutions. This lightweight CNN stem forms an efficient spectral-spatial feature extractor.

3.3.2. Tokenization and Positional Encoding

The feature map $\mathbf{F}_2 \in \mathbb{R}^{H \times W \times C}$ is reshaped into a sequence of tokens:

$$\mathbf{Z} \in \mathbb{R}^{N \times C}, \quad N = H \times W \quad (7)$$

where each token corresponds to a spatial location with embedded spectral features.

To preserve spatial relationships, a learnable positional embedding $\mathbf{P} \in \mathbb{R}^{N \times C}$ is added:

$$\mathbf{Z}_0 = \mathbf{Z} + \mathbf{P} \quad (8)$$

Unlike fixed sinusoidal encodings, learnable embeddings adapt to dataset-specific spatial structures.

3.3.3. Lite Self-Attention Module

To efficiently capture global contextual dependencies in hyperspectral data, a lightweight self-attention mechanism is employed. Unlike conventional multi-head attention, which introduces significant computational overhead, the proposed module adopts a simplified single-attention design to reduce parameter complexity while preserving global feature interaction.

Given the token representation $\mathbf{Z}_0 \in \mathbb{R}^{N \times d}$, where N denotes the number of tokens and d represents the embedding dimension, linear projections are first applied to generate the query, key, and value matrices:

$$\mathbf{Q} = \mathbf{Z}_0 \mathbf{W}_Q, \quad \mathbf{K} = \mathbf{Z}_0 \mathbf{W}_K, \quad \mathbf{V} = \mathbf{Z}_0 \mathbf{W}_V \quad (9)$$

where $\mathbf{W}_Q$, $\mathbf{W}_K$, and $\mathbf{W}_V$ are trainable projection matrices.

The attention map is computed using scaled dot-product attention:

$$\mathbf{A} = \text{Softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}}\right) \quad (10)$$

where the scaling factor $\sqrt{d}$ stabilizes gradient propagation and prevents excessively large attention scores.

The refined feature representation is obtained by:

$$\mathbf{Z}_1 = \mathbf{A}\mathbf{V} \quad (11)$$

To improve optimization stability and feature preservation, a residual connection followed by layer normalization is applied:

$$\mathbf{Z}_2 = \text{LayerNorm}(\mathbf{Z}_0 + \mathbf{Z}_1) \quad (12)$$

The proposed Lite Self-Attention module enables effective modeling of long-range spectral-spatial relationships while maintaining low computational and memory requirements. This makes the architecture particularly suitable for TinyML and resource-constrained edge deployment scenarios.

3.3.4. Lightweight Feed-Forward Network

A compact feed-forward network (FFN) is used for feature transformation:

$$\mathbf{Z}_3 = \text{LayerNorm}(\mathbf{Z}_2 + \sigma(\mathbf{Z}_2 \mathbf{W}_F)) \quad (13)$$

This single-layer FFN reduces parameter count while maintaining sufficient representational capacity.

3.3.5. Capsule-Inspired Feature Refinement

To improve discriminative representation learning, a capsule-inspired refinement block is introduced after the lightweight feed-forward stage. Unlike conventional dense transformations, the proposed module preserves feature orientation information while suppressing insignificant activations, enabling improved class separability with minimal computational overhead.

Given the intermediate feature representation $\mathbf{Z}_3 \in \mathbb{R}^{N \times d}$, the features are first projected into a higher-dimensional latent space $\mathbf{U}$.

$$\mathbf{U} = \sigma(\mathbf{Z}_3 \mathbf{W}_1) \quad (14)$$

where $\mathbf{W}_1$ denotes the trainable projection matrix, and $\sigma(\cdot)$ represents the ReLU activation function.

To normalize feature vectors while preserving directional information, a squash non-linearity is applied.

$$\text{squash}(\mathbf{u}) = \frac{\|\mathbf{u}\|^2}{1 + \|\mathbf{u}\|^2} \cdot \frac{\mathbf{u}}{\|\mathbf{u}\|} \quad (15)$$

The squash function constrains vector magnitudes to the range $(0, 1)$, allowing strongly activated features to retain higher importance while suppressing weaker responses. This improves representation robustness and enhances inter-class discrimination. The refined capsule representation is then projected back to the embedding dimension $\mathbf{V}$.

$$\mathbf{V} = \text{squash}(\mathbf{U}) \mathbf{W}_2 \quad (16)$$

where $\mathbf{W}_2$ denotes the reconstruction projection matrix.

Finally, residual scaling and layer normalization are applied to stabilize feature learning and improve optimization. The residual scaling factor α is empirically set to 0.5 to regulate feature propagation and prevent over-amplification of capsule responses.

$$\mathbf{Z}_4 = \text{LayerNorm}(\mathbf{Z}_3 + \alpha \mathbf{V}) \quad (17)$$

Compared to traditional capsule networks that rely on computationally expensive dynamic routing, the proposed capsule-inspired refinement block provides lightweight discriminative feature enhancement with significantly lower parameter and computational complexity, making it suitable for TinyML and edge-oriented hyperspectral image classification.

3.3.6. Classification Head

After feature refinement, global feature aggregation is performed using Global Average Pooling (GAP) to obtain a compact representation of the input hyperspectral patch:

$$\mathbf{z} = \text{GAP}(\mathbf{Z}_4) \quad (18)$$

where $\mathbf{z} \in \mathbb{R}^d$ represents the aggregated feature vector, and d denotes the embedding dimension.

The pooled representation is then passed through a dropout layer to reduce overfitting and improve generalization capability during training. Finally, a fully con-

nected classification layer followed by the Softmax activation function produces the class probability distribution:

$$\hat{\mathbf{y}} = \text{Softmax}(\mathbf{z}\mathbf{W}_c) \quad (19)$$

where $\mathbf{W}_c$ denotes the trainable classification weight matrix, and $\hat{\mathbf{y}}$ represents the predicted probability vector over all classes.

The use of global average pooling significantly reduces the number of trainable parameters compared to conventional fully connected feature flattening, thereby improving computational efficiency and making the proposed architecture more suitable for TinyML and edge deployment scenarios.

3.4. Computational Efficiency

The proposed TinyCapsViT model contains approximately 2.8K parameters, making it significantly more efficient than conventional CNN and transformer-based models. The use of depthwise separable convolutions, lightweight attention, and compact capsule refinement ensures minimal memory footprint and computational cost, enabling deployment on edge devices and TinyML platforms without sacrificing performance.

4. Experimental Results

This section explores the experimental results of the proposed TinyCapsViT model for hyperspectral image classification. The model is evaluated in terms of classification performance, computational efficiency, and deployment feasibility under TinyML constraints. Comparative analysis is conducted against widely used baseline models including CNN, 3D-CNN, ResNet, and Vision Transformer (ViT) using identical preprocessing, training, and testing settings.

4.1. Experimental Setup

Experiments were conducted on five publicly available hyperspectral benchmark datasets Table 1, namely Pavia University (PaviaU), Salinas Valley, Houston University 2013 (Houston), Kennedy Space Center (KSC), and Indian Pines, together with a custom UAV-acquired hyperspectral dataset collected at the Derrymore Saltmarsh site. The custom dataset was acquired using a Resonon Pika-L hyperspectral sensor [30] mounted on a DJI Matrice 300 (M300) UAV platform [31]. Further details regarding the hardware configuration, data acquisition, preprocessing, and ground-truth generation of the Derrymore dataset are provided in our previous work [4].

Table 1. Characteristics of the benchmark hyperspectral image datasets used for evaluating the proposed TinyCapsViT framework, including sensor type, number of labeled samples, classes, spectral bands, spatial resolution, and acquisition year.

Dataset	Sensor	Samples	Classes	Bands	Spatial Resolution	Year
Pavia University (PU) [32]	ROSIS	42,776	9	103	1.3 m	2001
Salinas Valley [32]	AVIRIS	54,129	16	204	3.7 m	2000
Houston (HU2013) [33]	CASI	15,029	15	144	2.5 m	2013
Kennedy Space Center (KSC) [34]	AVIRIS	5211	13	176	18 m	1996
Indian Pines [35]	AVIRIS	10,249	16	224	20 m	1992
Derrymore Saltmarsh [4]	Pika L	7,250,058	4	283	0.013 m	2024

A patch-based learning strategy with a spatial patch size of 9×9 was adopted consistently across all evaluated models. Principal Component Analysis (PCA) was employed for spectral dimensionality reduction, followed by feature standardization.

To ensure a fair comparison, all models were evaluated using the same experimental protocol. A total of 100 training samples per class were randomly selected for model training, while identical train–validation–test splits, preprocessing procedures, batch size, optimizer settings, and evaluation metrics were used across all models. The models were trained using the Adam optimizer with categorical cross-entropy loss, with early stopping employed to mitigate overfitting. For deployment evaluation, the trained models were converted to TensorFlow Lite (TFLite), and post-training quantization was applied prior to benchmarking on the target edge platforms.

