applied sciences

MDPI

Article

Optimizing Mobile Vision Transformers for Land Cover Classification

Papia F. Rozario [1,*], Ravi Gadgil [2], Junsu Lee [3], Rahul Gomes [3,*], Paige Keller [3], Yiheng Liu [3], Gabriel Sipos [1], Grace McDonnell [1], Westin Impola [4] and Joseph Rudolph [5]

[1] Department of Geography and Anthropology, University of Wisconsin-Eau Claire, Phillips Science Hall 255, 101 Roosevelt Ave., Eau Claire, WI 54701, USA; siposgr7007@uwec.edu (G.S.); mcdonngj7536@uwec.edu (G.M.)
[2] Department of Computer Science, San Jose State University, San Jose, CA 95192, USA; ravi.gadgil@sjsu.edu
[3] Department of Computer Science, University of Wisconsin-Eau Claire, Eau Claire, WI 54701, USA; leej8693@uwec.edu (J.L.); kellerpa2929@uwec.edu (P.K.); yihengl8313@uwec.edu (Y.L.)
[4] Department of Computer Science, University of Wisconsin-River Falls, River Falls, WI 54022, USA; westinimpola1@gmail.com
[5] Department of Computer Science, Connecticut College, New London, CT 06320, USA; jrudolph1@conncoll.edu
* Correspondence: rozaripf@uwec.edu (P.F.R.); gomesr@uwec.edu (R.G.)

Abstract: Image classification in remote sensing and geographic information system (GIS) data containing various land cover classes is essential for efficient and sustainable land use estimation and other tasks like object detection, localization, and segmentation. Deep learning (DL) techniques have shown tremendous potential in the GIS domain. While convolutional neural networks (CNNs) have dominated image analysis, transformers have proven to be a unifying solution for several AI-based processing pipelines. Vision transformers (ViTs) can have comparable and, in some cases, better accuracy than a CNN. However, they suffer from a significant drawback associated with the excessive use of training parameters. Using trainable parameters generously can have multiple advantages ranging from addressing model scalability to explainability. This can have a significant impact on model deployment in edge devices with limited resources, such as drones. In this research, we explore, without using pre-trained weights, how the inherent structure of vision transformers behaves with custom modifications. To verify our proposed approach, these architectures are trained on multiple land cover datasets. Experiments reveal that a combination of lightweight convolutional layers, including ShuffleNet, along with depthwise separable convolutions and average pooling can reduce the trainable parameters by 17.85% and yet achieve higher accuracy than the base mobile vision transformer (MViT). It is also observed that utilizing a combination of convolution layers along with multi-headed self-attention layers in MViT variants provides better performance for capturing local and global features, unlike the standalone ViT architecture, which utilizes almost 95% more parameters than the proposed MViT variant.

Keywords: vision transformers; MViT; ShuffleNet; CNN; land cover classification

Citation: Rozario, P.F.; Gadgil, R.; Lee, J.; Gomes, R.; Keller, P.; Liu, Y.; Sipos, G.; McDonnell, G.; Impola, W.; Rudolph, J. Optimizing Mobile Vision Transformers for Land Cover Classification. *Appl. Sci.* **2024**, *14*, 5920. https://doi.org/10.3390/app14135920

Academic Editor: Nikolaos Koukouzas

Received: 3 May 2024
Revised: 1 July 2024
Accepted: 3 July 2024
Published: 6 July 2024

1. Introduction

Deep learning is a subset of machine learning that has exploded in popularity and has become prominent in many industries around the world today because it is based on powerful artificial neural networks that are capable of learning and performing complex tasks such as natural language processing [1] and image classification [2]. As a result, deep learning has also become useful and ubiquitous for remote sensing tasks because it has the computational power to extract compact features from data with high spectral and spatial resolution for purposes such as object detection, land use and landscape classification [3–5], and multi-class classification [6–8]. Based on the successful performance of parallel text

processing, transformers [9] have gained significant momentum in the deep learning domain. Transformer neural networks are models that are highly adept at learning context from input data using parallel multi-head attention mechanisms [10]. As a result, they can be applied to remote sensing tasks such as multi-class classification of images because transformers can extract long-range dependencies from the relationships between elements of an image sequence to generate global representations.

Vision transformers [11] split an image into patches before flattening those patches and converting them into linear embeddings with positional embeddings added to them. The sequence is then fed into a standard transformer encoder, which consists of alternating multi-head self-attention (MHSA) layers that map the input sequence to linear embeddings, which are decoded to produce the logits. Since ViT models deal with image patches at a global level using the self-attention mechanism, they can attain high levels of performance because they can capture contextual information and long-range dependencies between image pixels. Vision transformers (ViTs) and transformer architectures have already been applied in the remote sensing domain for tasks such as classifying various types of high-resolution UAV images. For example, the researchers of the AiTLAS Benchmark Arena train ViT and nine other representative architectures that are either convolutional neural networks (CNNs) or transformer-based on various multi-class classification datasets [12]. In addition, some of the models are trained from scratch, while others are pre-trained using ImageNet-1K weights [13]. More than 500 models are then evaluated on their respective datasets before having their accuracies averaged to get the final results. The authors of [14] utilize ViTs and their self-attention mechanism to achieve extremely accurate results when attempting to classify images of various crops and plant life. ViT's architecture allows it to focus on specific parts by enhancing or weakening predicted pixels within a feature map while ignoring the other perceptible aspects. However, these works do not address an MViT model that combines CNNs and transformers. As a result, experiments do not support how computationally expensive CNN layers are and if modifying them would provide a better outcome.

There are significant drawbacks of ViT, such as the exorbitant amount of training parameters as well as the high computational costs, which include the excessive number of images are required to train the model. Although transformers can be applied for image analysis, CNNs are generally the dominant model for most aspects of computer vision. CNN models are typically more compact and resource-efficient, while transformers are usually large: requiring a significant number of graphics processing units (GPUs) for training. However, transformers can get contextual understanding and global dependencies from images using self-attention, unlike CNNs, which typically use local operations that are restricted to small parts of images. This research demonstrates that a combination of CNNs and transformers can be optimized to extract both local and global contextual information for image classification. To achieve this objective, we propose modifications to the mobile vision transformer (MViT) [15] model, as it closely relates to a blend of CNN and vision transformers. Our findings demonstrate that higher classification performance can be achieved even by using lightweight convolutional variants, but only if they are used strategically in the entire architecture. Motivated by our previous work on ShuffleNet optimization [16], this research further solidifies our approach to building lightweight deep learning models without reducing accuracy.

Building upon these intricate models, our research opts for a simpler, less parameter-intensive architecture. This strategic simplicity aims to explore the potential of efficient training while maintaining accuracy in remote sensing image analysis. Our focus is on practicality and applicability, particularly in GIS scenarios, where resource efficiency is a key concern. As we progress to the specific methods in the following sections, our emphasis on the practical advantages of reduced complexity remains a cornerstone of our approach.

In summary, here are our main contributions in this paper:

- Model variants reduce the original MViT's training parameters by replacing expensive CNN modules with a combination of average pooling, depthwise separable convolution, and ShuffleNet blocks. Our models retain the benefits of CNN and transformers and use them to boost performance on geospatial datasets without some of the other unnecessary costs.
- The usage of convolution layers combined with the self-attention layers of transformers inside the MViT variants provides better performance across all geospatial datasets when compared with the standalone ViT model, which uses 95% more parameters than the MViT variants.
- We test our proposed architectures on four geospatial datasets; we generate several models and conform to testing standards as presented in the literature. The trained models are also made available on GitHub (https://github.com/rahulgomes19/gis-transformer (accessed on 23 September 2023)) for benchmarking and research purposes.

2. Previous Work

MViT has been applied for various purposes in fields where computer science is heavily applied, as demonstrated in [17]. In this research, the authors create a drone detection algorithm that has a lightweight MViTv1 backbone feature extractor and multi-scale attention feature fusion network (CA-PANet). The backbone allows the algorithm to fully extract local and global features due to its combined CNN and transformer architecture, which helps the network extract target location information with high accuracy while having a lesser number of parameters relative to other methods. Another application was outlined in [18]; the authors create an automatic diabetic retinopathy (DR) grading framework that has a ResNet101 (CNN) backbone and a custom MViT-Plus backbone to extract local and global information that is fused to assign DR grades (none, mild, moderate, or severe) for 2D fundus images. The custom MViT-Plus backbone was made during the research and was made using depthwise separable convolutions as well as MViT-Plus blocks to replace the transformer block with a lightweight transformer block. As a result, the model can obtain results quickly with lower costs relative to ViT models while achieving better results when compared to MViT and other models such as Resnext101 [19] and Se_resnet101 [20]. Although work applying the relatively lightweight but powerful MViT to different fields such as drones and healthcare is important, this research highlighted the fusion of ResNets with MobileNets, which increases model complexity. ResNets are, to some extent, costlier replacements to options like pooling, MobileNet [21], and ShuffleNet [22]. Additionally, the model was trained to predict only five classes compared to the extensive number of land cover classes in geospatial datasets with textual variation.

Two noteworthy contributions in this area are the studies by Huang et al. (2023) [23] and Zhang et al. [24]. Huang et al. introduced LTNet. This novel model fuses CNNs with transformers, focusing on efficient scene analysis in remote sensing. LTNet incorporates a multi-level group convolution module and the LightFormer block to effectively balance local and long-range dependencies. Its efficiency is marked by reduced parameters, leading to enhanced performance with shorter training durations, which is a critical aspect in rapid-deployment scenarios. Conversely, Zhang et al.'s work presented MLDANets. This model is designed for swift change detection and employs a unique attention aggregation and flexible sampling strategy. The MLDANets model excels at capturing the intricate details necessary for analyzing images captured at different times and showcases computational efficiency and strategic multi-level information coordination. Another work, proposed by Pengyuan et al. [25], developed a spatial-channel-feature-preserving ViT (SCViT) model that added a progressive aggregation (PA) strategy capable of combining neighboring tokens that overlap so that spatial information can be retained. The process was able to increase land cover classification accuracy. However, the number of parameters used for training was significantly large. For example, the SCViT-L variant utilized more than 40 million trainable parameters. With a vision transformer at its core, the model lacks the

benefits of convolutional layers, so these complex iterative additions may hold the key to exploring simpler channel shuffling operations from lightweight CNN models used in the past.

A list of notable works in the domain of reducing deep learning trainable parameters in geospatial applications is presented in Table 1. While these results are promising compared to the baseline architectures, the number of training parameters is still significantly higher compared to the modifications proposed in this research. Moreover, usage of pre-trained models freezes the majority of trainable parameters, rendering them unusable. In [12], an open-source benchmark to evaluate deep learning models for image classification in Earth observation (EO) was proposed. The authors conducted a comprehensive analysis of models from ten different deep learning architectures by comparing them to a variety of multi-class and multi-label image classification tasks from 22 datasets along with a related repository that can help build the foundation of guiding design principles for evaluating and documenting machine learning approaches in the different domains of EO. The process mostly used pre-trained models with a significantly large number of trainable parameters. In [26], Liu et al. proposed the RemoteCLIP model, which has rich semantics and aligned text embeddings for seamless downstream application. The researchers evaluated the model based on 16 datasets. The results indicated that RemoteCLIP consistently outperformed baseline foundation models across different model scales. In [27], the authors proposed a model based on MobileNetV3 for remote sensing image classification. They aimed to develop a model with fewer parameters so that it could be run on portable devices with reasonable accuracy. After comparing with other models by conducting experiments on different datasets, the results showed that the proposed model is not only lightweight but also has improved accuracy.

The authors of [28] proposed a model that enables and enhances a transfer learning model (Xception) for scene classification. After setting up the model, they evaluated it using datasets, including EuroSAT, UC-Merced, and AID. The results indicated that the proposed model outperformed state-of-the-art methods and had better accuracy and computational efficiency in each case. Chen et al. [29] proposed the RSCNet model, with the aim of improving the efficiency of remote sensing scene classification through lightweight neural networks. The model was evaluated by using the AID and UC-Merced datasets. The results showed that the RSCNet model has higher classification accuracy and faster processing speed on the two datasets, which provides the basic theory and key technical support for conducting fast classification of large amounts of remote sensing images. He et al. [30] proposed the BPKM model, with aim of building a lightweight network that can be applied on a mobile terminal or embedded device. After setting up the network, they evaluated the model by using the AID and UC-Merced datasets. The results showed that the BPKM model can make great classification of similar categories in aerial scene images, and the authors reduced the size of the network by about 24 times compared with popular networks.

