sustainability

MDPI

Article

A Comparison of Deep Transfer Learning Methods for Land Use and Land Cover Classification

Hatef Dastour and Quazi K. Hassan *

Department of Geomatics Engineering, University of Calgary, 2500 University Drive NW, Calgary, AB T2N 1N4, Canada
* Correspondence: qhassan@ucalgary.ca

Abstract: The pace of Land Use/Land Cover (LULC) change has accelerated due to population growth, industrialization, and economic development. To understand and analyze this transformation, it is essential to examine changes in LULC meticulously. LULC classification is a fundamental and complex task that plays a significant role in farming decision making and urban planning for long-term development in the earth observation system. Recent advances in deep learning, transfer learning, and remote sensing technology have simplified the LULC classification problem. Deep transfer learning is particularly useful for addressing the issue of insufficient training data because it reduces the need for equally distributed data. In this study, thirty-nine deep transfer learning models were systematically evaluated alongside multiple deep transfer learning models for LULC classification using a consistent set of criteria. Our experiments will be conducted under controlled conditions to provide valuable insights for future research on LULC classification using deep transfer learning models. Among our models, ResNet50, EfficientNetV2B0, and ResNet152 were the top performers in terms of kappa and accuracy scores. ResNet152 required three times longer training time than EfficientNetV2B0 on our test computer, while ResNet50 took roughly twice as long. ResNet50 achieved an overall f1-score of 0.967 on the test set, with the Highway class having the lowest score and the Sea Lake class having the highest.

Keywords: land use and land cover; image classification; deep learning; transfer learning; EuroSAT; earth observation; satellite image classification

Citation: Dastour, H.; Hassan, Q.K. A Comparison of Deep Transfer Learning Methods for Land Use and Land Cover Classification. *Sustainability* **2023**, *15*, 7854. https://doi.org/10.3390/su15107854

Academic Editor: Antonio Miguel Martínez-Graña

Received: 11 March 2023
Revised: 3 April 2023
Accepted: 9 May 2023
Published: 11 May 2023

1. Introduction

The impact of changes in Land Use/Land Cover (LULC) has become a cause for concern due to their contribution to reducing biodiversity, influencing climate patterns, and creating Urban Heat Islands (UHI) in cities [1]. Land cover pertains to the physical characteristics of the land's surface, while land use refers to how humans utilize a specific area of land. As they are intertwined, land cover and land use are frequently studied together [1–4].

The classification of Land Use/Land Cover (LULC) is a crucial and challenging task in the field of earth observation that contributes to farming decision making, urban management for economic sustainability, regional management, and environmental conservation and management [5–7]. Land cover land use image classification presents several other challenges, such as the expensive nature of gathering hyperspectral data, scarcity of labeled training samples, the inclusion of irrelevant and redundant spectral bands, which can result in the Hughes phenomenon, and the need to address issues related to atmospheric correction, noise removal, and band calibration [8,9]. Land cover land use image classification also faces other challenges such as the need to process large amounts of data, which may require big data infrastructure and the limited availability of cloud-free images for large areas. While satellite and aerial imagery can provide land cover information, they cannot always accurately determine land use. Additionally, temporal changes in land cover can be better captured by analyzing land cover maps over several years [8,9].

Recent advancements in LULC classification have improved noise reduction, cloud shadow suppression, segmentation, and classification techniques [10–13]. However, the classification remains a complex process due to the increased level of abstraction from pixels to items to landscapes and the diverse geographic distribution of various land cover categories [12,14]. The aim of this study is to compare thirty-nine open-source deep transfer learning methods for LULC classification using various accuracy metrics and training times, given the importance of this task. This is a novel contribution, as no previous studies have systematically evaluated multiple deep transfer learning models for LULC classification using a consistent set of criteria. By conducting rigorous experiments on these 39 methods under controlled conditions, we can provide valuable insights for future research on LULC classification using deep transfer learning models. These insights will help to reduce the time required to choose the best model for the dataset size and thus enable more efficient and effective LULC classification.

Recent studies have demonstrated that Convolutional Neural Networks (CNNs), which are deep learning methods, are commonly utilized in remote sensing image classification due to their remarkable accuracy and performance [15]. However, when training deep-learning models from scratch, the added complexity may lead to overfitting [12,16]. To address this issue, deep transfer learning has emerged as a new learning framework [17]. By utilizing transfer learning, it is possible to significantly reduce the need for training data and training time in the target domain [18]. This is because the model does not need to be trained from scratch in the target domain, and the train and test sets do not need to be independent or have identical distributions [18]. The primary objective of transfer learning is to improve the performance of designated learners on specific domains by transferring the knowledge gained in various but related source domains [19]. Transfer learning's diverse application opportunities have made it a prominent and captivating subject in the realm of machine learning [19]. Alem et al. [20] applied deep transfer learning techniques to the UC Merced dataset. Their models achieved overall accuracy scores of 92.46%, 94.38%, and 99.64% using Resnet50V2, InceptionV3, and VGG19, respectively.