4.2. Derrymore Saltmarsh Dataset

The Derrymore Saltmarsh dataset was acquired using a UAV-based hyperspectral imaging platform comprising a DJI Matrice 300 UAV and a Pika L hyperspectral sensor as in Figure 3. The dataset contains four annotated land-cover classes and was collected over the Derrymore Saltmarsh site in County Kerry, Ireland. Comprehensive details of the hardware configuration, flight campaign, data preprocessing, and ground-truth generation can be found in our previous CrossCapsViT study [4]. The same dataset is used in this work to assess the classification performance and deployment efficiency of TinyCapsViT.



Figure 3. Experimental setup for hyperspectral data acquisition, including the DJI M300 UAV equipped with a Resonon hyperspectral imager, Emlid GNSS receiver, sampling quadrat, and calibration tarp.

4.3. Evaluation Metrics

The performance of all models was evaluated using standard classification metrics including Overall Accuracy (OA), Cohen’s Kappa coefficient, model parameter count, model size, and inference latency.

Overall Accuracy is computed as:

$$OA = \frac{N_{correct}}{N_{total}} \times 100 \quad (20)$$

where $N_{correct}$ represents the number of correctly classified samples, and N_{total} denotes the total number of test samples.

The Kappa coefficient is used to measure classification agreement beyond random chance and is defined as:

$$\kappa = \frac{p_o - p_e}{1 - p_e} \quad (21)$$

where p_o is the observed agreement, and p_e is the expected agreement.

To evaluate the computational efficiency of the proposed model for TinyML and edge deployment, inference latency (Latency) is measured as the average time required to process a single input sample.

$$\text{Latency} = \frac{T_{\text{total}}}{N} \quad (22)$$

where T_{total} denotes the total inference time for processing N samples. The latency is reported in milliseconds (ms/sample) and measured under identical hardware and software conditions to ensure fair comparison across models.

In addition to latency, model complexity is evaluated using the total number of trainable parameters P_{total} .

$$P_{\text{total}} = \sum_{l=1}^L P_l \quad (23)$$

where P_l represents the number of parameters in the l -th layer, and L denotes the total number of layers.

These metrics collectively provide a comprehensive evaluation of classification performance, computational efficiency, and deployment suitability of the proposed TinyCapsViT framework.

4.4. Training Convergence Analysis

Figure 4 presents the training and validation convergence behavior of CNN, 3D-CNN, ResNet, ViT, SSFTTNet, BioLiteNet, and the proposed TinyCapsViT on the Pavia University dataset. The corresponding accuracy and loss curves provide insight into the optimization stability and generalization behavior of the different architectures. Overall, all models show successful optimization, although noticeable differences can be observed in the gap between training and validation performance.

The CNN model in Figure 4A exhibits rapid convergence, reaching nearly 100% training accuracy within the early training epochs. The validation accuracy stabilizes at approximately 90–91%, while the training and validation losses converge to approximately 0.24 and 0.45, respectively. The persistent gap between training and validation performance indicates moderate overfitting, although the validation behavior remains relatively stable.

The 3D-CNN model shown in Figure 4B demonstrates a more gradual convergence pattern. Its training accuracy progressively approaches 98–99%, whereas the validation accuracy remains around 90%. Similarly, the training loss continuously decreases while the validation loss stabilizes at a higher level with moderate fluctuations. This behavior indicates a degree of overfitting despite relatively stable classification performance.

As illustrated in Figure 4C, ResNet converges rapidly and achieves nearly 100% training accuracy, with validation accuracy stabilizing at approximately 90–91%. The training and validation losses converge to approximately 0.24 and 0.48, respectively. Compared with the other conventional CNN-based architectures, ResNet exhibits relatively smooth and consistent convergence.

The ViT model in Figure 4D also reaches nearly 100% training accuracy; however, its validation accuracy remains around 88–89%. A progressively increasing separation between the training and validation losses is observed during training, with final values of approximately 0.24 and 0.55, respectively. This indicates a stronger tendency toward

overfitting, suggesting that the transformer architecture is more sensitive to the available training samples.

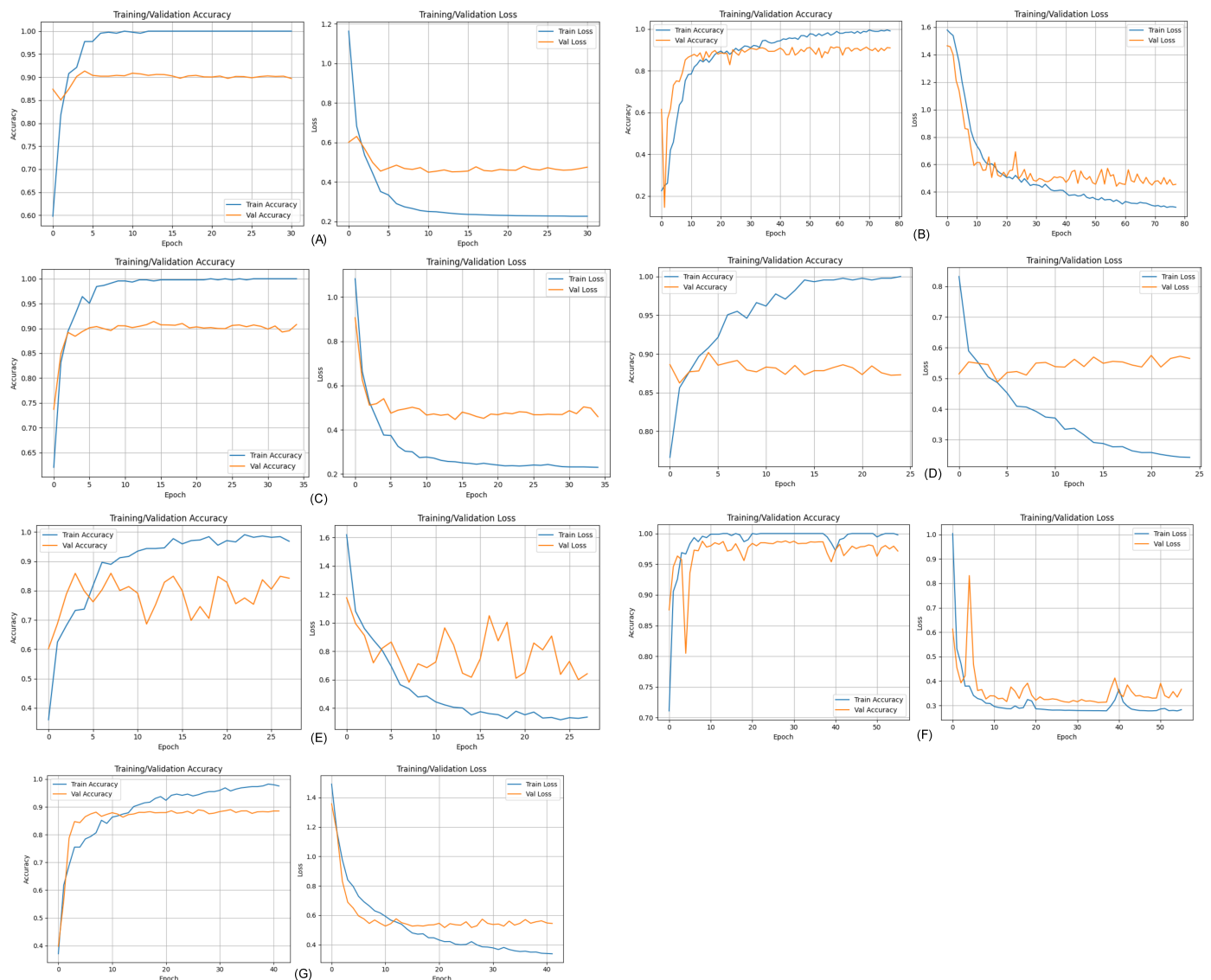


Figure 4. Training and validation accuracy and loss curves of the evaluated models on the Pavia University dataset: (A) CNN, (B) 3D-CNN, (C) ResNet, (D) ViT, (E) BioLiteNet, (F) SSFTTNet, and (G) TinyCapsViT. The curves illustrate the convergence and generalization behavior of each architecture during training.

BioLiteNet [36], presented in Figure 4E, achieves approximately 98–99% training accuracy, while its validation accuracy fluctuates between approximately 83% and 85% toward the later training epochs. The corresponding training loss decreases to approximately 0.34, whereas the validation loss remains considerably higher and exhibits noticeable fluctuations. These results indicate a larger training–validation gap and reduced generalization stability.

In contrast, SSFTTNet in Figure 4F demonstrates strong convergence behavior. The model reaches nearly 100% training accuracy while maintaining validation accuracy of approximately 97–98%. Its training and validation losses stabilize at approximately 0.28 and 0.34, respectively, with only occasional fluctuations in the validation curves. This indicates strong optimization behavior and a comparatively small training–validation gap.