Shi et al. [31] explored improving the classification performance while also avoiding drastically increasing the complexity of the model by using an AMB-CNN model for remote sensing image scene classification. They conducted experiments on the proposed model by using datasets, including AID, UC-Merced, NWPU45, and RSSCN7. Compared with some state-of-the-art methods, the number of parameters of the proposed method is only 5.6 million, and it has a great advantage in classification accuracy. In [32], Xu et al. proposed a novel scene classification model that integrates multi-source heterogeneous features, addressing the challenges of difficult distinctions of socio–economic attributes, visual–semantic discrepancies, intra-class differences, and high inter-class similarity. The experimental results indicated that the proposed model was better at solving the first three challenges. Since deep learning requires a huge number of training samples to ensure the optimal learning procedure, Lakshmi et al. [33] tried to address the issue of limited training samples in real-life situations for land use and land cover deep learning classification. The research focused on considering the fraction of multi-spectral data and evaluated the exemplary CNN architectures with different tuning variants along with

additional layers before classification. These changes increased the training characteristics on multi-spectral data.

Table 1. Notable work in geospatial image classification to reduce model complexity. Parameter size is in millions.

Paper	Datasets	Model	Parameter Size	Paper	Datasets	Model	Parameter Size
[34]	AID, UC-Merced, EuroSAT and WHURS19	MobileNetV2	2.5 m	[32]	UC-Merced, AID, and NWPU-RESISC45	LGDL	2.107 m
		DenseNet201	18.7 m			RSNet	2.997 m
		Xception	21.8 m			MobileNet-V2	3.5 m
		InceptionResNetV2	54.8 m			LGRIN	4.63 m
		ResNet152V2	58.8 m			SE-MDPMNet	5.17 m
		NASNetLarge	87.3 m			GoogLeNet	7 m
[29]	AID, UC-Merced	ShuffleNet v2	1.3 m			ResNet50	25.61 m
		SqueezeNet	1.3 m			Inception V3	45.37 m
		MobileNet v2	2.3 m			CaffeNet	60.97 m
		DenseNet-121	7 m			SPG-GAN	87.36 m
		ResNet-50	25.6 m			VGG-VD-16	138.36 m
		VGG-16	102 m			TSAN	381.67 m
[33]	UC-Merced, WHU-RS19, and AID	VGG16 Lightly Fine-tuned	9.47 m	[12]	EuroSAT, UC-Merced, WHU-RS19	EfficientNetBO	5.2 m
		VGG16 Heavily Fine-tuned	14.7 m			ResNet50	23.5 m
		ResNet152V2 Lightly Fine-tuned	53.97 m			DenseNet161	26.4 m
		ResNet152V2 Heavily Fine-tuned	58.32 m			ConvNeXt	28 m
[31]	UC-Merced, AID30 and NWPU45	AMB-CNN	5.6 m			SwinT	49.7 m
		HABFNet	6.2 m			AlexNet	57 m
		LCNN-BFF method	6.2 m			ResNet152	58.1 m
		Inceptionv3 + CapsNet	22 m			MLPMixer	59.8 m
		ResNet+WSPM-CRC	23 m			ViT	86.5 m
		Proposed method	28 m			VGG16	134.2 m
[26]	EuroSAT, AID, WHU-RS19	ResNet-50	38 m	[27]	NWPU, AID and UC-Merced	MBV3_SE_G	5.66 m
		Vit-Base-32	87 m			MBV3_G	7.65 m
		Vit-Large-14	304 m			ShuffleNet	8.69 m
[30]	AID-10, UC-Merced	Slim Network	2.5 m			MobilenetV2	13.37 m
		Slim Network (BPKM)	2.5 m			MBV3_SE	15.15 m
		SqueezeNet	4.8 m			MobilenetV3	20.92 m
		Plump Network	9 m			ResNet	83.15 m
		AlexNet	60 m	[28]	EuroSAT, UC-Merced, AID	EfficientNet	11 m
		VGG-VD-16	140 m			Xception	20 m
						ResNet-50	23 m

A prior work that attempted to modify MViT models to boost performance was also reported in [35]; the authors replaced a 3×3 convolution layer in the fusion block with a 1×1 convolution layer and replaced a 3×3 convolution layer in the global representation block with a depthwise convolution layer. Also, the model fuses the input features, combines local and global features, and increases the number of channels of the layers. Moreover, the model, which is called MViTv3, can outperform MViT variants such as MViTv1-XS and MViTv2-0.75 while maintaining a similar but slightly higher number of parameters. Another attempt to modify MViT for better performance and lower latency is demonstrated in [36]; the authors replaced the MHSA in the MViTv1_Block's transformer block with a separable self-attention method and did not use MViTv1_Block's skip connection and

fusion block. As a result, MViTv2 maintains a similar or smaller number of parameters and outperforms the MViTv1 model by about 0.9% on the ImageNet dataset. While these works can modify MViTs in a way that minimizes costs and boosts performance, the number of parameters used is still too high at around more than 1.25 million (MViTv3-XXS), and the increase in accuracy (0.9% by MViTv2) from the original MViT architecture is negligible. This is a drawback to implementing plug-and-play AI models in the real world with resource-constrained environments.

3. Materials and Methods

3.1. Vision Transformers

The architecture of ViT models can be seen in Figure 1. The ViT model reshapes the input image tensor into a sequence of flattened patches with dimensions $3P \times N$. The dimension $3P$ is obtained by multiplying the height and width of the pixels in the patches to produce P. Then, P is multiplied by the number of channels in the input image tensor to produce $3P$. The dimension N represents the number of patches. The sequence of flattened patches is then projected onto a fixed dimensional space with dimensions $d \times N$. The dimension d represents the size of the fixed dimensional space, while the dimension N represents the number of patches. Finally, a stack of L transformer blocks is used to learn long-range dependencies and global attention. The dimension L represents the number of transformer blocks utilized for this purpose.

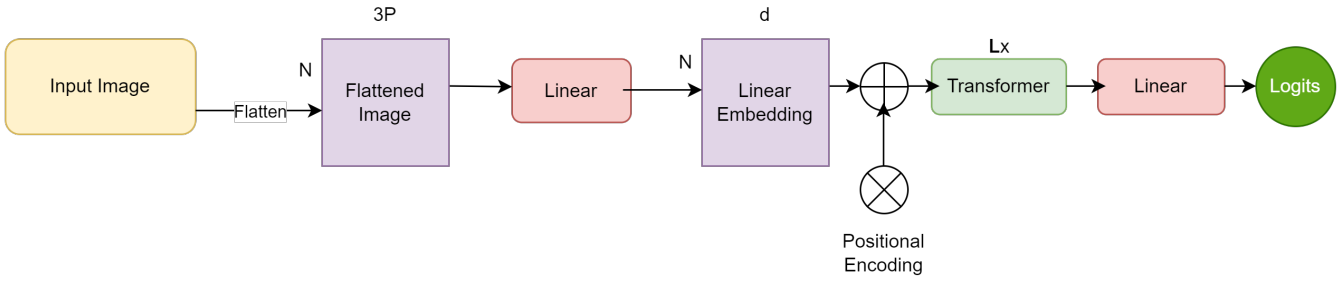


Figure 1. Architectureof the vision transformer model.

The mobile-friendly vision transformer or MViT model [15] contains MViT blocks that use convolution layers to generate local representations of the input tensor. Also, this block contains a transformer block with an MHSA mechanism that is used to generate global representations with spatial inductive biases that are fused with the local representations to preserve the benefits of transformers and CNNs.

The overall architecture of the model is displayed in Figure 2a. The architecture starts with a striped 3×3 convolutional layer. This layer is followed by four MobileNetv2 (MV2 blocks). The second and fourth blocks both use a stride of 2. These MV2 blocks are quite narrow and shallow, which means they do not significantly factor into the training parameter count. This is because their main responsibility is downsampling. These blocks are followed by an MViT block, which utilizes two transformer blocks that are represented by $L = 2$. Also, the spatial dimensions of the feature maps are often multiples of 2. As a result, the height and width dimensions of the feature maps, represented by h and w, respectively, are set to 2 at all spatial levels. Another MV2 block, with a stride of 2, is used before an MViT block with 4 transformer blocks. A dimension of 2 for the height and width spatial dimensions of the feature maps is applied. Then, an MV2 block with a stride of 2 is applied before the final MViT block, which has 3 transformer blocks and a dimension of 2 for the height and width spatial dimensions of the feature maps. Finally, a 1×1 convolutional layer and a global average pooling operation for spatial data are applied to produce the logits or the output of the last layer inside the model. Also, the output spatial dimensions of the model get smaller as the model gets closer to generating the logits. The output spatial dimensions used are 128×128, 64×64, 32×32, 16×16, 8×8, and 1×1.

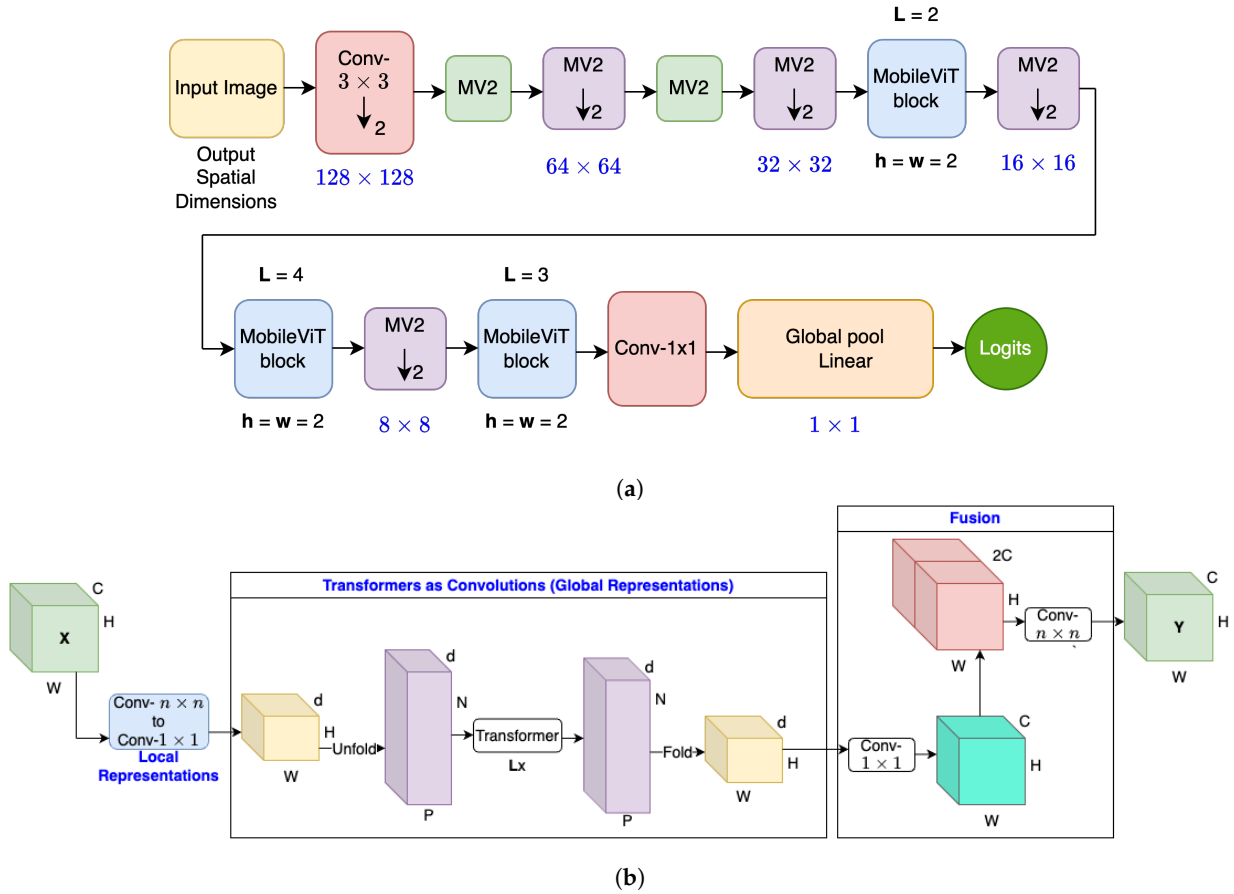


Figure 2. (**a**) Overall architecture of mobile-friendly vision transformer (MViT), and (**b**) shows the architecture of one MViT block in (**a**).