Several studies have explored the use of pre-trained models on remote sensing image classification using CNN-based models [21–24]. Huang et al. [6] proposed a method that utilizes transferring learning on the high-resolution SAR-labeled land cover labeled dataset [25]. Their method exhibited high efficiency and achieved an overall accuracy of 99.46%. Li et al. [26] presented a High-Spatial-Resolution Remote Sensing (HSRRS) [27] image scene classification procedure based on transfer learning and the DeCNN (TL-DeCNN) model. Compared to the VGG19 [28], ResNet50 [29], and InceptionV3 [30], their results showed that CNNs with transfer learning can deliver substantially greater accuracy without overfitting [26].

Regarding LULC classification on the EuroSAT dataset [31,32], a labeled dataset using Sentinel-2 satellite images, Helber et al. implemented a patch-based classification model for LULC classification [31,33]. They developed a dataset with exactly 27,000 labeled and geo-referenced images, each with 13 bands and divided into ten categories. Based on these images, they examined several band configurations of the GoogleNet [34] and ResNet-50 [29] architectures and discovered that ResNet-50 with RGB bands provided the best accuracy [31]. The EuroSAT dataset was separated into two sets: a train set (80%) for training and a test set (20%) for validation and testing. The overall classification accuracy for the new adaptive dataset was 98.57% [31,33]. Using the RGB bands, the Deep Discriminative Representation Learning with Attention Map (DDRL-AM) approach was presented by Li et al. [35]. The authors segregated the EuroSAT dataset into a training set (80%) and a testing set (20%) and achieved an overall accuracy of 98.74% [35]. Yassine et al. [36] investigated two methods for enhancing the accuracy of LULC classification using the EuroSAT dataset. The first procedure utilized Sentinel-2's thirteen spectrums for feature extraction, with 98.78% accuracy. In the second technique, thirteen Sentinel-2 spectral feature bands were utilized for feature extraction coupled with computed indices such as Vegetation Index based on the Red Edge (VIRE), Normalized Near-Infrared (NNIR), and Blue Ratio

(BR). They divided the Eurosat dataset into a train set (70%), a validation set (20%), and a test set (10%) per class. Their overall accuracy is 99.58% [36]. Naushad et al. [12] used the RGB version of the EuroSAT dataset in conjunction with transfer learning. They enhanced performance and processing times by employing techniques such as adaptive learning rates, early stopping, gradient clipping, and data augmentation. They divided the dataset into a train (75%) set and a test (25%) set which was utilized for both validating and testing. Their results indicated that their proposed method based on Wide ResNet-50 reached an overall accuracy of 99.17% [12].

The remainder of this article is organized as follows. Section 2 describes the EuroSAT dataset, deep transfer learning algorithms, data augmentations, EuroSAT dataset subsets for training, validating and evaluating our models, early stopping and learning rate optimizations, the loss and optimizer of models, and accuracy metrics. Section 3 summarizes the key findings of our numerical experiments, including model comparisons through the training time and accuracy metrics. Section 4 discusses the results, and Section 5 concludes this article.

2. Materials and Methods

2.1. Dataset

2.1.1. EuroSAT Dataset

Helber et al. [31] utilized multi-spectral labeled image data from the Sentinel-2A satellite for LULC classification. Sentinel-2A and Sentinel-2B are two comparable land monitoring satellites that constitute a two-satellite constellation. The satellites were efficiently deployed in June 2015 (Sentinel-2A) and March 2017 (Sentinel-2B) [31]. A collection of 27,000 images, each 64 by 64 pixels in size, was manually labeled and grouped into 10 categories, each with 2000–3000 images [31]. Helber et al. specified ten LULC classes that are distinguishable at a resolution of 10 m/pixel and are regularly covered by the European Urban Atlas [31,33]. The categorized EuroSAT dataset is already accessible to the general public [32]. The RGB variant of the EuroSAT collection was utilized for training, validation, and testing purposes. Figure 1 illustrates a subset sample of EuroSAT collection from the EuroSAT collection [31,33].