The proposed TinyCapsViT model, shown in Figure 4G, progressively converges to approximately 99% training accuracy while maintaining validation accuracy of approximately

89–90%. The training and validation losses stabilize at approximately 0.35 and 0.52, respectively. Although its validation performance does not exceed the higher-capacity models, TinyCapsViT achieves stable convergence using a substantially smaller architecture. This behavior reflects the intended accuracy–efficiency trade-off of the proposed model, where a modest reduction in classification performance is accepted in exchange for significantly lower model complexity and improved suitability for resource-constrained deployment.

The convergence characteristics are summarized in Table 2. The results demonstrate that larger and more complex architectures generally achieve high training accuracy but may exhibit varying degrees of overfitting. In comparison, TinyCapsViT maintains competitive validation performance while operating with substantially fewer trainable parameters. Therefore, the primary advantage of TinyCapsViT lies not in achieving the highest classification accuracy, but in providing a practical balance among classification performance, model complexity, and deployment efficiency for TinyML-based hyperspectral image classification.

Table 2. Comparison of training and validation performance across different models.

Model	Train Acc. (%)	Val. Acc. (%)	Train Loss	Val. Loss	Observation
CNN	~100	~90–91	~0.24	~0.45	Moderate overfitting, stable convergence
3D-CNN	~100	~90	~0.25	~0.50	Slight overfitting, slower convergence
ResNet	~100	~90–91	~0.24	~0.48	Stable and consistent performance
ViT	~100	~88–89	~0.24	~0.55	Higher overfitting, unstable validation
SSFTTNet	~100	~97–98	~0.28	~0.34	Strong convergence with minor validation fluctuations
BioLiteNet	~98–99	~83–85	~0.34	~0.64	Noticeable overfitting with validation fluctuations
TinyCapsViT	~99	~89–90	~0.35	~0.52	Balanced performance with lower complexity

4.5. Classification Performance Evaluation

The classification performance of the evaluated models across five public benchmark datasets and the custom Derrymore dataset is summarized in Table 3. Overall, all architectures achieve strong classification performance on most datasets, although differences are observed across datasets and model complexities. ResNet provides the strongest overall performance, achieving the highest OA on PaviaU (99.29%), Salinas (99.38%), Indian Pines (98.18%), and Derrymore (90.50%). On the Houston dataset, CNN, 3D-CNN, and ResNet achieve an OA of 100%, while SSFTTNet achieves 100% OA on KSC.

The conventional CNN-based architectures remain highly competitive. CNN achieves OA values above 97% on all public datasets, while 3D-CNN performs strongly on PaviaU, Houston, and Salinas but shows a comparatively lower OA of 93.91% on Indian Pines. ViT also demonstrates competitive performance, particularly on Houston, KSC, and Salinas, although its accuracy decreases to 95.75% on Indian Pines and 89.33% on Derrymore.

The additional lightweight HSI-specific baselines further demonstrate the effectiveness of compact architectures for spectral–spatial classification. SSFTTNet achieves strong performance across the public benchmarks, including 100% OA on KSC, 99.76% on Houston, and 98.66% on Salinas. It also obtains 97.23% on Indian Pines and 89.46% on Derrymore. BioLiteNet achieves 99.92% OA on KSC and 99.53% on Houston, but its performance

decreases on Salinas (96.17%) and Derrymore (86.08%), indicating greater variation in classification performance across datasets.

Table 3. Overall classification accuracy (OA%) and Kappa coefficient of the evaluated models across benchmark and custom hyperspectral image datasets. Bold values indicate the best performance for each dataset.

Model	PaviaU		Houston		KSC		Salinas		Indian Pines		Derrymore	
	OA	Kappa	OA	Kappa	OA	Kappa	OA	Kappa	OA	Kappa	OA	Kappa
CNN	98.92	0.9856	100.00	1.0000	97.85	0.9756	98.36	0.9817	97.31	0.9687	90.38	0.8358
3D-CNN	98.98	0.9864	100.00	1.0000	95.68	0.9509	98.49	0.9832	93.91	0.9292	90.31	0.8348
ResNet	99.29	0.9905	100.00	1.0000	99.65	0.9960	99.38	0.9931	98.18	0.9788	90.50	0.8376
ViT	97.54	0.9672	99.84	0.9983	99.01	0.9887	98.65	0.9849	95.75	0.9507	89.33	0.8180
BioLiteNet	96.99	0.9599	99.53	0.9949	99.92	0.9991	96.17	0.9573	95.81	0.9512	86.08	0.7641
SSFTTNet	98.39	0.9786	99.76	0.9975	100.00	0.9994	98.66	0.9851	97.23	0.9678	89.46	0.8196
TinyCapsViT	98.50	0.9800	99.61	0.9958	99.19	0.9908	98.44	0.9826	92.03	0.9079	89.00	0.8153

The proposed TinyCapsViT achieves 98.50% OA on PaviaU, 99.61% on Houston, 99.19% on KSC, and 98.44% on Salinas. Its performance decreases to 92.03% on Indian Pines and 89.00% on the Derrymore dataset. Therefore, TinyCapsViT does not consistently match or exceed the classification accuracy of higher-capacity architectures or all lightweight baselines. Instead, the proposed model is designed to achieve a practical trade-off between classification performance and computational efficiency. With only 2781 trainable parameters, TinyCapsViT maintains competitive performance on several benchmark datasets while substantially reducing model complexity and memory requirements.

The qualitative classification maps in Figure 5 further illustrate the spatial classification behavior of the evaluated models on the Pavia University dataset. The models generally preserve the major spatial structures and land-cover regions present in the ground truth, while differences are observed primarily around class boundaries and heterogeneous regions. TinyCapsViT retains the principal spatial structures despite its substantially reduced architecture. Together with the quantitative results, these observations demonstrate that TinyCapsViT intentionally trades a modest amount of classification accuracy for substantial reductions in model complexity, supporting its intended use in resource-constrained TinyML and edge-based hyperspectral image classification applications.

4.6. Feature Representation Analysis

The quality of the learned feature representations is evaluated using t-SNE visualization, as shown in Figure 6. To complement the qualitative analysis, Table 4 reports the Silhouette score, Davies–Bouldin Index (DBI), Calinski–Harabasz Index (CHI), intra-class distance, inter-class distance, and k-nearest neighbour (kNN) classification accuracy. Collectively, these metrics provide insight into the compactness, separability, and discriminative capability of the feature embeddings produced by each architecture.

Among the evaluated models, ResNet exhibits the strongest overall cluster structure, achieving the highest Silhouette score (0.2278), the lowest DBI (0.9175), and the highest CHI (24,909.40). These results indicate relatively compact and well-separated feature distributions and are consistent with its strong classification performance. CNN also produces well-structured embeddings, with a Silhouette score of 0.1887, DBI of 0.9490, and kNN accuracy of 0.9362. In contrast, 3D-CNN achieves the highest inter-class distance (103.32) and the highest kNN accuracy (0.9464), demonstrating strong class discrimination, although its higher intra-class distance (47.13) and DBI (1.1810) indicate comparatively less compact clusters.