The architecture of one MViT block is displayed in Figure 2b. The MViT block receives an input tensor with the dimensions C, H, and W, which represent the channels, height, and width of the input tensor. Then, an $n \times n$ convolutional layer and a 1×1 convolutional layer are applied to the tensor to encode local spatial information and project the tensor to a high-dimensional space, respectively. As a result, this produces a modified tensor with the dimensions d, H, and W, where d is the size of the fixed dimensional space, while the dimensions H and W represent the height and width of the input tensor. After this, the modified tensor is unfolded into non-overlapping flattened patches with the dimensions d, N, and P, which represent the size of the fixed dimensional space, the number of patches, and the product of the height and width of the pixels in the patches, respectively. A stack of L transformer blocks, where L represents the number of transformer blocks, is applied to the non-overlapping flattened patches to generate a new sequence of non-overlapping flattened patches with the same dimensions d, N, and P. However, unlike ViT, MViT can remember the patch order and spatial order of pixels within each patch. The new sequence of flattened patches is then folded and projected to a high-dimensional space to make a tensor with dimensions d, H, and W, where d represents the size of the fixed dimensional space. Then, a 1×1 convolutional layer is applied to the tensor to project it to a low-dimensional space with the dimensions C, H, and W. The newly formed tensor is concatenated with the input tensor to produce a new tensor with dimensions $2C$, H, and W. Finally, an $n \times n$ convolutional layer is applied to fuse the concatenated features and generate the output tensor with dimensions C, H, and W. MViT is a powerful model because it can achieve high performance with a reduced number of parameters relative to heavyweight ViTs and CNNs. It leverages the inherent advantages of both architectures.

While MViT can utilize the dependable aspects of both CNNs and transformers, there are still potential improvements that can further optimize MViT to boost its performance on geospatial datasets and reduce its training parameter count. In this research, we incrementally explore these parameters to further optimize this architecture by strategically replacing blocks that are used to extract higher-order features.

3.2. Proposed Lightweight Transformer Modifications

The first variant tested in our research is called MViT-Depth. To create this variant, a 1×1 convolution layer in the local representations section of the MViT block was removed. An average pooling layer that features a pool_size of 2×2 was added. A 1×1 upsampling layer was introduced in between the folded global feature map; this layer contains the global features extracted by the transformer block and the concatenate layer, which combines the folded global features with the local features extracted using the $N \times N$ convolution layer and the average pooling layer. We further modified the MViT block inside the model by replacing the first $N \times N$ convolution layer, which takes in the input tensor to generate local representations, with an $N \times N$ depthwise separable convolution layer. When compared to the ViT model, MViT-Depth has 20,509,607 fewer parameters than ViT on average (94.62% decrease). When compared to the MViT model, MViT-Depth had exactly 145,200 fewer parameters than MViT across all four datasets (11.06% decrease).

The architecture of the MViT-Depth block is shown in Figure 3. This block receives an input tensor with the dimensions C, H, and W. Then, the model modifies the local representations section by using a $n \times n$ separable convolutional layer instead of a regular convolutional layer as well as an average pooling layer with a pool size of 2×2 to encode local spatial information and project the tensor to a high-dimensional space, respectively. This produces a modified tensor with the dimensions d, H, and W that passes through the transformer layers as explained in the previous MViT-Avg section. The new sequence of flattened patches is folded and projected to a high-dimensional space to make a tensor with dimensions d, H, and W. Then, a 1×1 convolutional layer is applied to the tensor to project it to a low-dimensional space with dimensions C, H, and W. The tensor is then fed into an upsampling layer that helps balance the class labels followed by concatenation with the input tensor to produce a new tensor with dimensions $2C$, H, and W. Finally, an $n \times n$ convolutional layer is applied to fuse the concatenated features and generate an output tensor with the same dimensions C, H, and W.

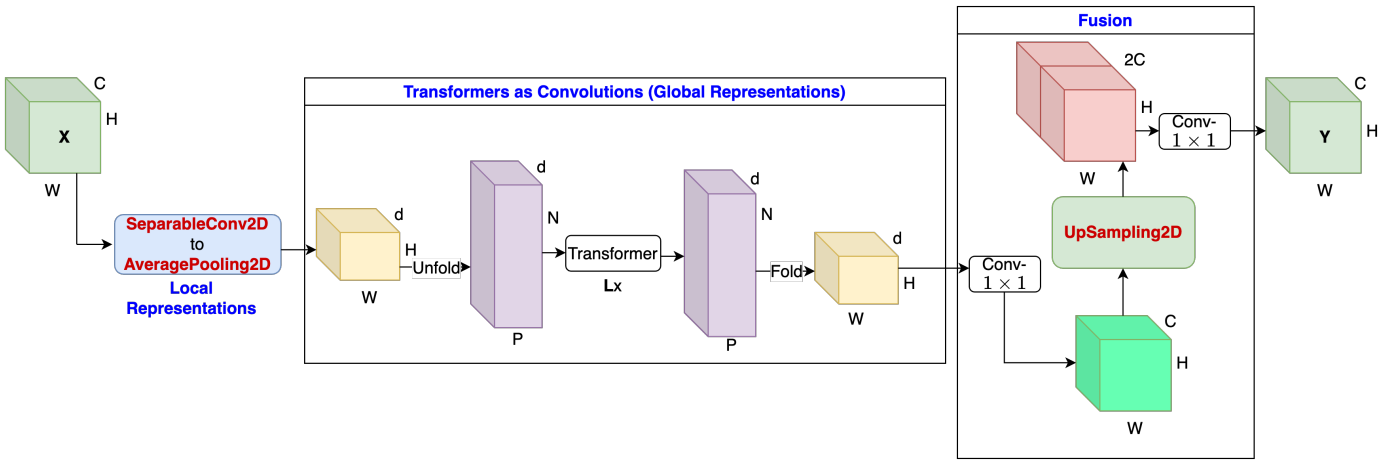


Figure 3. Architecture of the proposed MViT-Depth transformer block, with changes in red.

In the MViT-combined variant, a custom ShuffleNet block replaces the final MViT block, as shown in Figure 2a, to create a new model, as shown in Figure 4a. This ShuffleNet block sets the default kernel size of its layers to be 1×1, and it sets the default number of filters to be 320. The custom ShuffleNet block can be seen in Figure 4b. This block consists of two pathways of layers that are fused to generate the output tensor. Both pathways

receive an input tensor with dimensions C, H, and W. The first pathway of layers is called A, and it starts by feeding the input tensor into a 3×3 depthwise convolution layer. Then, the output from that initial layer is processed by a batch normalization layer as well as a 1×1 convolution layer. Finally, the pathway ends with the output being processed by another batch normalization and a ReLU layer. The second pathway of layers is labeled with B, and it starts by feeding the input tensor into a 1×1 convolution layer. The output from the initial layer is processed by a batch normalization layer and a ReLU Layer. Then, the new output is processed by a 3×3 depthwise convolution layer as well as another 1×1 convolution layer. The activation map is then processed by another batch normalization layer and another ReLU layer. To fuse the two pathways, a concatenate layer is utilized. The fusion results in an output tensor with dimensions C, H, and W. After these changes were made, the proposed model had 20,598,679 fewer parameters than ViT on average (a 95.03% decrease). Also, it had exactly 234,272 fewer parameters than MViT across all datasets (a 17.85% decrease).

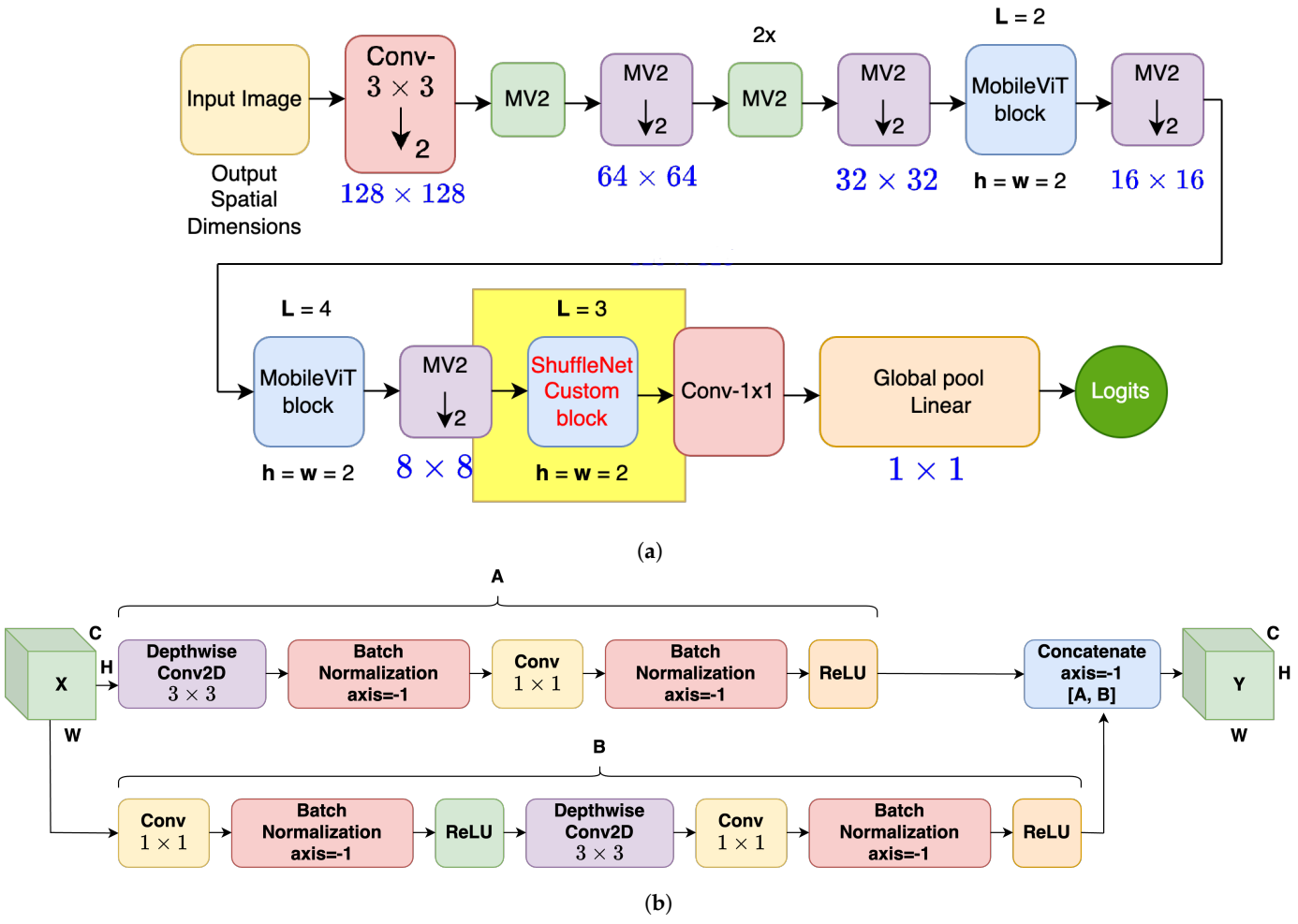


Figure 4. (**a**) Architecture of the entire MViT block and (**b**) architecture of the ShuffleNet block that replaces the final MViT block to create the final MViT-combined variant.

For the SWIN model and its variants, the standard SWIN architecture from Liu et al. [37] was used as a reference. Detailed dimensions and configurations play a crucial role in achieving balance between model complexity and accuracy while ensuring the parameters remain lightweight. In the SWIN-Regular model, the PatchExtract layer processes input images into 2×2 patches, reflecting the defined patch size. These patches are embedded with positional information in the PatchEmbedding layer and then processed through SwinTransformer layers, which are characterized by a window size of 2 and a shift size of 1. This design reduces computational complexity compared to global attention mechanisms.

The embedding dimension is set at 64, with the number of multi-layer perceptron (MLP) nodes at 256.