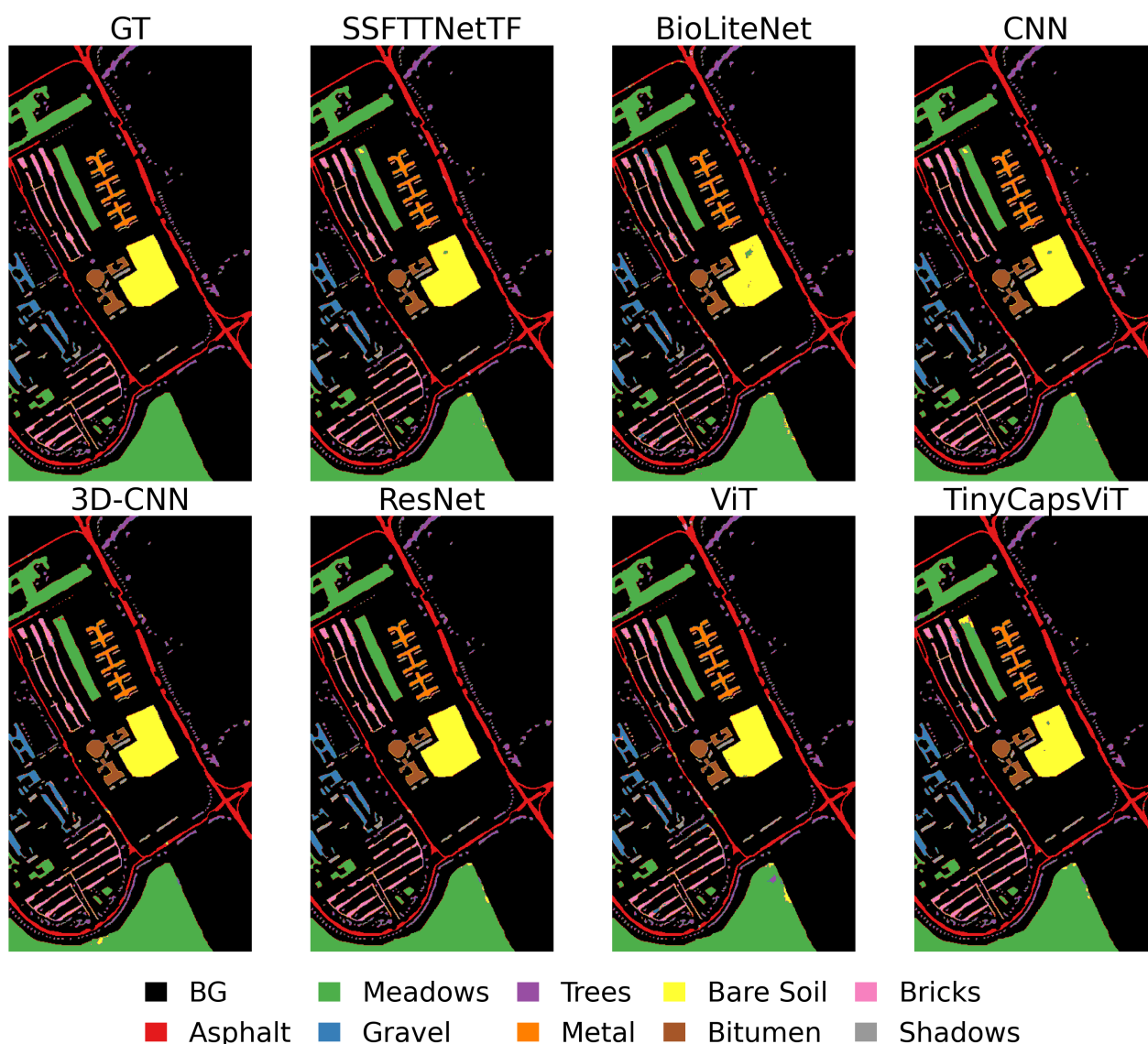


Figure 5. Pixel-wise classification maps of the Pavia University dataset obtained using the evaluated models. Consistent color coding is used across all maps to represent the nine land-cover classes, enabling qualitative comparison of spatial classification performance among different architectures.

Table 4. Quantitative comparison of t-SNE feature embedding quality across different models on the Derrymore dataset. Higher (↑) Silhouette score, CHI, inter-class distance, and kNN accuracy indicate better performance, whereas lower (↓) DBI and intra-class distance are preferred. Bold values indicate the best performance for each metric.

Model	Silhouette ↑	DBI ↓	CHI ↑	Intra ↓	Inter ↑	kNN Acc. ↑
CNN	0.1887	0.9490	21429.37	38.27	92.85	0.9362 ± 0.0022
3D-CNN	0.1473	1.1810	17540.80	47.13	103.32	0.9464 ± 0.0018
ResNet	0.2278	0.9175	24,909.40	32.87	85.29	0.9421 ± 0.0010
ViT	0.2177	1.0972	23,819.22	22.98	52.54	0.9052 ± 0.0017
SSFTTNet	0.1683	1.2102	20,036.59	43.03	90.38	0.9437 ± 0.0012
BioLiteNet	0.1144	1.7645	13,305.76	45.13	75.95	0.9109 ± 0.0021
TinyCapsViT	0.1442	1.5992	19,234.68	43.30	86.37	0.9249 ± 0.0025

ViT produces highly compact feature representations, achieving the lowest intra-class distance of 22.98 and a relatively high Silhouette score of 0.2177. However, its inter-class distance is the lowest among the evaluated models at 52.54, resulting in a comparatively lower

kNN accuracy of 0.9052. This indicates that although the individual class distributions are compact, the separation between different class clusters is comparatively limited.

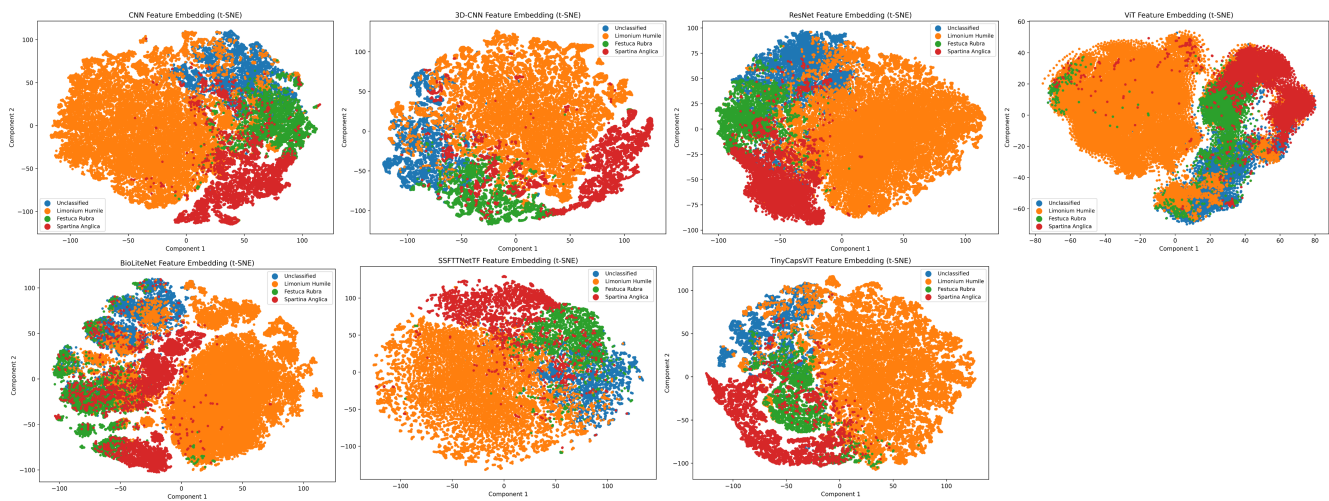


Figure 6. t-SNE visualization of feature embeddings generated by CNN, 3D-CNN, ResNet, ViT, SSFTTNet, BioLiteNet, and the proposed TinyCapsViT on the Derrymore dataset. Each point represents a feature sample projected into a two-dimensional space, with colors denoting different land-cover classes. The visualization highlights differences in intra-class compactness and inter-class separation among the evaluated models.

The lightweight HSI-specific architectures exhibit different feature-space characteristics. SSFTTNet achieves a Silhouette score of 0.1683, an inter-class distance of 90.38, and a high kNN accuracy of 0.9437. These results indicate effective class discrimination, although its DBI of 1.2102 suggests lower cluster compactness than ResNet and CNN. BioLiteNet exhibits comparatively weaker cluster separation, with the lowest Silhouette score of 0.1144 and the highest DBI of 1.7645. Its inter-class distance of 75.95 and kNN accuracy of 0.9109 further indicate reduced feature-space separability compared with SSFTTNet.

The proposed TinyCapsViT achieves a Silhouette score of 0.1442, DBI of 1.5992, intra-class distance of 43.30, and inter-class distance of 86.37, resulting in a competitive kNN accuracy of 0.9249. Although these feature-space metrics do not exceed those of the higher-capacity ResNet or SSFTTNet, TinyCapsViT maintains meaningful class separability and structured feature representations with a substantially smaller architecture. This reflects the intended accuracy–efficiency trade-off of the proposed model, where competitive discriminative capability is retained while substantially reducing model complexity for resource-constrained deployment.

The quantitative results in Table 4 and the visual distributions in Figure 6 demonstrate distinct feature-learning characteristics among the evaluated architectures. ResNet provides the strongest overall cluster quality, while 3D-CNN achieves the highest inter-class separation and kNN accuracy. SSFTTNet also demonstrates strong discriminative feature learning among the lightweight comparison models. Although TinyCapsViT does not achieve the best individual feature-space metric, it preserves competitive class discrimination while operating with substantially reduced computational and memory requirements. These findings further support the suitability of TinyCapsViT for resource-constrained hyperspectral image classification, where the balance between feature discrimination and computational efficiency is an important design consideration.

4.7. Complexity Analysis and Deployment Evaluation

The computational complexity of the evaluated models is compared in Table 5 in terms of trainable parameters, model size, MFLOPs, and inference latency. The results demonstrate

substantial differences in computational requirements among the evaluated architectures. SSFTTNet has the highest parameter count with 177,213 parameters, followed by ViT and ResNet with 85,509 and 85,029 parameters, respectively. BioLiteNet contains 38,885 parameters, while CNN and 3D-CNN require 31,653 and 26,533 parameters, respectively.

Table 5. Computational complexity comparison of the evaluated models on the Derrymore dataset in terms of trainable parameters, model size, MFLOPs, and inference latency. Bold values indicate the lowest computational requirement for each corresponding metric.

Model	Parameters	Size (MB)	MFLOPs	Latency (ms)
CNN	31,653	0.4043	2.006	64.62
3D-CNN	26,533	0.3448	16.018	65.33
ResNet	85,029	1.0393	13.713	65.12
ViT	85,509	1.0617	3.243	67.11
SSFTTNet	177,213	1.9564	11.403	74.36
BioLiteNet	38,885	0.5848	15.648	70.88
TinyCapsViT	2781	0.1431	0.633	65.64

In comparison, the proposed TinyCapsViT requires only 2781 trainable parameters, representing approximately $11.4\times$ and $9.5\times$ fewer parameters than CNN and 3D-CNN, respectively, and more than $30\times$ fewer parameters than ResNet and ViT. TinyCapsViT also achieves the smallest model size of 0.1431 MB and the lowest computational complexity of 0.633 MFLOPs. This substantial reduction in model complexity is particularly important for resource-constrained edge devices, where memory availability and computational capacity are limited. Although the measured inference latency of 65.64 ms is comparable to CNN, 3D-CNN, and ResNet, the considerably smaller parameter count and computational requirement demonstrate the lightweight characteristics of the proposed architecture.

To further investigate practical deployment performance, all models were converted to TensorFlow Lite (TFLite) and evaluated on the PaviaU dataset. The results are reported in Table 6. The TFLite evaluation considers classification accuracy, Kappa coefficient, compressed model size, and inference latency, providing a practical assessment of the suitability of each architecture for resource-constrained deployment.

As shown in Table 6, BioLiteNet achieves the highest TFLite classification accuracy of 97.06% and the highest Kappa coefficient of 0.9608, while SSFTTNet achieves 90.36% accuracy with a Kappa coefficient of 0.8720. However, these models require larger TFLite model sizes of 0.0648 MB and 0.2057 MB and exhibit inference latencies of 25.34 ms and 34.84 ms, respectively. ResNet achieves 88.75% accuracy with a Kappa coefficient of 0.8519 and a model size of 0.0945 MB.

Table 6. TensorFlow Lite (TFLite) deployment performance of the evaluated hyperspectral image classification models on the PaviaU dataset. Bold values indicate the best performance for each metric.

Model	Accuracy (%)	Kappa	Size (MB)	Latency (ms)
CNN	84.20	0.7961	0.0369	0.05
3D-CNN	84.87	0.8043	0.0336	0.09
ResNet	88.75	0.8519	0.0945	0.27
ViT	84.46	0.7980	0.1088	10.88
BioLiteNet	97.06	0.9608	0.0648	25.34
SSFTTNet	90.36	0.8720	0.2057	34.84
TinyCapsViT	88.85	0.8519	0.0388	0.312

TinyCapsViT achieves 88.85% classification accuracy and a Kappa coefficient of 0.8519 with a compact TFLite model size of only 0.0388 MB. Although CNN and 3D-CNN

provide smaller inference latencies of 0.05 ms and 0.09 ms, respectively, their classification accuracies are lower at 84.20% and 84.87%. TinyCapsViT also achieves a lower deployment latency than the transformer-based ViT, BioLiteNet, and SSFTTNet models. In particular, its 0.312 ms inference latency is lower than ViT (10.88 ms), BioLiteNet (25.34 ms), and SSFTTNet (34.84 ms).

The results demonstrate that no single architecture provides the best performance across all deployment metrics. BioLiteNet achieves the highest TFLite classification accuracy, whereas CNN and 3D-CNN provide the lowest inference latencies. In contrast, TinyCapsViT combines competitive classification accuracy with a small memory footprint, low computational complexity, and practical inference latency. These characteristics demonstrate the intended accuracy–efficiency trade-off of TinyCapsViT and support its suitability for resource-constrained TinyML and edge-based hyperspectral image classification applications.

4.8. Robustness Analysis

To evaluate the reliability of the models under degraded input conditions, robustness experiments were conducted using Gaussian noise perturbation and spectral corruption. The analysis includes CNN, 3D-CNN, ResNet, ViT, BioLiteNet, SSFTTNet, and the proposed TinyCapsViT across the six evaluated datasets. Gaussian noise was introduced at four levels ($\sigma = 0.01, 0.02, 0.05$, and 0.1), while spectral corruption was evaluated at levels of $0.05, 0.1, 0.2$, and 0.3 . The average classification accuracies across these perturbation levels are reported in Table 7 and Table 8, respectively.

Table 7. Average classification accuracy (%) under Gaussian noise perturbations ($\sigma = 0.01, 0.02, 0.05$, and 0.1) across the evaluated datasets. Bold values indicate the highest average robustness for each dataset.

Dataset	CNN	3D-CNN	ResNet	ViT	BioLiteNet	SSFTTNet	TinyCapsViT
PaviaU	52.30	55.44	67.31	62.83	47.06	44.38	61.31
Houston	48.21	23.09	52.38	54.22	44.33	45.88	55.52
KSC	9.65	7.45	3.84	15.24	12.50	11.96	11.26
Derrymore	73.33	62.96	70.08	66.04	78.55	79.56	82.96
Salinas	81.55	52.56	79.51	80.05	81.32	79.54	41.32
Indian Pines	30.03	19.45	38.39	25.42	31.67	29.63	26.54

Table 8. Average classification accuracy (%) under spectral corruption levels of $0.05, 0.1, 0.2$, and 0.3 across the evaluated datasets. Bold values indicate the highest average robustness for each dataset.

Dataset	CNN	3D-CNN	ResNet	ViT	BioLiteNet	SSFTTNet	TinyCapsViT
PaviaU	92.00	90.32	93.20	91.72	90.53	90.73	87.54
Houston	94.33	90.74	92.67	94.32	91.67	92.71	86.63
KSC	69.97	63.10	78.43	70.33	72.15	70.21	59.99
Derrymore	85.33	85.40	85.33	83.57	82.65	83.54	84.75
Salinas	93.92	87.84	92.75	94.14	95.32	91.72	87.87
Indian Pines	89.47	79.86	90.37	87.88	84.38	80.18	73.70

Under Gaussian noise perturbation, no single architecture consistently achieves the highest robustness across all datasets. ResNet obtains the highest average accuracy on PaviaU (67.31%) and Indian Pines (38.39%), while ViT performs best on KSC (15.24%). CNN achieves the highest average robustness on Salinas (81.55%), although BioLiteNet provides a closely comparable result of 81.32%. TinyCapsViT achieves the highest average accuracy on Houston (55.52%) and Derrymore (82.96%), demonstrating that the lightweight architecture can retain competitive robustness under Gaussian perturbations for some

datasets. However, its robustness is dataset-dependent, as evidenced by the lower average accuracy of 41.32% on Salinas.

Figure 7 provides a detailed view of the variation in classification accuracy with increasing Gaussian noise on PaviaU. The results further demonstrate that robustness to additive noise varies considerably across architectures and is not determined solely by model complexity. In particular, the performance of TinyCapsViT remains competitive with several comparison models, although ResNet provides stronger overall robustness on this dataset.

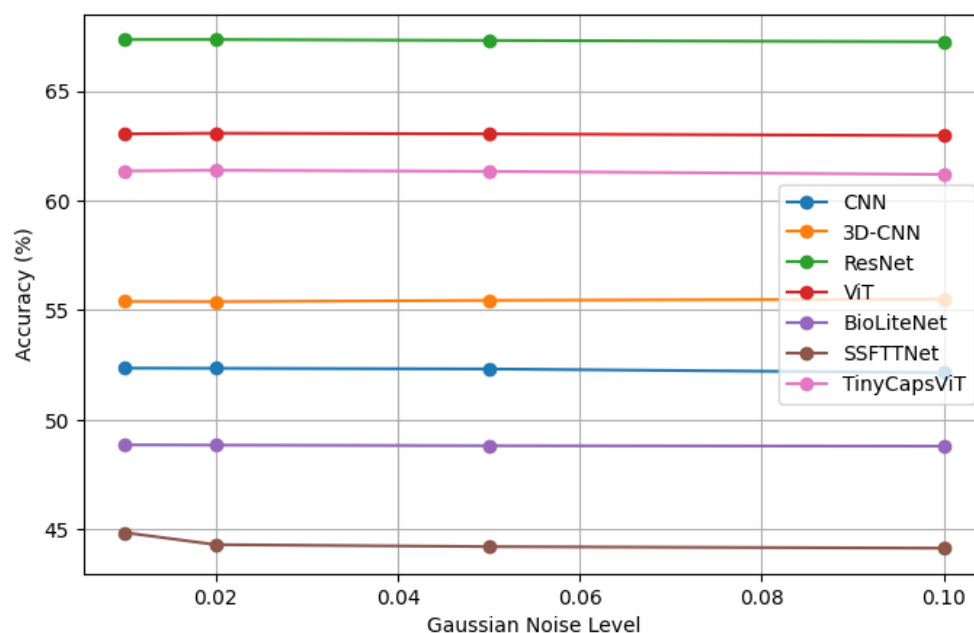


Figure 7. Classification performance under increasing Gaussian noise levels ($\sigma = 0.01, 0.02, 0.05$, and 0.1) on the PaviaU dataset for the evaluated models.