Two different SWIN variants were also explored. In the SWIN-ConvBlock variant, the introduction of a convolutional block before feeding into the SWIN model enhances the feature space and is beneficial for performance improvement. This block comprises two sets of layers: a convolutional layer, a batch normalization layer, and an activation layer with a ReLU function. By replacing the third patch extract layer in the SWIN model with this convolutional block, the model achieves an increased feature space and a reduction in the number of parameters. Secondly, the SWIN-ShuffleNet variant features a ShuffleNet block, which is designed to create a more efficient architecture through a channel shuffle operation. This block consists of separable convolutional layers, group normalization layers, and activation layers (ReLU function). By replacing the third patch extract layer with the ShuffleNet block, it is expected that SWIN-ShuffleNet maintains or boosts accuracy while reducing the parameter count, capitalizing on the efficiency of the channel shuffle operation.

The total number of training parameters for each model architecture trained on the four GIS datasets AID, EuroSAT, UC-Merced, and WHU-RS19 is summarized in Table 2. None of the hyperparameters except the number of classes and train–test splits used by the model influence the number of training parameters, so there are only four entries for each model. An observable trend in the table is that the number of training model parameters for each architecture increases for datasets that have a larger number of image labels. This shows that each model's output layer, which maps to a differing number of image classes for each dataset, slightly influences the number of training parameters of each model.

Table 2. Comparison of the number of trainable parameters for the transformer variants.

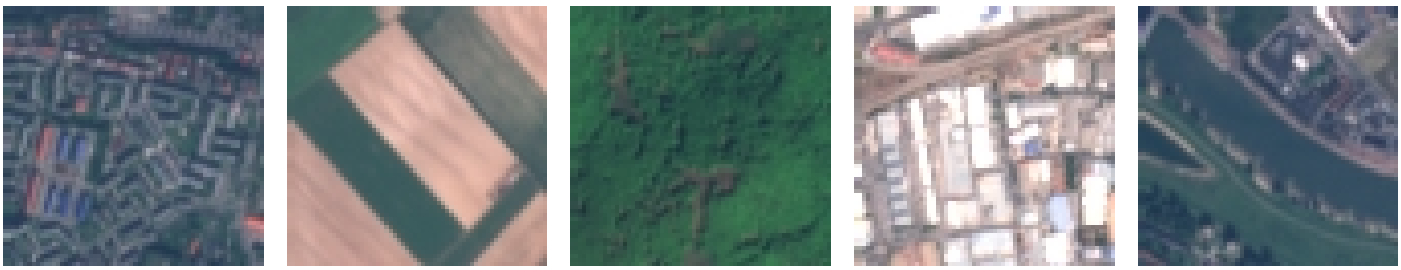
Dataset	AID Parameters	EuroSAT Parameters	UC-Merced Parameters	WHU-RS19 Parameters
ViT	21,687,269	21,665,744	21,678,044	21,675,994
MViT	1,315,646	1,308,905	1,312,757	1,312,115
MViT-DepthConv	1,170,446	1,163,705	1,167,557	1,166,915
MViT-Combined	1,016,846	1,010,105	1,013,957	1,013,315
SWIN-Reg	155,022	152,313	153,861	153,603
SWIN-ConvBlock	172,846	170,137	171,685	171,427
SWIN-Shuffle	164,073	161,364	162,912	162,654

3.3. Datasets

Models were trained on four multi-class classification GIS datasets: AID [38], EuroSAT [39,40], UC-Merced [41], and WHU-RS19 [42,43]. These datasets feature diverse image sizes, spatial resolutions, image types, image formats, and labels. These statistics and data are displayed in Table 3. The labels and their sequence numbers are provided in Table 4. Due to the variety of datasets utilized for this project, we ensured that the better performance of our custom MViT variants was consistent across a wide range of labeled images. The AID dataset is a large aerial image dataset that was formed by collecting images from Google Earth imagery. These Google Earth images are also post-processed with RGB renderings extracted from the original optical aerial images. The images of the AID dataset are labeled with 30 aerial scene class labels, which is the most out of all the datasets. Also, there are 10,000 JPG images inside AID with a size of 600 × 600 and an image resolution ranging from 0.5 m to 8 m. Figure 5a–e display five sample images from the AID dataset. The EuroSAT dataset is a large-scale satellite multispectral image dataset that was collated by using Sentinel-2 satellite images that are accessible from the open-source Earth observation program Copernicus [44]. Also, this dataset is unique because the images are multispectral and cover 13 spectral bands that are in the short infrared, near-infrared, and visible parts of the spectrum. Our experiments used a smaller version of EuroSAT, and the images of the dataset are labeled with nine class labels, which is one less than the original and the least out of the four datasets. In addition, EuroSAT is the largest dataset,

as it contains 24,500 JPG images that have a size of 64 pixels by 64 pixels and an image resolution of 10 m. Figure 5f–j display sample images from the EuroSAT dataset.

(**a**) AID—Residential (**b**) Farmland (**c**) Forest (**d**) Industrial (**e**) River



(**f**) Euro—Residential (**g**) Annual Crop (**h**) Forest (**i**) Industrial (**j**) River

(**k**) UC-Merced—Residential (**l**) Agriculture (**m**) Forest (**n**) Overpass (**o**) River

(**p**) WHU—Residentiall (**q**) Farmland (**r**) Forest (**s**) Industrial (**t**) River

Figure 5. Sample images from the four datasets used in this study.

Table 3. Dataset features used in this study.

Name	# of Images	Image Size	Spatial Resolution	Image Type	Image Format	# of Labels
AID	10,000	600 × 600	0.5–8 m	Aerial RGB	JPG	30
EuroSAT	24,500	64 × 64	10 m	Multispectral	JPG	9
UC-Merced	2100	256 × 256	0.3 m	Aerial RGB	TIF	21
WHU-RS19	1005	600 × 600	≤0.5 m	Aerial RGB	JPG	19

Table 4. Dataset labels used in this study.

Class #	Labels			
	AID	**EuroSAT**	**UC-Merced**	**WHU RS-19**
0	Airport	Annual Crop	Agricultural	Airport
1	Bare Land	Forest	Airplane	Beach
2	Baseball Field	Herbaceous Vegetation	Baseball Diamond	Bridge
3	Beach	Highway	Beach	Commercial
4	Bridge	Industrial	Buildings	Desert
5	Center	Pasture	Chaparral	Farmland
6	Church	Residential	Dense Residential	Forest
7	Commercial	River	Forest	Industrial
8	Dense Residential	Sea Lake	Freeway	Meadow
9	Desert		Golf Course	Mountain
10	Farmland		Harbor	Park
11	Forest		Intersection	Parking
12	Industrial		Medium Residential	Pond
13	Meadow		Mobile Home Park	Port
14	Medium Residential		Overpass	Residential
15	Mountain		Parking Lot	River
16	Park		River	Viaduct
17	Parking		Runway	Football Field
18	Playground		Sparse Residential	Railway Station
19	Pond		Storage Tanks	
20	Port		Tennis Court	
21	Railway Station			
22	Resort			
23	River			
24	School			
25	Sparse Residential			
26	Square			
27	Stadium			
28	Storage Tanks			
29	Viaduct			

The UC-Merced dataset [41] is a large aerial image dataset that was formed by extracting a diverse range of smaller images from large images that were collated in the USGS National Map Urban Area Imagery Collection. The sizes of these smaller images are 256 × 256, and the images come from different urban areas around the United States of America. The dataset's images feature the smallest spatial resolution of the four datasets at 0.3 m, and they are the only dataset with TIF images. In addition, the dataset features images that belong to 21 classes, and there are 100 images per image class, which leads to a total of 2100 images in the dataset. Figure 5k–o display five sample images from the UC-Merced dataset.

The WHU-RS19 dataset is a large-scale aerial image dataset that consists of satellite images that were collected from Google Earth. The dataset is similar to AID in that both datasets have the same image sizes (600 × 600), and they both originate from Google Earth imagery. In addition, WHU-RS19 is the smallest dataset out of the four datasets, with 1005 JPG images. Also, the images in the dataset range have a spatial resolution of up to 0.5 m as well as a diverse range of orientations and illuminations. Also, the images of the dataset belong to 19 image classes. Figure 5p–t display sample images from the WHU-RS19 dataset.

4. Results

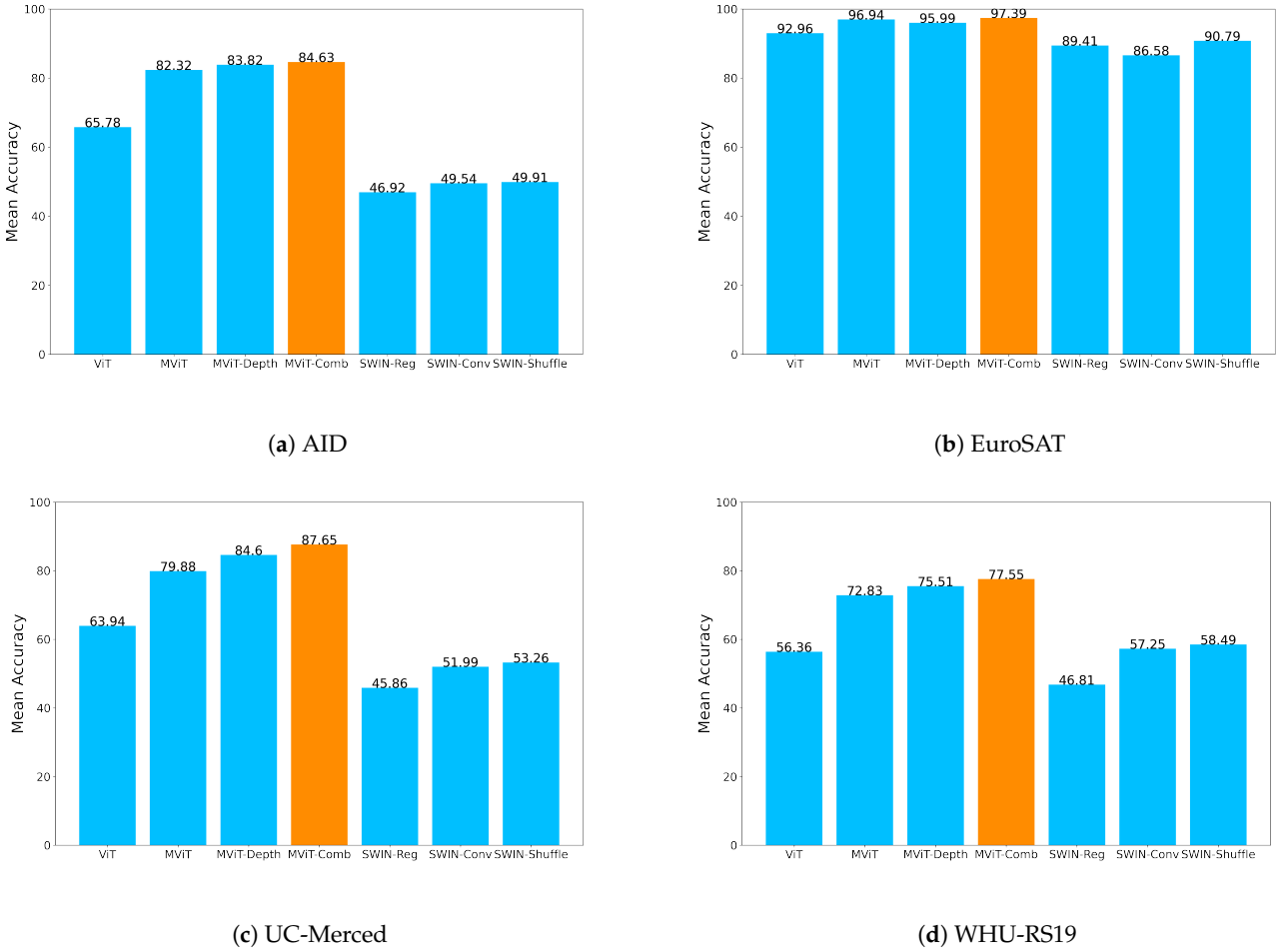
For the training and evaluation stage of the completed model architectures, a group of four experiments were devised, with each experiment corresponding to one of the four datasets that we planned to train and evaluate the models on. This was followed by the application of each representative architecture, which included the two benchmarks ViT and MViT along with the three MViT variants, to all of the datasets. Multiple train–test splits for each model architecture were also introduced to ascertain if the models could achieve a high level of performance with lesser amounts of training data. The train–test splits used were 20–80%, 40–60%, 50–50%, and 60–40%. Hence, each experimental group corresponding to one of the four datasets had 28 models used for training, bringing the number of trained models to 112.