The spectral corruption results in Table 8 show that ResNet provides the strongest overall robustness, achieving the highest average accuracy on PaviaU (93.20%), KSC (78.43%), and Indian Pines (90.37%). CNN achieves the highest performance on Houston (94.33%), although ViT produces a nearly identical result of 94.32%. BioLiteNet performs best on Salinas with an average accuracy of 95.32%, while 3D-CNN achieves the highest result on Derrymore at 85.40%.

TinyCapsViT maintains average accuracies of 87.54%, 86.63%, 59.99%, 84.75%, 87.87%, and 73.70% on PaviaU, Houston, KSC, Derrymore, Salinas, and Indian Pines, respectively. Although these results demonstrate that the proposed model retains useful classification capability under spectral degradation, its performance decreases more rapidly than that of higher-capacity architectures under severe corruption, particularly on KSC and Indian Pines.

As illustrated in Figure 8, increasing spectral corruption progressively reduces classification performance. ResNet maintains the most stable performance on PaviaU, whereas TinyCapsViT exhibits a larger degradation as the corruption severity increases. This behavior can be attributed to the design trade-off introduced by the ultra-lightweight architecture. TinyCapsViT contains only 2781 trainable parameters and therefore has substantially lower representational capacity and redundancy than higher-capacity models such as ResNet. Under severe spectral corruption, the reduced feature capacity limits the ability of the model to compensate for heavily degraded spectral information.

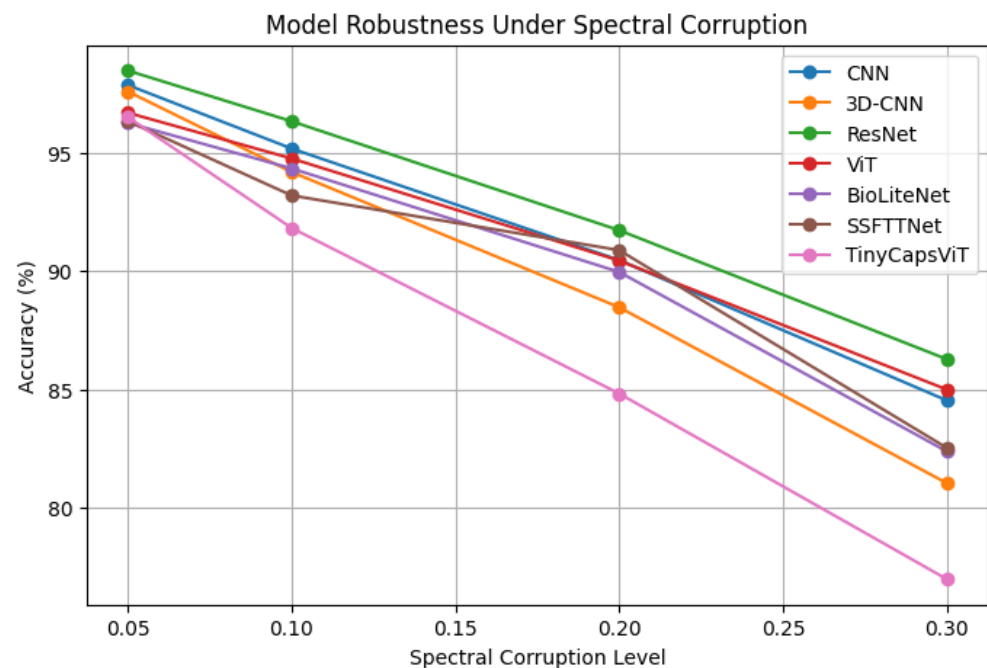


Figure 8. Classification performance under increasing spectral corruption levels (0.05, 0.1, 0.2, and 0.3) on the PaviaU dataset for the evaluated models.

The capsule-inspired refinement block is intended to improve feature discrimination and preserve informative relationships within the compact representation; however, it does not provide explicit invariance to severe spectral perturbations. Consequently, its benefits should be interpreted in the context of efficient feature refinement rather than guaranteed robustness against substantial spectral degradation. Incorporating noise-aware training, spectral augmentation, or corruption-specific regularization may further improve the robustness of TinyCapsViT and represents a potential direction for future work.

Overall, the robustness experiments demonstrate that higher-capacity models, particularly ResNet, generally provide stronger resilience to severe spectral corruption. TinyCapsViT does not consistently outperform these architectures in robustness; instead, it provides a trade-off between robustness, classification performance, and substantially reduced computational complexity. This trade-off is consistent with the primary objective of TinyCapsViT: enabling practical hyperspectral image classification on resource-constrained TinyML and edge platforms.

5. Discussion

The experimental results demonstrate that the evaluated deep learning architectures exhibit distinct trade-offs among classification performance, feature representation, robustness, computational complexity, and deployment efficiency. Among the higher-capacity models, ResNet provides the strongest overall classification performance across the evaluated datasets. It achieves the highest accuracy on PaviaU, Salinas, Indian Pines, and Derrymore, while also demonstrating strong robustness under Gaussian noise and spectral corruption. The feature representation analysis further shows that ResNet produces well-structured embeddings, achieving the highest Silhouette score and Calinski–Harabasz Index and the lowest Davies–Bouldin Index. However, these advantages are accompanied by a considerably larger parameter count and memory footprint, which are important considerations for resource-constrained deployment.

CNN and 3D-CNN also demonstrate strong classification capability by effectively extracting local spectral–spatial information. CNN provides consistently high classification

accuracy while maintaining relatively low computational complexity and inference latency. The 3D-CNN architecture achieves particularly strong inter-class separation in the feature representation analysis; however, its three-dimensional convolution operations result in substantially higher computational complexity. ViT performs competitively on several datasets and produces compact feature representations, but its validation behavior and robustness vary across datasets, indicating greater sensitivity to the available training data and input perturbations.

The lightweight HSI-specific models, BioLiteNet and SSFTTNet, provide additional insight into the relationship between model design and efficiency. SSFTTNet achieves strong classification performance on several public datasets and demonstrates high feature discrimination, including a kNN feature-space accuracy of 0.9437. BioLiteNet also performs strongly on selected datasets and achieves the highest TFLite classification accuracy of 97.06% on PaviaU. However, these models do not necessarily translate their lightweight design into the lowest computational or deployment cost in the present implementation. BioLiteNet requires 38,885 parameters and 15.648 MFLOPs, while SSFTTNet requires 177,213 parameters and 11.403 MFLOPs. Their TFLite inference latencies are also higher than that of TinyCapsViT, demonstrating that parameter count alone does not fully determine practical deployment efficiency.

The proposed TinyCapsViT is designed specifically to balance classification capability with stringent computational constraints. With only 2781 trainable parameters and 0.633 MFLOPs, it has the lowest parameter count and computational complexity among the evaluated architectures. This corresponds to approximately $11.4\times$ fewer parameters than CNN, $9.5\times$ fewer than 3D-CNN, and more than $30\times$ fewer than ResNet and ViT. Despite this substantial reduction, TinyCapsViT maintains competitive classification performance, achieving 98.50%, 99.61%, 99.19%, and 98.44% OA on PaviaU, Houston, KSC, and Salinas, respectively. Its performance is lower on Indian Pines and Derrymore, indicating that the proposed architecture intentionally trades a modest amount of classification accuracy for substantial reductions in model complexity and memory requirements.

The feature representation analysis supports this interpretation. TinyCapsViT achieves a kNN accuracy of 0.9249 with an inter-class distance of 86.37, indicating that meaningful discriminative representations are retained despite the substantial reduction in model capacity. Although its clustering metrics do not exceed those of ResNet or SSFTTNet, the results demonstrate that the combination of lightweight attention and capsule-inspired refinement can preserve useful spectral–spatial information within a highly compact representation. Therefore, the primary advantage of TinyCapsViT is not superior feature separability over higher-capacity models, but its ability to retain competitive discriminative capability under a substantially reduced computational budget.

The robustness experiments further highlight this accuracy–efficiency trade-off. ResNet generally provides the strongest resilience to spectral degradation, while the performance of TinyCapsViT varies across datasets. Under Gaussian noise, TinyCapsViT achieves the highest average accuracy on Houston and Derrymore, but lower robustness is observed on datasets such as Salinas. Similarly, under spectral corruption, TinyCapsViT experiences greater degradation than higher-capacity architectures, particularly on KSC and Indian Pines. This behavior can be attributed partly to its limited representational capacity and redundancy. The capsule-inspired refinement mechanism improves feature discrimination within the compact architecture but does not explicitly provide invariance to severe spectral corruption. Consequently, robustness under substantial spectral degradation remains an important area for further improvement.

The hardware evaluation in Table 9 further demonstrates that computational complexity does not translate directly into identical latency rankings across hardware platforms

Table 10. CNN achieves the lowest latency on the workstation, Intel NUC, and Jetson Xavier NX, whereas TinyCapsViT achieves the lowest measured latency on the Raspberry Pi 3 at 0.36 ms. TinyCapsViT also maintains low latency on the Intel NUC and Jetson Xavier NX, substantially outperforming ViT, BioLiteNet, and SSFTTNet in inference time. These results indicate that the compact architecture is particularly advantageous on highly resource-constrained hardware, where reduced computational and memory requirements become increasingly important.