All experiments were conducted at the Blugold Center for High-Performance Computing using an NVIDIA Tesla V100 GPU (Nvidia, Santa Clara, CA, USA) with 32 GB memory and an AMD EPYC CPU at 2.35 GHz. Data augmentation for the training images started with resizing all of the images to 72×72 followed by randomly flipping those images along a horizontal axis. The images were randomly rotated before being randomly zoomed in on during the training process. The hyperparameters were configured to control the training process. The number of epochs was kept constant at 500, and the batch size was kept constant at 64. Also, all of the models used an Adam optimizer [45] to help change the weights and loss rates. For this optimizer, the learning rate was set to 0.001, which allowed for slow and precise learning. The weight decay was set to 0.0001 to regularize the neural network by adding a penalty to the loss function. The only hyperparameter that we varied across the models was the number of classes or labels because each dataset has a different number of class labels. This hyperparameter played a role in influencing the total number of training parameters because it modified the size of the output dense layer in the ViT models. The training accuracies for all model variants are shown in Table A1 in Appendix A.

The testing accuracy serves as a vital metric for evaluating the model's performance on unseen data by providing insights into its generalization capabilities. A high testing accuracy indicates that the model has successfully learned meaningful patterns from the training data and can effectively make predictions on new, unseen samples. Moreover, monitoring testing accuracy helps with detecting overfitting, wherein the model performs well on the training data but fails to generalize to new data. By continuously assessing and improving testing accuracy throughout the training process, developers can fine-tune their models, enhance their predictive capabilities, and build robust solutions that perform reliably in real-world scenarios. The testing accuracy results can be seen in Table A1. The mean testing accuracy of all MViT-Combined models is shown in Figure 6. These models were able to converge to an optimal value at the end of 500 epochs, showcasing their robustness. Throughout the research setup, we monitored the testing accuracy from the splits and observed that all the proposed MViT variants seemed to outperform traditional versions.

For the AID experiment group, the MViT variants outperformed the ViT model across all of the splits. A consistent trend was noted among all architectures, where the 60–40 split was typically the highest-performing model compared to other splits, with the MViT-Combined model achieving a test accuracy of 89.42%, which outperformed ViT's best validation accuracy by 17.92% and MViT's best test accuracy by 2.84%. The MViT-Combined model, with only 1,016,846 parameters, extended this performance, achieving the highest accuracy among MViT's variants while using significantly fewer parameters than the ViT model, which stands at 21,687,269 parameters. The class-based performance for the 60–40 AID testing dataset is shown in Table 5. The SWIN-Reg model and its variants, while not outperforming the MViT series, still offered a different approach to model architecture, emphasizing regularization and compactness with only 155,022 parameters for SWIN-Reg itself. SWIN-ShuffleNet, another variant, showed an interesting combination of SWIN's approach with ShuffleNet's efficiency. The per-class validation accuracy of the 60–40 split on the AID dataset showed significant improvements. For example, the accuracy for Airports

(class 0) jumped from 40% to 86.81% for MViT-Combined. Similar trends were observed for other classes, indicating that the MViT variants can provide substantial accuracy benefits over the baseline ViT model. However, challenges persisted in classifying Squares (class 26), which could be due to the presence of various sub-classes within the category. Loss functions reported a consistent trend for both the training and testing data, as shown in Figure 7.



(**a**) AID

(**b**) EuroSAT

(**c**) UC-Merced

(**d**) WHU-RS19

Figure 6. Mean testing accuracy graphs of the best models evaluated in this study.

The findings from the EuroSAT experiment group demonstrated that the highest accuracy was uniformly distributed across all types of splits, as indicated in Table 6. The MViT-Combined model demonstrated strong performance on the EuroSAT dataset, achieving a test accuracy of 97.8% on the 40–60 split with just 1,010,105 parameters. Unlike with the AID dataset, where shifts from ViT to MViT variants manifested noticeable improvements, the EuroSAT dataset saw less significant increases in class accuracy upon implementing advanced models. Despite the modest relative value changes, the absolute values of the MViT variants were notably high. This is particularly evident in the case of Residential (class 6), where all MViT variants achieved 100% class accuracy. Beyond Residential, several other classes also benefited from the advanced architecture. For example, the accuracies of Highways (Class 3) and Industrial (Class 4) show significant increases from 82.4% and 93.27% in ViT to 95.67% and 95.73%, respectively, in MViT-Combined models. The SWIN-Reg model's accuracy of 89.23% on the 40–60 split is notable in terms of parameter efficiency. The SWIN-ShuffleNet variant further underscores the efficacy of combining architectures, achieving 92.07% on the 40–60 split. The advancements in EuroSAT's class accuracies warrant further analysis to understand the underlying factors contributing to these results. Training loss outcomes can be seen in Figure 8.

Table 5. Accuracy of 60–40 split models on the AID dataset. The proposed model shows the highest overall accuracy (in red).

Class Number	ViT	MViT	MViT-Depth	MViT-Combined	SWIN-Reg	SWIN-Conv Block	SWIN-ShuffleNet
0	40.00%	85.56%	87.22%	86.81%	41.67%	47.92%	37.50%
1	84.52%	87.74%	73.55%	84.68%	90.32%	74.19%	67.74%
2	76.37%	92.72%	94.55%	95.45%	67.05%	81.82%	64.77%
3	87.50%	95.00%	97.00%	95.00%	72.50%	85.63%	90.63%
4	70.00%	89.44%	85.56%	88.89%	61.81%	63.19%	63.89%
5	71.54%	80.00%	69.23%	89.42%	42.31%	44.23%	38.46%
6	60.83%	95.00%	93.33%	93.75%	39.58%	47.92%	34.38%
7	86.86%	72.57%	89.71%	77.86%	65.00%	75.00%	31.43%
8	80.98%	95.12%	81.46%	88.41%	61.59%	76.83%	54.88%
9	84.67%	82.00%	96.00%	87.50%	72.50%	80.00%	89.17%
10	64.86%	89.73%	89.73%	92.57%	35.14%	45.27%	43.92%
11	76.00%	97.60%	97.60%	99.00%	76.00%	85.00%	85.00%
12	65.64%	78.46%	86.15%	91.67%	30.77%	32.69%	39.10%
13	80.71%	95.71%	95.00%	98.21%	88.39%	97.32%	83.93%
14	76.55%	91.72%	92.41%	93.10%	40.52%	37.07%	26.72%
15	54.12%	85.29%	97.65%	94.12%	57.35%	44.12%	41.18%
16	62.86%	77.71%	82.86%	79.29%	36.43%	51.43%	52.86%
17	81.54%	98.46%	96.41%	97.44%	80.77%	72.44%	62.18%
18	81.62%	92.43%	88.11%	89.19%	72.30%	76.35%	66.89%
19	83.33%	79.05%	87.62%	94.05%	73.81%	77.38%	73.81%
20	75.79%	97.37%	95.79%	95.39%	80.26%	74.34%	74.34%
21	36.92%	94.62%	74.62%	87.50%	47.12%	30.77%	39.42%
22	62.07%	68.97%	68.97%	77.59%	31.03%	27.59%	25.00%
23	45.37%	87.80%	90.73%	92.07%	34.76%	32.93%	45.12%
24	42.00%	57.33%	78.00%	70.83%	14.17%	20.83%	11.67%
25	84.67%	96.67%	96.00%	98.33%	59.17%	76.67%	70.83%
26	38.79%	44.24%	63.64%	66.67%	18.18%	12.12%	18.94%
27	86.21%	86.90%	95.17%	93.97%	62.93%	72.41%	65.52%
28	63.89%	84.44%	82.22%	90.97%	39.58%	36.11%	42.36%
29	53.81%	93.81%	91.90%	92.26%	22.02%	18.45%	21.43%
Mean Acc	68.61%	85.93%	87.31%	89.40%	53.83%	56.60%	52.10%
Overall Acc	68.62%	85.97%	87.50%	89.42%	53.70%	56.35%	52.15%
Kappa	0.6750	0.8548	0.8705	0.8905	0.5206	0.5479	0.5044

The results for the UC-Merced experiment group show that the 50–50 split model typically performed exceptionally well, with the MViT-Combined variant achieving the highest test accuracy of 92.1% on this split. In this configuration, the MViT-Depth variant reached an accuracy of 91.71%, significantly outperforming ViT's best split by over 16% and outdoing the benchmark MViT's best split by nearly 4.09%.

All MViT variants surpassed the ViT model across each evaluated split. The impressive performance of these models, particularly MViT-Combined, suggests an optimal balance between the transformer architecture's capabilities and the efficiency needed for mobile deployment, positioning them as a scalable solution for high-resolution satellite imagery analysis. The class-based performance for the 50–50 test split is detailed in Table 7. The MViT architecture contributed to substantial improvements across multiple classes, with notable advances in distinguishing different types of urban structures. For example, in Classes 2, 9, 12, and 19, we noticed a significant improvement for MViT-Combined compared to the baseline MViT. Baseball Diamond (Class 2) saw scores increase from 88% to 96%, while Golf Course (Class 9) scores increased from 80% to 100%. Medium Residential

(Class 12) increased from 48% to 74%, and Storage Tanks (Class 19) increased from 50% to 70%. Although the SWIN-Reg and SWIN-ShuffleNet models did not achieve the top accuracies, with SWIN-ShuffleNet reaching 58% on the 50–50 split, their designs highlight the trade-offs between accuracy and compactness of the model. Training loss outcomes can be seen in Figure 9.

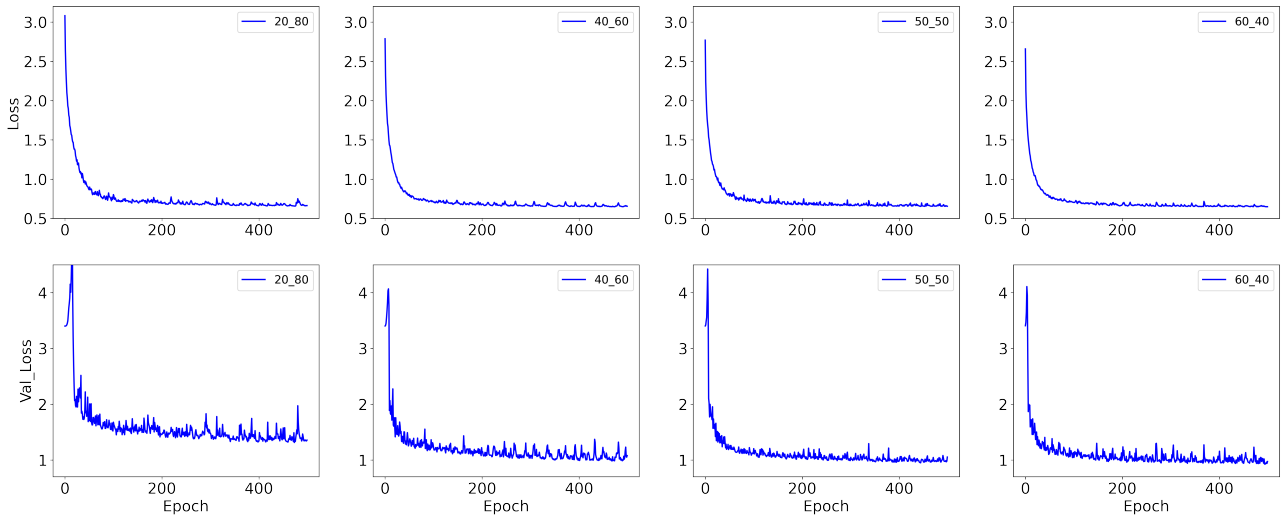


Figure 7. Training (**top row**) and testing (**bottom row**) loss for AID datasets across multiple splits.

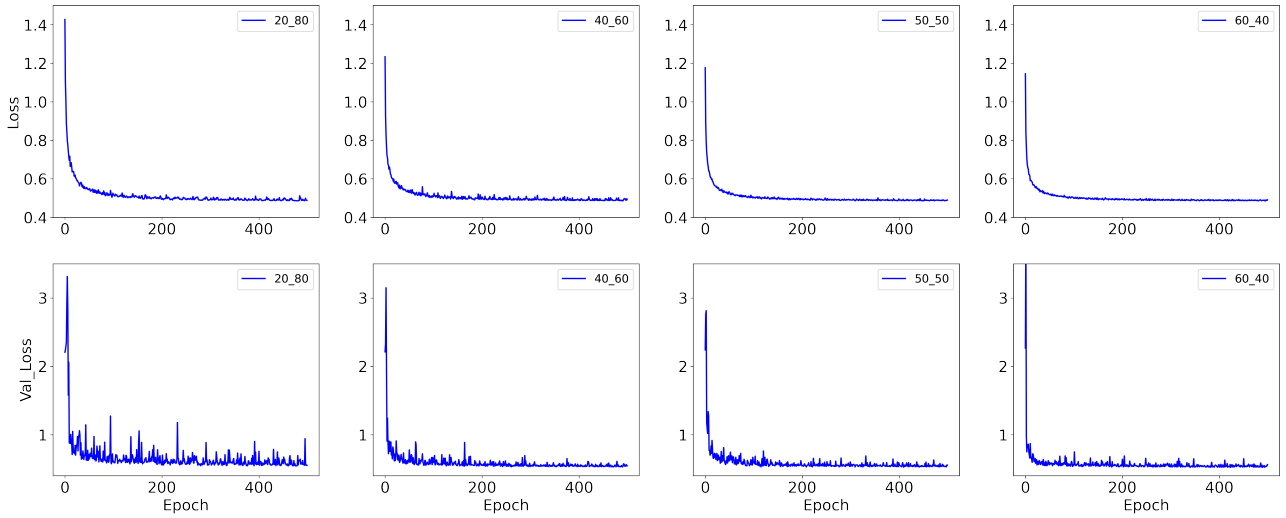


Figure 8. Training (**top row**) and testing (**bottom row**) losses for EuroSat datasets across multiple splits.

The results for the WHU-RS19 experiment group reveal that the MViT-Combined model on the 60–40 split was the highest-performing model, with a test accuracy of 86.32%. It outperformed the other splits using the same architecture. Moreover, these results were achieved with significant improvements in class-specific accuracies, with the highest overall accuracy and Kappa.

The class-based performance for the 60–40 WHU-RS19 dataset is shown in Table 8. We also note that, like with AID and UC-Merced, using the MViT variants resulted in a significant accuracy increase for several classes, with MViT-Combined showing exceptional performance. In Classes 2 (Bridge), 5 (Farmland), 9 (Mountain), and 10 (Park), a comparison of the results between MViT and MViT-Combined shows increases from 80.77% to 90.48%, 72% to 85%, 56% to 90%, and 52% to 75%, respectively. The training and testing losses can be seen in Figure 10.

Table 6. Accuracy of 40–60 split models on the EuroSAT dataset. The proposed model shows the highest overall accuracy (in red).

Class Number	ViT	MViT	MViT-Depth	MViT-Combined	SWIN-Reg	SWIN-Conv Block	SWIN-ShuffleNet
0	90.78%	92.78%	97.83%	97.06%	92.11%	89.11%	92.72%
1	98.00%	99.17%	99.56%	99.33%	94.28%	80.61%	98.00%
2	92.06%	98.33%	96.28%	97.50%	92.17%	77.33%	90.39%
3	82.40%	96.13%	97.33%	95.67%	74.40%	72.27%	83.27%
4	93.27%	89.40%	96.33%	95.73%	92.60%	88.93%	94.93%
5	91.83%	92.83%	95.42%	98.17%	90.75%	89.67%	90.50%
6	98.94%	100.00%	99.94%	99.89%	94.28%	99.06%	97.72%
7	88.27%	96.80%	96.47%	97.07%	73.60%	79.93%	78.87%
8	95.67%	99.50%	98.94%	99.11%	98.89%	92.72%	98.50%
Mean Acc	92.36%	96.10%	97.57%	97.72%	89.23%	85.51%	91.66%
Overall Acc	92.65%	96.36%	97.71%	97.80%	89.72%	85.66%	92.07%
Kappa	0.9171	0.959	0.9742	0.9752	0.8841	0.8383	0.9106

Table 7. Accuracy of 50–50 split models on the UC-Merced dataset. The proposed model shows the highest overall accuracy (in red).

Class Number	ViT	MViT	MViT-Depth	MViT-Combined	SWIN-Reg	SWIN-Conv Block	SWIN-ShuffleNet
0	90.00%	98.00%	98.00%	98.00%	76.00%	84.00%	78.00%
1	68.00%	92.00%	92.00%	96.00%	54.00%	50.00%	78.00%
2	84.00%	88.00%	92.00%	96.00%	66.00%	82.00%	68.00%
3	100.00%	94.00%	98.00%	98.00%	90.00%	100.00%	98.00%
4	56.00%	72.00%	72.00%	86.00%	24.00%	38.00%	32.00%
5	100.00%	100.00%	100.00%	100.00%	92.00%	98.00%	100.00%
6	38.00%	92.00%	84.00%	80.00%	50.00%	54.00%	40.00%
7	80.00%	100.00%	92.00%	92.00%	56.00%	82.00%	82.00%
8	74.00%	88.00%	94.00%	96.00%	32.00%	52.00%	48.00%
9	50.00%	80.00%	92.00%	100.00%	60.00%	86.00%	66.00%
10	94.00%	100.00%	98.00%	100.00%	90.00%	96.00%	98.00%
11	74.00%	88.00%	86.00%	86.00%	20.00%	26.00%	24.00%
12	56.00%	48.00%	84.00%	74.00%	10.00%	24.00%	12.00%
13	80.00%	92.00%	98.00%	96.00%	76.00%	86.00%	74.00%
14	48.00%	90.00%	90.00%	96.00%	40.00%	16.00%	20.00%
15	74.00%	98.00%	100.00%	100.00%	44.00%	70.00%	50.00%
16	80.00%	94.00%	98.00%	94.00%	52.00%	44.00%	60.00%
17	78.00%	100.00%	100.00%	100.00%	38.00%	92.00%	88.00%
18	22.00%	92.00%	94.00%	94.00%	16.00%	22.00%	56.00%
19	26.00%	50.00%	78.00%	70.00%	40.00%	20.00%	22.00%
20	36.00%	74.00%	86.00%	82.00%	30.00%	38.00%	24.00%
Mean Acc	67.05%	87.14%	91.71%	92.10%	50.29%	60.00%	58.00%
Overall Acc	67.05%	87.14%	91.71%	92.10%	50.29%	60.00%	58.00%
Kappa	0.6540	0.8650	0.913	0.917	0.4780	0.5800	0.5590

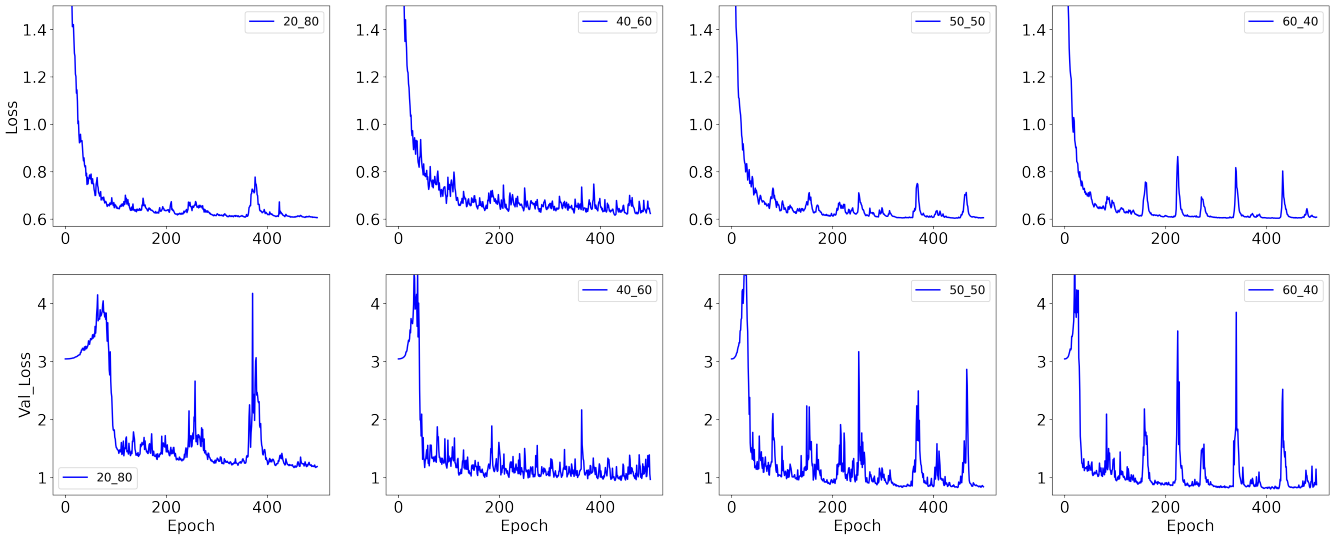


Figure 9. Training (**top row**) and testing (**bottom row**) losses for UC-Merced dataset across multiple splits.

Table 8. Accuracies of 60–40 split models on the WHU-RS19 dataset. The proposed model shows the highest overall accuracy (in red).

Class Number	ViT	MViT	MViT-Depth	MViT-Combined	SWIN-Reg	SWIN-Conv Block	SWIN-ShuffleNet
0	28.57%	67.86%	60.71%	72.73%	40.91%	45.45%	59.09%
1	100.00%	96.00%	96.00%	95.00%	85.00%	95.00%	95.00%
2	65.38%	80.77%	80.77%	90.48%	52.38%	66.67%	90.48%
3	32.14%	85.71%	60.71%	86.96%	34.78%	47.83%	30.43%
4	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
5	60.00%	72.00%	84.00%	85.00%	80.00%	70.00%	65.00%
6	88.46%	100.00%	88.46%	90.48%	76.19%	95.24%	80.95%
7	50.00%	80.77%	92.31%	76.19%	23.81%	33.33%	33.33%
8	93.55%	77.42%	77.42%	79.17%	75.00%	100.00%	91.67%
9	44.00%	56.00%	96.00%	90.00%	30.00%	25.00%	45.00%
10	68.00%	52.00%	88.00%	75.00%	55.00%	75.00%	60.00%
11	28.00%	92.00%	96.00%	100.00%	35.00%	40.00%	45.00%
12	66.67%	92.59%	100.00%	95.45%	54.55%	59.09%	63.64%
13	37.04%	88.89%	77.78%	76.19%	28.57%	57.14%	42.86%
14	70.37%	74.07%	25.93%	86.36%	31.82%	27.27%	40.91%
15	64.29%	71.43%	82.14%	81.82%	72.73%	59.09%	50.00%
16	24.14%	79.31%	62.07%	69.57%	17.39%	34.78%	39.13%
17	92.00%	88.00%	92.00%	100.00%	40.00%	80.00%	95.00%
18	48.00%	92.00%	92.00%	95.00%	40.00%	30.00%	40.00%
Mean Acc	61.08%	81.41%	80.71%	86.60%	51.22%	60.05%	61.45%
Overall Acc	60.83%	81.31%	79.53%	86.32%	51.00%	59.95%	61.19%
Kappa	0.5865	0.8027	0.7840	0.8556	0.4825	0.5769	0.5904

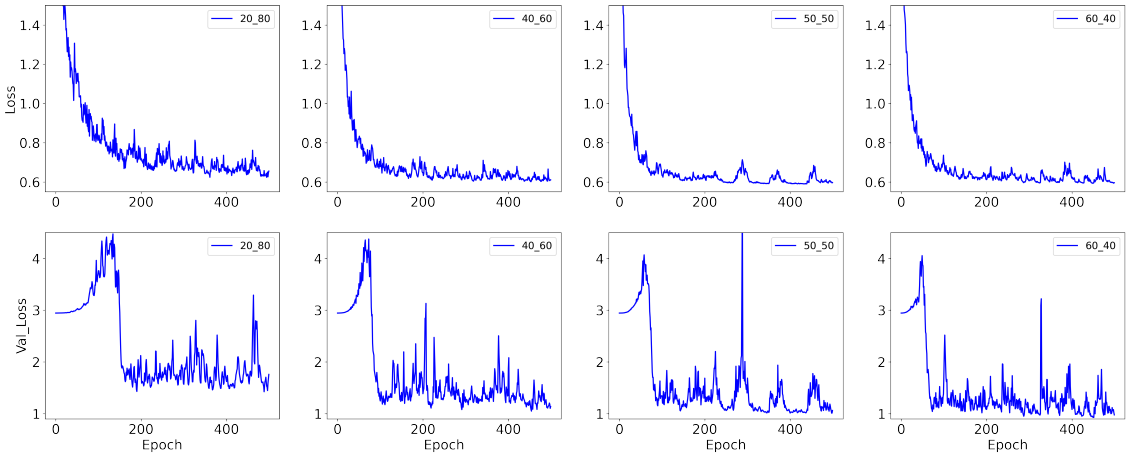


Figure 10. Training (**top row**) and testing (**bottom row**) losses on WHU-RS19 dataset across multiple splits.