Table 9. Average inference latency (ms) of the evaluated models across different hardware platforms. Bold values indicate the lowest latency on each platform.

Model	Workstation (ms)	Intel NUC (ms)	Jetson Xavier NX (ms)	Raspberry Pi 3 (ms)
CNN	0.03	0.18	0.08	3.63
ResNet	0.17	0.65	0.24	4.92
ViT	1.58	3.42	1.21	18.54
BioLiteNet	25.34	28.23	26.22	34.16
SSFTTNet	34.84	37.45	41.57	48.43
TinyCapsViT	0.312	0.34	0.24	0.36

Table 10. Hardware configuration used for training and deployment evaluation on workstation and edge computing platforms.

Workstation	
Processor	Intel Core i9
GPU	NVIDIA Tesla P100
RAM	128 GB
Operating System	Windows 11
Intel NUC	
Processor	Intel Core i5
GPU	Intel Iris Xe Graphics
RAM	16 GB
Operating System	Ubuntu 22.04 LTS
Jetson Xavier NX	
Processor	6-core NVIDIA Carmel ARM v8.2 64-bit CPU
GPU	384-core NVIDIA Volta GPU with 48 Tensor Cores
RAM	8 GB LPDDR4x
Operating System	JetPack SDK (Ubuntu-based)
Raspberry Pi 3 Model B+	
Processor	Quad-core ARM Cortex-A53 @ 1.4 GHz
GPU	Broadcom VideoCore IV
RAM	1 GB LPDDR2
Operating System	Raspberry Pi OS

Taken together, the experimental results show that no single architecture dominates across all evaluation criteria. ResNet provides the strongest overall classification and robustness performance, 3D-CNN and SSFTTNet demonstrate strong feature discrimination, BioLiteNet achieves strong TFLite classification accuracy, and CNN provides highly efficient inference on several hardware platforms. In contrast, TinyCapsViT targets a different operating point by combining competitive classification performance with only 2781 parameters, low computational complexity, a small memory footprint, and practical edge

inference. This balance makes the proposed architecture particularly relevant for embedded hyperspectral and multispectral sensing applications where computational resources, memory, and power consumption are constrained.

To further demonstrate the evolution of our research, Table 11 compares the proposed TinyCapsViT with our previously published CrossCapsViT architecture. CrossCapsViT was designed to maximize classification performance, achieving an overall accuracy of 99.89%, but at the expense of high computational complexity, with 6.626 million trainable parameters, 0.1209 GFLOPs, and a model size of 26.5 MB.

Table 11. Comparison illustrating the evolution of CrossCapsViT into the proposed TinyCapsViT architecture on Workstation.

Metric	CrossCapsViT	TinyCapsViT
Overall Accuracy (%)	99.89	98.50
Kappa Coefficient	0.998	0.980
Trainable Parameters	6.626 M	2781
Model Size (MB)	26.5	0.143
FLOPs (G)	0.1209	0.0006
Inference Time (ms/sample)	22.32	0.312
Convergence Epochs	140	90
Peak GPU Memory (GB)	7.8	1.2

In contrast, TinyCapsViT was specifically developed for TinyML deployment by emphasizing computational efficiency while maintaining competitive classification performance. Compared with CrossCapsViT, TinyCapsViT reduces the number of trainable parameters by approximately 99.96%, decreases the model size by nearly 185 $\times$, and lowers the computational complexity by about 93 $\times$. Moreover, the TensorFlow Lite implementation achieves an inference time of only 0.312 ms per sample, making it well suited for real-time deployment on resource-constrained edge devices.

Although TinyCapsViT exhibits a modest reduction in classification accuracy (98.50% versus 99.89%), the significant gains in efficiency, memory footprint, and deployment capability demonstrate that it provides a more practical solution for TinyML-based hyperspectral image classification. This comparison highlights the progression of our research from a performance-oriented architecture to a lightweight, deployment-oriented framework for embedded environmental monitoring applications.

Despite these advantages, several limitations remain. TinyCapsViT exhibits reduced robustness under severe spectral corruption and does not consistently match the classification accuracy of higher-capacity architectures or all lightweight baselines. Furthermore, deployment latency depends strongly on hardware characteristics, software frameworks, and operator-level optimization; therefore, a low parameter count does not necessarily guarantee the lowest latency on every platform. Future work will investigate quantization-aware training, hardware-aware architecture optimization, spectral augmentation, and noise-aware training to improve robustness and deployment efficiency without substantially increasing model complexity. Additional evaluation on embedded sensing platforms and real-time UAV-based environmental monitoring scenarios will further assess the practical scalability of the proposed approach.

6. Conclusions and Future Work

This work presented TinyCapsViT, an ultra-lightweight capsule-inspired vision transformer for resource-constrained hyperspectral image classification, and evaluated its performance against CNN, 3D-CNN, ResNet, ViT, BioLiteNet, and SSFTTNet across multiple benchmark datasets and a custom UAV-acquired hyperspectral dataset. The

experimental results demonstrated that higher-capacity architectures, particularly ResNet, generally provide stronger overall classification performance and robustness. Feature representation analysis further showed that ResNet produces compact and well-separated embeddings, whereas TinyCapsViT retains meaningful class separability using a substantially smaller architecture.

The proposed TinyCapsViT achieved competitive classification performance while substantially reducing model complexity. With only 2781 trainable parameters and low computational requirements, the model provides a compact alternative to conventional CNN and transformer-based architectures. TensorFlow Lite evaluation and hardware deployment experiments further demonstrated the feasibility of executing TinyCapsViT on resource-constrained edge platforms. Although TinyCapsViT does not consistently achieve the highest classification accuracy among the evaluated models, it provides a practical trade-off between classification performance, computational complexity, memory requirements, and deployment efficiency, supporting its use in TinyML-based hyperspectral image classification applications.

The robustness experiments demonstrated that TinyCapsViT retains useful classification capability under input perturbations, with competitive performance under Gaussian noise on selected datasets. However, its robustness is dataset-dependent, and its performance degrades more rapidly than that of higher-capacity architectures under severe spectral corruption. This behavior reflects the inherent trade-off introduced by the ultra-lightweight design, where reduced representational capacity is accepted in exchange for substantial reductions in computational and memory requirements. Deployment experiments on the Intel NUC, Jetson Xavier NX, and Raspberry Pi 3 further confirmed the practical feasibility of the proposed architecture for edge-AI applications.

Future work will focus on improving robustness under severe spectral degradation and limited training conditions. Additional research will investigate quantization-aware training, pruning, adaptive spectral attention mechanisms, spectral augmentation, noise-aware training, and hardware-aware optimization strategies to further improve deployment efficiency and robustness without substantially increasing model complexity. Future extensions will also explore real-time onboard hyperspectral image processing, multispectral classification, multimodal sensor fusion, and autonomous edge-based environmental monitoring applications.

Author Contributions: Conceptualization, S.D., and G.D.; methodology, S.D.; software, M.M. and S.D.; validation, S.D., M.M. and G.D.; formal analysis, S.D. and M.M.; investigation, S.D., M.M., T.N., E.O. and G.D.; resources, T.N., E.O. and G.D.; data curation, M.M. and S.D.; writing—original draft preparation, S.D.; writing—review and editing, S.D., M.M., T.N., E.O. and G.D.; visualization, S.D.; supervision, T.N., E.O. and G.D.; project administration, G.D.; funding acquisition, T.N., E.O. and G.D. All authors have read and agreed to the published version of the manuscript.

Funding: The authors gratefully acknowledge the financial support provided by the University of Limerick through the Science and Engineering Early Career PhD Scholarship Programme. This work was also supported by the BLUEPOINT project (EAPA_0035/2022), co-financed by the European Regional Development Fund (ERDF) through the Interreg Atlantic Area Programme, and by the Climate+ Co-Centre, funded by Research Ireland under Grant No. 22/CC/11103.