5. Discussion

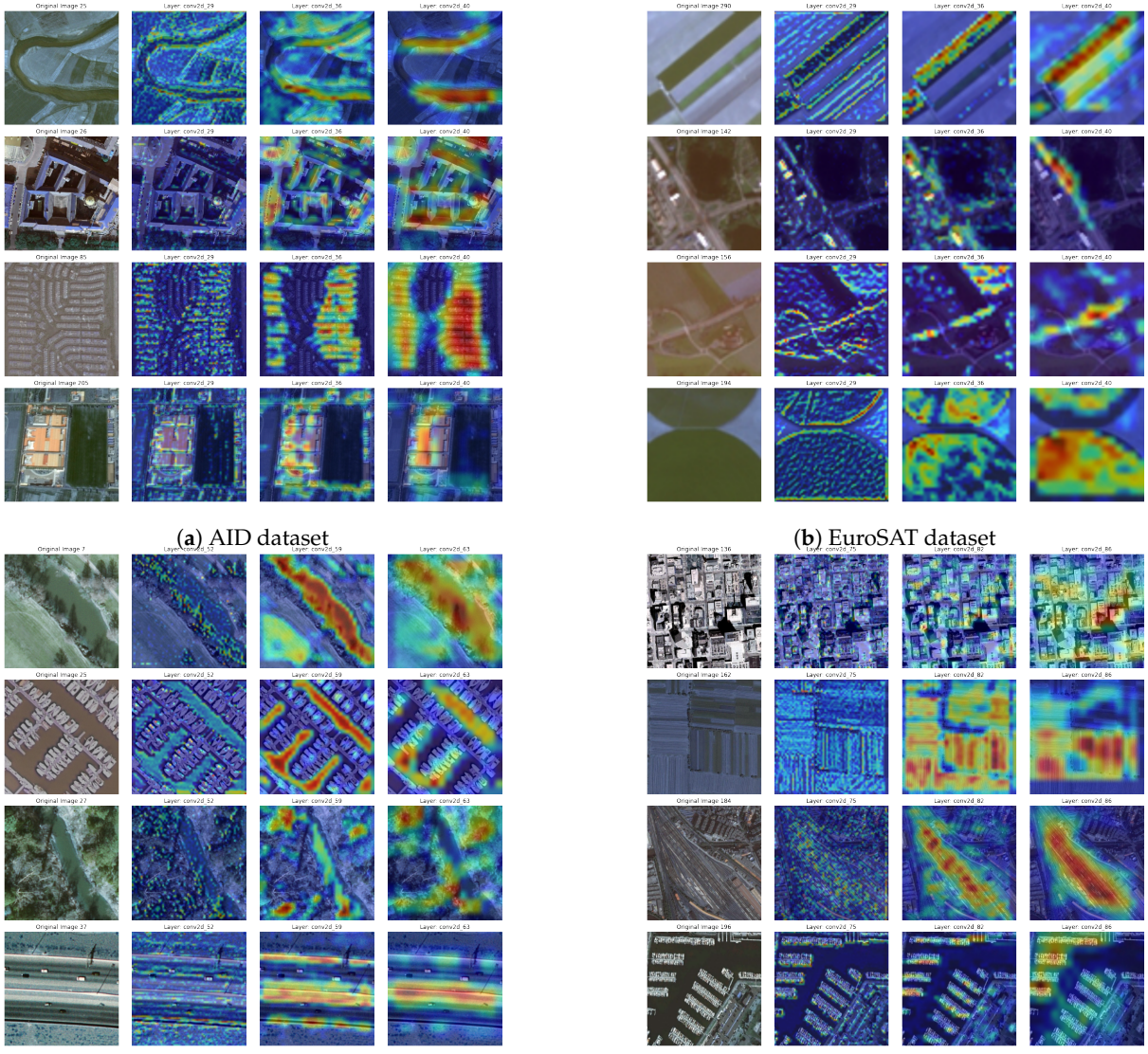
This work introduced variants of the benchmark MViT model and presented their training and evaluation outcomes on four GIS datasets: AID, EuroSAT, UC-Merced, and WHU-RS19. The results highlighted that MViT-Combined outperformed both the benchmark MViT and ViT architectures, achieving this with a significantly reduced number of parameters. While the SWIN models, including SWIN-ConvBlock and SWIN-ShuffleNet, did not surpass the MViT variants or the ViT in terms of performance, they demonstrated a commendable balance between parameter efficiency and accuracy, contributing a valuable perspective on model optimization. As a result, we can reason that our methods retained the benefits of CNNs and transformers while replacing some of the expensive deep-learning computational layers and boosting accuracy and reducing the number of training parameters.

The activation maps from these datasets can be seen in Figure 11. It is worth noting that, along with the successful variants, we also explored other modifications to CNN architectures that did not have a similar outcome. Five of the unsuccessful MViT variants involved removing entire MViT blocks from the overall architecture of the benchmark model in an attempt to decrease the number of training parameters. All of these variants had their respective architectures configured on the UC-Merced dataset and a 50–50 train–test split. One of these variants had a modified structure that removed the first MViT block from the overall MViT architecture. The removal resulted in the variant having 1,112,725 training parameters, which meant the number of training parameters was reduced by 15.24% relative to the benchmark MViT architecture. However, this variant could not maintain the benchmark's performance, as it achieved an accuracy of 78.00%, which was 9.14% lower than the 87.14% accuracy achieved by MViT. In addition, we created another variant that removed the last MViT block from the overall MViT architecture. The removal resulted in the variant having 687,605 training parameters, which meant the number of training parameters was reduced by 47.62% relative to the benchmark. Unfortunately, the variant attained a testing accuracy of 83.24%, which was lower than the benchmark's accuracy.

The other three out of the five variants for which we removed entire blocks from the overall architecture of the benchmark model focused on eliminating some of the inverted residual blocks. One of these variants had a modified architecture for which we removed the first inverted residual MV2 block from the overall benchmark architecture. This removal resulted in the variant having 1,309,501 training parameters, which meant the number of training parameters was reduced by 0.25%. However, the variant could not achieve the benchmark's performance, as it attained an accuracy of 84.19%, which was 2.95% lower than the benchmark's accuracy. In addition, we also created another variant that removed the second inverted residual MV2 block from the overall benchmark architecture. This

elimination resulted in the variant having 1,309,501 training parameters which meant the number of training parameters was reduced by 0.25%. Unfortunately, the variant also could not achieve the benchmark's accuracy as it attained an accuracy of 84.19% which was 2.95% lower than the benchmark's performance on the same dataset and train–test split.

Our final unsuccessful variant featured changes that were similar to the more fine-tuned and subtle manipulations made on the benchmark MViT model to generate the three successful variants. This variant was also configured on the UC-Merced dataset and a 50–50 train–test split before being compared to the original MViT benchmark configured on the same dataset and train–test split. It had a modified structure that replaced the last N × N convolutional layer in the MViT block's fusion section with an N × N depthwise separable convolution layer. In addition, this variant also had the changes from the MViT-Avg variant, which include the average pooling layer that replaced the second N × N convolutional layer in the MViT block's local representations section as well as an added upsampling layer in the MViT block's fusion section. This replacement resulted in the variant having 1,042,309 training parameters, which meant the number of training parameters was reduced by 20.60%. However, the variant could not attain the benchmark's accuracy, as it attained an accuracy of 77.81%, which was 9.33% lower than the benchmark's performance.



(**a**) AID dataset (**b**) EuroSAT dataset

(**c**) UC-Merced dataset (**d**) WHU-RS19 dataset

Figure 11. Class Activation Maps (CAMs) for different convolution layers of the MViT-Combined model where red highlights the class-specific image regions used by the model for classification.

Overfitting and Model Complexity

To ascertain if 500 epochs cause models to overfit, a benchmark test with a reduced training period of 200 epochs was conducted. In this test, the MViT model on the AID dataset with a 50–50 split showed a decrease in training accuracy from 99.46% to 98.54%, while the validation accuracy slightly improved from 85.82% to 86.06%. However, the MViT-Depth model on the same dataset and split saw a decrease in training accuracy from 99.46% to 99.37%, but the validation accuracy decreased from 87.44% to 82.90%, indicating that reduction in the number of epochs alone is not a universal solution for overfitting. The complex architectures of vision transformer models like MViT are a significant factor in this context. While their advanced self-attention mechanisms are adept at capturing intricate data patterns, this can sometimes lead to over-specialization on the training data, which does not always generalize to new, unseen data. Our 200-epoch benchmarks reveal that, while a reduced number of training epochs can lower the training accuracy, it does not automatically translate into higher test accuracy, emphasizing the need for a balanced approach in model training.

Moving forward, our efforts will focus on tailoring the architecture of vision transformers to better suit the characteristics of the datasets they are trained on. We aim to employ strategies like rigorous regularization, prudent model pruning, and extensive data augmentation to create models that are not only accurate but also possess strong generalization capability, which is essential for practical applications in GIS.

6. Conclusions

Our experiments revealed that a confluence of modern transformer architectures with lightweight CNN frameworks has the potential to yield superior outcomes. As a result, it opens up a vast array of possibilities wherein deep learning can be optimized and generalized across multiple domains without the need for complex loss functions or hyperparameters. Future work will explore further modifications: mostly to optimize the transformer layers. Model performance will also be integrated with the ImageNet dataset to explore possibilities for transfer learning. We note that lightweight models developed as a part of this initiative can have significant implications on edge devices such as sensors or UAVs. Deep learning models that are implemented directly on edge devices can reduce latency, minimize bandwidth usage, and ensure data privacy, since sensitive data may not need to leave the edge device. Finally, an endeavor to create usable GIS deep learning transformer-based models will be developed for the community by attempting to train and evaluate these models on a merged dataset composed of images from the AID, EuroSAT, UC-Merced, and WHU-RS19 datasets to ensure the continuity and validity of our results.

Author Contributions: Conceptualization, P.F.R.; Methodology, P.F.R. and R.G. (Rahul Gomes); Software, R.G. (Ravi Gadgil) and J.L.; Validation, R.G. (Ravi Gadgil), J.L. and P.K.; Formal analysis, R.G. (Ravi Gadgil), R.G. (Rahul Gomes) and P.K.; Investigation, G.S., G.M., W.I. and J.R.; Resources, P.K., Y.L., G.S., G.M., W.I. and J.R.; Data curation, R.G. (Rahul Gomes); Writing—original draft, Papia Rozario, R.G. (Ravi Gadgil) and Y.L.; Writing—review & editing, J.L. and R.G. (Rahul Gomes); Visualization, J.L. and W.I.; Supervision, P.F.R.; Project administration, P.F.R. All authors have read and agreed to the published version of the manuscript.

Funding: This article was funded by the National Science Foundation (NSF) Research Experience for Undergraduates (REU) grant OAC-2150191. We would also like to thank the Office of Research and Sponsored Programs (ORSP) at UW-Eau Claire for funding support. The computational resources of this study was provided by the Blugold Center for High-Performance Computing under NSF grant CNS-1920220.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data from this research are available on GitHub (https://github.com/rahulgomes19/gis-transformer (accessed on 23 September 2023)).

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

Abbreviations

The following abbreviations are used in this manuscript:

CNN	Convolutional Neural Network
ViT	Vision Transformer
MViT	Mobile Vision Transformer
GIS	Geographic Information Systems
ResNet	Residual Network
AID	Aerial Image Dataset
NWPU45	Northwestern Polytechnical University Remote Sensing Image Scene Classification
RSSCN7	Remote Sensing Scene Classification Network
ReLU	Rectified Linear Unit
SWIN	Shifted Windows
MHSA	Multi-Head Self-Attention

Appendix A. Training and Testing Accuracy Values

Table A1. Final training accuracies for all the models.