Data Availability Statement: The datasets generated during this study will be made available by the authors upon request.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Stuart, M.B.; McGonigle, A.J.; Willmott, J.R. Hyperspectral imaging in environmental monitoring: A review of recent developments and technological advances in compact field deployable systems. *Sensors* **2019**, *19*, 3071. [\[CrossRef\]](#) [\[PubMed\]](#)
2. Bhargava, A.; Sachdeva, A.; Sharma, K.; Alsharif, M.H.; Uthansakul, P.; Uthansakul, M. Hyperspectral imaging and its applications: A review. *Heliyon* **2024**, *10*, e33208. [\[CrossRef\]](#) [\[PubMed\]](#)
3. Li, S.; Song, W.; Fang, L.; Chen, Y.; Ghamisi, P.; Benediktsson, J.A. Deep learning for hyperspectral image classification: An overview. *IEEE Trans. Geosci. Remote Sens.* **2019**, *57*, 6690–6709. [\[CrossRef\]](#)
4. Dalai, S.; Moreno, M.; Riordan, J.; Newe, T.; O'Connell, E.; Dooly, G. Enhancing Hyperspectral Image Classification Through Reinforcement Learning Guided Active Learning. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2026**, *19*, 16314–16332. [\[CrossRef\]](#)
5. Moreno, M.; Dalai, S.; Cott, G.; Bartlett, B.; Santos, M.; Dorian, T.; Riordan, J.; McGonigle, C.; Sacchetti, F.; Dooly, G. Multi-camera machine learning for salt marsh species classification and mapping. *Remote Sens.* **2025**, *17*, 1964. [\[CrossRef\]](#)
6. Dalai, S.; Moreno, M.; Irfan, M.; Dooley, A.; Newe, T.; Dooly, G.; O'Connell, E. Enhancing Saltmarsh Monitoring Using UAV Based Hyperspectral Imaging Sensor: A Case Study from Derrymore Island. In Proceedings of the 2025 35th Irish Signals and Systems Conference (ISSC), Letterkenny, Ireland, 9–10 June 2025; pp. 1–7. [\[CrossRef\]](#)
7. Camps-Valls, G.; Tuia, D.; Bruzzone, L.; Benediktsson, J.A. Advances in hyperspectral image classification: Earth monitoring with statistical learning methods. *IEEE Signal Process. Mag.* **2013**, *31*, 45–54.
8. Ahmad, M.; Distefano, S.; Khan, A.M.; Mazzara, M.; Li, C.; Li, H.; Aryal, J.; Ding, Y.; Vivone, G.; Hong, D. A comprehensive survey for hyperspectral image classification: The evolution from conventional to transformers and mamba models. *Neurocomputing* **2025**, *644*, 130428. [\[CrossRef\]](#)
9. Praveen, B.; Menon, V. HYPER-VIT: A novel light-weighted visual transformer-based supervised classification framework for hyperspectral remote sensing applications. In Proceedings of the 2022 12th Workshop on Hyperspectral Imaging and Signal Processing: Evolution in Remote Sensing (WHISPERS), Rome, Italy, 13–16 September 2022; pp. 1–5.
10. Gu, Q.; Luan, H.; Huang, K.; Sun, Y. Hyperspectral image classification using multi-scale lightweight transformer. *Electronics* **2024**, *13*, 949. [\[CrossRef\]](#)
11. Zou, J.; He, W.; Zhang, H. PSFormer: Pyramid superpixel transformer for hyperspectral image classification. *IEEE Trans. Geosci. Remote Sens.* **2024**, *62*, 5532816.
12. Melgani, F.; Bruzzone, L. Classification of hyperspectral remote sensing images with support vector machines. *IEEE Trans. Geosci. Remote Sens.* **2004**, *42*, 1778–1790. [\[CrossRef\]](#)
13. Pal, M. Random forest classifier for remote sensing classification. *Int. J. Remote Sens.* **2005**, *26*, 217–222. [\[CrossRef\]](#)
14. Roy, S.K.; Krishna, G.; Dubey, S.R.; Chaudhuri, B.B. HybridSN: Exploring 3-D–2-D CNN feature hierarchy for hyperspectral image classification. *IEEE Geosci. Remote Sens. Lett.* **2019**, *17*, 277–281. [\[CrossRef\]](#)
15. Zhong, Z.; Li, J.; Luo, Z.; Chapman, M. Spectral–spatial residual network for hyperspectral image classification: A 3-D deep learning framework. *IEEE Trans. Geosci. Remote Sens.* **2017**, *56*, 847–858. [\[CrossRef\]](#)
16. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
17. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv* **2020**, arXiv:2010.11929.
18. Hong, D.; Han, Z.; Yao, J.; Gao, L.; Zhang, B.; Plaza, A.; Chanussot, J. SpectralFormer: Rethinking hyperspectral image classification with transformers. *IEEE Trans. Geosci. Remote Sens.* **2021**, *60*, 1–15. [\[CrossRef\]](#)
19. Jia, S.; Zhao, B.; Tang, L.; Feng, F.; Wang, W. Spectral–spatial classification of hyperspectral remote sensing image based on capsule network. *J. Eng.* **2019**, *2019*, 7352–7355. [\[CrossRef\]](#)
20. Paoletti, M.E.; Moreno-Alvarez, S.; Haut, J.M. Multiple attention-guided capsule networks for hyperspectral image classification. *IEEE Trans. Geosci. Remote Sens.* **2021**, *60*, 1–20. [\[CrossRef\]](#)
21. Wang, W.; Xu, Y.; Xu, Z.; Kong, C.; Niu, X.; Huang, J. Multiscale spectral–spatial capsule neural network for hyperspectral image classification. In *Proceedings of the International Conference on the Efficiency and Performance Engineering Network*; Springer: Berlin/Heidelberg, Germany, 2023; pp. 185–194.
22. Ramnarayan, S.; Sharma, S.; Abidi, A.I.; Chohan, J.S.; Singh, D.; Thallal, S. 3D Convolutional Capsule Network Integrating Spectral–Spatial Features for Hyperspectral Image Classification. In Proceedings of the 2025 7th International Symposium on Advanced Electrical and Communication Technologies (ISAECT), Mohali, India, 18–20 December 2025; pp. 1–6.
23. Sabour, S.; Frosst, N.; Hinton, G.E. Dynamic routing between capsules. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*; Curran Associates Inc.: Red Hook, NY, USA, 2017; Volume 30, pp. 3859–3869.
24. Song, Y.; Ding, X.; Liu, L.; Liao, K.; Jiang, W.; Liu, Z.; Ma, J. Efficient capsule network with attention mechanism for hyperspectral remote sensing classification. *Geocarto Int.* **2026**, *41*, 2614736. [\[CrossRef\]](#)
25. Gao, Z.; Wang, J.; Shen, H.; Dou, Z.; Zhang, X.; Huang, K. Discrete Wavelet Transform-Based Capsule Network for Hyperspectral Image Classification. *arXiv* **2025**, arXiv:2501.04643.

26. Warden, P.; Situnayake, D. *Tinyml: Machine Learning with Tensorflow Lite on Arduino and Ultra-Low-Power Microcontrollers*; O'Reilly Media: Sebastopol, CA, USA, 2019.
27. Hütner, J.V.S.; Viel, F.; Zeferino, C.A.; Bezerra, E.A. TinyML Applied in Hyperspectral Image Classification on COTS Microcontroller. In Proceedings of the 2024 XIV Brazilian Symposium on Computing Systems Engineering (SBESC), Recife, Brazil, 26–29 November 2024; pp. 1–6.
28. Lamaakal, I.; Yahyati, C.; Ouahbi, I.; El Makkaoui, K.; Maleh, Y. A survey of model compression techniques for TinyML applications. In Proceedings of the 2025 International Conference on Circuit, Systems and Communication (ICCSC), Fez, Morocco, 19–20 June 2025; pp. 1–6.
29. Preetha, J.; Thamizharasan, P.; Ajay Surya, B.; Thamizharasan, P. Distilling Intelligence: Deploying Lightweight Neural Networks on ESP32 for Edge AI. In Proceedings of the 2025 IEEE International Conference on Computer Vision and Machine Intelligence (CVMII), Rourkela, India, 12–13 October 2025; pp. 1–6.
30. Resonon. Pika L Hyperspectral Imaging System. Available online: <https://resonon.com/Pika-L> (accessed on 6 March 2025).
31. DJI. DJI M300 RTK Drone. Available online: <https://www.dji.com/ie/support/product/matrice-300> (accessed on 29 March 2025).
32. Amin, K. Hyperspectral Remote Sensing Datasets: Indian Pines, Pavia University, Botswana and Salinas. Available online: <https://ieee-dataport.org/documents/hyperspectral-remote-sensing-datasets-indian-pines-pavia-university-botswana-and-salinas> (accessed on 29 March 2025).
33. Debes, C.; Merentitis, A.; Heremans, R.; Hahn, J.; Frangiadakis, N.; Van Kasteren, T.; Liao, W.; Bellens, R.; Pižurica, A.; Gautama, S.; et al. Hyperspectral and LiDAR data fusion: Outcome of the 2013 GRSS data fusion contest. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2014**, *7*, 2405–2418. [CrossRef]
34. Green, R.O. Imaging spectroscopy and the airborne visible/infrared imaging spectrometer (AVIRIS). *Remote Sens. Environ.* **1998**, *65*, 227–248. [CrossRef]
35. Baumgardner, M.F.; Biehl, L.L.; Landgrebe, D.A. 220 Band AVIRIS Hyperspectral Image Data Set: June 12, 1992 Indian Pine Test Site 3. 2015. Available online: <https://api.semanticscholar.org/CorpusID:114694234> (accessed on 29 March 2025).
36. Zeng, B.; Suwen, C.; Jialang, L.; Yanming, G.; Yingmei, W.; Huimin, Y.; Bin, X.; Yaowen, H.; Li, L. BioLiteNet: A biomimetic lightweight hyperspectral image classification model. *Remote Sens.* **2025**, *17*, 2833. [CrossRef]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.