	Dataset	Model	Split Category				Dataset	Split Category			
			20–80	40–60	50–50	60–40		20–80	40–60	50–50	60–40
Train Acc	AID	ViT	99.25	99.1	99.14	98.82	UC-Merced	99.52	99.76	98.86	99.76
		MViT	99.05	99.4	99.46	99.28		98.57	98.69	99.76	98.95
		MViT-Depth	99.45	99.62	99.46	99.9		100	99.64	100	99.84
		MViT-Combined	100	99.5	99.96	99.93		100	100	100	100
		SWIN-Reg	97.1	97.12	97.34	97.18		75.71	90.48	92.38	98.25
		SWIN-Conv Block	99.95	99.73	99.62	99.45		100	99.88	100	99.92
		SWIN-ShuffleNet	99.95	99.67	99.36	99.47		100	100	99.9	99.84
	EuroSat	ViT	99.8	99.5	99.6	99.44	WHU-RS19	100	99.25	99.8	99
		MViT	99.8	99.8	99.84	99.85		98.51	99.25	99.8	96.35
		MViT-Depth	99.78	99.87	99.75	99.63		99	98.51	99.6	99.67
		MViT-Combined	99.96	99.67	99.88	99.83		99	100	100	100
		SWIN-Reg	98.41	98.5	98.56	97.89		88.06	95.27	94.82	96.02
		SWIN-Conv Block	99.8	99.65	99.62	99.77		100	100	100	100
		SWIN-ShuffleNet	99.55	99.68	99.52	99.61		100	100	99.8	100
Test Acc	AID	ViT	56.71	66.22	68.7	71.5	UC-Merced	50.77	62.22	67.05	75.71
		MViT	73.96	82.9	85.82	86.58		63.39	81.35	87.14	87.62
		MViT-Depth	75.29	84.28	87.44	88.25		76.9	83.25	91.71	86.55
		MViT-Combined	76.89	85.67	86.52	89.42		80.48	88.02	92.1	90
		SWIN-Reg	39.47	49.47	45.08	53.65		35.06	42.7	50.57	55.12
		SWIN-Conv Block	41.96	45.7	53.74	56.77		41.37	48.41	60.1	58.1
		SWIN-ShuffleNet	40.89	50.87	55.12	52.75		44.4	54.76	57.43	56.43
	EuroSat	ViT	91.78	92.65	92.97	94.43	WHU-RS19	46.77	57.38	60.83	60.45
		MViT	95.71	96.36	97.98	97.69		66.04	79.77	81.31	64.18
		MViT-Depth	92.08	97.71	97.67	96.49		63.81	79.77	81.11	77.36
		MViT-Combined	97.35	97.8	97.32	97.08		61.32	80.43	82.11	86.32
		SWIN-Reg	86.48	89.6	90.8	90.77		34.08	49.92	52.49	50.75
		SWIN-Conv Block	86.03	85.61	89.65	85.04		54.73	53.57	60.24	60.45
		SWIN-ShuffleNet	87.23	92.07	91.92	91.94		52.61	59.54	59.64	62.19

References

1. Chowdhary, K.; Chowdhary, K. Natural language processing. In *Fundamentals of Artificial Intelligence*; Springer: New Delhi, India, 2020; pp. 603–649.
2. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. In Proceedings of the Advances in Neural Information Processing Systems, Lake Tahoe, NV, USA, 3–6 December 2012; Volume 25.
3. Madurapperuma, B.; Rozario, P.; Oduor, P.; Kotchman, L. Land use and land cover change detection in Pipestem Creek watershed, North Dakota. *Int. J. Geomat. Geosci.* **2015**, *5*, 416–426.
4. Haffner, M.; DeWitte, M.; Rozario, P.F.; Ovando-Montejo, G.A. A Neural-Network-Based Landscape Search Engine: LSE Wisconsin. *Appl. Sci.* **2023**, *13*, 9264. [CrossRef]
5. Rozario, P.F.; Oduor, P.; Kotchman, L.; Kangas, M. Quantifying spatiotemporal change in landuse and land cover and accessing water quality: A case study of Missouri watershed james sub-region, north Dakota. *J. Geogr. Inf. Syst.* **2016**, *8*, 663. [CrossRef]
6. Kussul, N.; Lavreniuk, M.; Skakun, S.; Shelestov, A. Deep learning classification of land cover and crop types using remote sensing data. *IEEE Geosci. Remote Sens. Lett.* **2017**, *14*, 778–782. [CrossRef]
7. Ma, L.; Liu, Y.; Zhang, X.; Ye, Y.; Yin, G.; Johnson, B.A. Deep learning in remote sensing applications: A meta-analysis and review. *ISPRS J. Photogramm. Remote Sens.* **2019**, *152*, 166–177. [CrossRef]
8. Hong, D.; Gao, L.; Yokoya, N.; Yao, J.; Chanussot, J.; Du, Q.; Zhang, B. More diverse means better: Multimodal deep learning meets remote-sensing imagery classification. *IEEE Trans. Geosci. Remote Sens.* **2020**, *59*, 4340–4354. [CrossRef]
9. Lin, T.; Wang, Y.; Liu, X.; Qiu, X. A survey of transformers. *AI Open* **2022**, *3*, 111–132. [CrossRef]
10. Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; Zagoruyko, S. End-to-end object detection with transformers. In Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 213–229.
11. 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.
12. Dimitrovski, I.; Kitanovski, I.; Kocev, D.; Simidjievski, N. Current trends in deep learning for Earth Observation: An open-source benchmark arena for image classification. *ISPRS J. Photogramm. Remote Sens.* **2023**, *197*, 18–35. [CrossRef]
13. Deng, J.; Dong, W.; Socher, R.; Li, L.; Li, K.; Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Conference On Computer Vision And Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 248–255.
14. Reedha, R.; Dericquebourg, E.; Canals, R.; Hafiane, A. Transformer neural network for weed and crop classification of high resolution UAV images. *Remote Sens.* **2022**, *14*, 592. [CrossRef]
15. Mehta, S.; Rastegari, M. Mobilevit: Light-weight, general-purpose, and mobile-friendly vision transformer. *arXiv* **2021**, arXiv:2110.02178.
16. Gomes, R.; Rozario, P.; Adhikari, N. Deep learning optimization in remote sensing image segmentation using dilated convolutions and ShuffleNet. In Proceedings of the 2021 IEEE International Conference on Electro Information Technology (EIT), Mt. Pleasant, MI, USA, 14–15 May 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 244–249.
17. Cheng, Q.; Li, X.; Zhu, B.; Shi, Y.; Xie, B. Drone detection method based on MobileViT and CA-PANet. *Electronics* **2023**, *12*, 223. [CrossRef]
18. Wan, Z.; Wan, J.; Cheng, W.; Yu, J.; Yan, Y.; Tan, H.; Wu, J. A Wireless Sensor System for Diabetic Retinopathy Grading Using MobileViT-Plus and ResNet-Based Hybrid Deep Learning Framework. *Appl. Sci.* **2023**, *13*, 6569. [CrossRef]
19. Xie, S.; Girshick, R.; Dollár, P.; Tu, Z.; He, K. Aggregated residual transformations for deep neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 1492–1500.
20. Hu, J.; Shen, L.; Sun, G. Squeeze-and-excitation networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7132–7141.
21. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv* **2017**, arXiv:1704.04861.
22. Zhang, X.; Zhou, X.; Lin, M.; Sun, J. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 6848–6856.
23. Huang, X.; Liu, F.; Cui, Y.; Chen, P.; Li, L.; Li, P. Faster and better: A lightweight transformer network for remote sensing scene classification. *Remote Sens.* **2023**, *15*, 3645. [CrossRef]
24. Zhang, X.; Yu, W.; Pun, M.O. Multilevel deformable attention-aggregated networks for change detection in bitemporal remote sensing imagery. *IEEE Trans. Geosci. Remote Sens.* **2022**, *60*, 5621518. [CrossRef]
25. Lv, P.; Wu, W.; Zhong, Y.; Du, F.; Zhang, L. SCViT: A spatial-channel feature preserving vision transformer for remote sensing image scene classification. *IEEE Trans. Geosci. Remote Sens.* **2022**, *60*, 4409512. [CrossRef]
26. Liu, F.; Chen, D.; Guan, Z.; Zhou, X.; Zhu, J.; Zhou, J. RemoteCLIP: A Vision Language Foundation Model for Remote Sensing. *arXiv* **2023**, arXiv:2306.11029.
27. Yuan, Z.; Liu, X. Research on Remote Sensing Image Classification Based on Lightweight Convolutional Neural Network. In Proceedings of the 2022 International Conference on Artificial Intelligence, Internet and Digital Economy (ICAID 2022), Shenzhen, China, 15–17 April 2022; Atlantis Press: Amsterdam, The Netherlands, 2022; pp. 127–138.

28. Balarabe, A.T.; Jordanov, I. Interpolation and Context Magnification Framework for Classification of Scene Images. In Proceedings of the International Conferences Computer Graphics, Visualization, Computer Vision and Image Processing (CGVCVIP), Warsaw, Poland, 19–21 September 2022; pp. 93–100.
29. Chen, Z.; Yang, J.; Feng, Z.; Chen, L. RSCNet: An Efficient Remote Sensing Scene Classification Model Based on Lightweight Convolution Neural Networks. *Electronics* **2022**, *11*, 3727. [CrossRef]
30. He, C.; He, B.; Yin, X.; Wang, W.; Liao, M. Relationship prior and adaptive knowledge mimic based compressed deep network for aerial scene classification. *IEEE Access* **2019**, *7*, 137080–137089. [CrossRef]
31. Shi, C.; Zhao, X.; Wang, L. A multi-branch feature fusion strategy based on an attention mechanism for remote sensing image scene classification. *Remote Sens.* **2021**, *13*, 1950. [CrossRef]
32. Xu, C.; Shu, J.; Zhu, G. Scene Classification Based on Heterogeneous Features of Multi-Source Data. *Remote Sens.* **2023**, *15*, 325. [CrossRef]
33. Lakshmi, T.V.; Reddy, C.V.K.; Kora, P.; Swaraja, K.; Meenakshi, K.; Kumari, C.U.; Reddy, L.P. Classification of multi-spectral data with fine-tuning variants of representative models. *Multimed. Tools Appl.* **2024**, *83*, 23465–23487. [CrossRef]
34. Noppitak, S.; Surinta, O. Deep Learning for Land Use and Land Cover in Aerial Images. Ph.D. Thesis, Mahasarakham University, Kham Riang, Thailand, 2022.
35. Wadekar, S.N.; Chaurasia, A. Mobilevitv3: Mobile-friendly vision transformer with simple and effective fusion of local, global and input features. *arXiv* **2022**, arXiv:2209.15159.
36. Mehta, S.; Rastegari, M. Separable self-attention for mobile vision transformers. *arXiv* **2022**, arXiv:2206.02680.
37. Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 10012–10022.
38. Xia, G.S.; Hu, J.; Hu, F.; Shi, B.; Bai, X.; Zhong, Y.; Zhang, L.; Lu, X. AID: A benchmark data set for performance evaluation of aerial scene classification. *IEEE Trans. Geosci. Remote Sens.* **2017**, *55*, 3965–3981. [CrossRef]
39. Helber, P.; Bischke, B.; Dengel, A.; Borth, D. Introducing EuroSAT: A Novel Dataset and Deep Learning Benchmark for Land Use and Land Cover Classification. In Proceedings of the IGARSS 2018—2018 IEEE International Geoscience and Remote Sensing Symposium, Valencia, Spain, 22–27 July 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 204–207.
40. Helber, P.; Bischke, B.; Dengel, A.; Borth, D. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2019**, *12*, 2217–2226. [CrossRef]
41. Yang, Y.; Newsam, S. Bag-of-visual-words and spatial extensions for land-use classification. In Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems, San Jose, CA, USA, 2–5 November 2010; pp. 270–279.
42. Xia, G.S.; Yang, W.; Delon, J.; Gousseau, Y.; Sun, H.; MaÎtre, H. Structural high-resolution satellite image indexing. In Proceedings of the SPRS TC VII Symposium-100 Years ISPRS 2010, Vienna, Austria, 5–7 July 2010.
43. Dai, D.; Yang, W. Satellite Image Classification via Two-Layer Sparse Coding With Biased Image Representation. *IEEE Trans. Geosci. Remote Sens.* **2011**, *8*, 173–176. [CrossRef]
44. Gascon, F.; Cadau, E.; Colin, O.; Hoersch, B.; Isola, C.; Fernández, B.L.; Martimort, P. Copernicus Sentinel-2 mission: Products, algorithms and Cal/Val. In Proceedings of the Earth Observing Systems XIX, San Diego, CA, USA, 17–21 August 2014; SPIE: Bellingham, WA, USA, 2014; Volume 9218, pp. 455–463.
45. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.