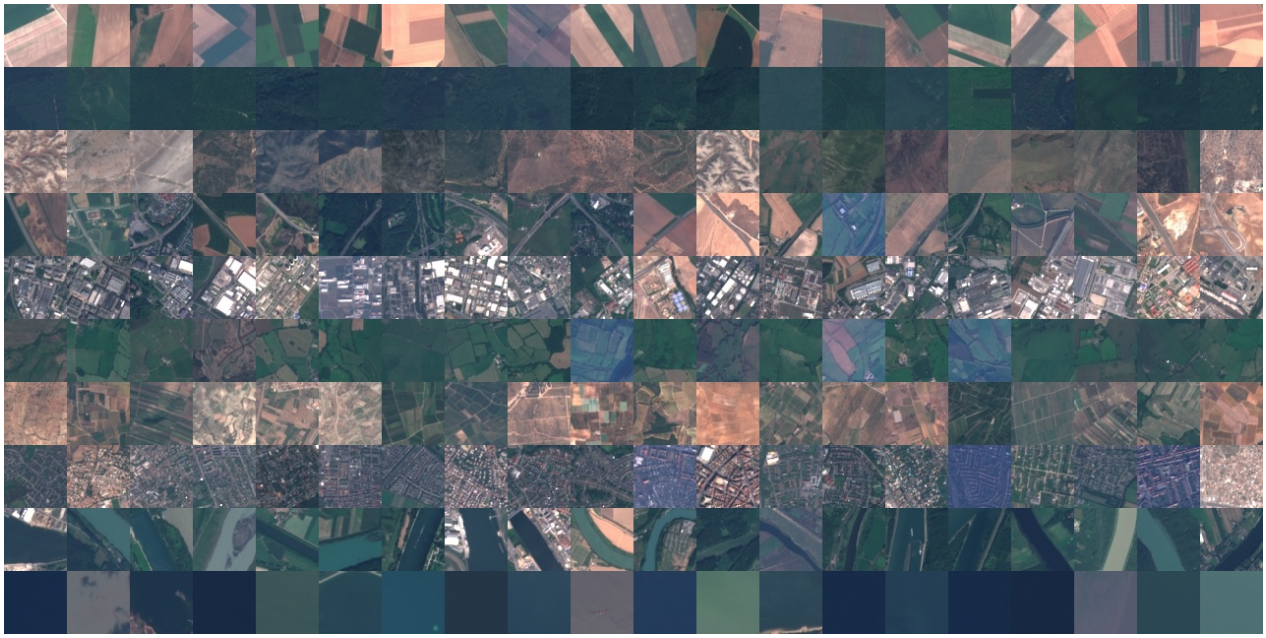

Figure 1. A sample of images from the EuroSAT dataset. There are 20 sample images per row from the following categories (from top to bottom), respectively: Annual Crop, Forest, Herbaceous Vegetation, Highway, Pasture, Industrial, Permanent Crop, Residential, River, and Sea/Lake.

2.1.2. Data Augmentation

The term "data augmentation" refers to a group of methods performed to increase the amount and caliber of training datasets so that better deep-learning models may be created with their support [37,38]. Data augmentations include geometric modifications, RGB color modifications, kernel filters, image blending, arbitrary cleaning, feature augmentation, and so on [37,38]. The data augmentation techniques employed in this study for the EuroSAT dataset were horizontal/vertical flipping, random brightness, random saturation, random contrast, and random cropping.

2.1.3. Train, Validation and Test Sets

The ratio of train, validation, and test set sizes do not have a perfect balance and cannot be estimated beforehand; instead, it needs to be determined based on the abundance of data supplied [39]. For a small dataset, the validation and testing ratio will be considerably higher than the amount used for training. On the contrary, as the volume of data increases, the validation and testing ratio will decrease [39]. Ideally, the test set should be a dataset acquired independently. Having a large number of independent test sets strengthens the generalization argument [39]. The EuroSAT dataset was divided into three groups: the train set (80%), the validation set (10%), and the test set (10%). These EuroSAT dataset subsets were randomly chosen; nonetheless, the same subsets were utilized for training and evaluating all models. All images were rescaled to 200 by 200 pixels, 128-photo mini-batches were used, and only images from the train set were subjected to data augmentation methods.

2.2. Methods

2.2.1. A Brief Review of Used Deep Learning Models for Transfer Learning

The DenseNet, introduced by Huang et al. [40], is a convolutional neural network that connects each layer to every other layer in a feed-forward manner. Unlike traditional convolutional networks, which have one connection between each layer and its next layer, the DenseNet incorporates all previous layers' representations as inputs for each layer. Moreover, each layer's own feature maps are used as inputs for all subsequent layers, resulting in a network with $L(L+1)/2$ direct links, where L is the number of layers in the network [40].

Research conducted by Tan et al. [41] analyzed model scaling and exhibited that performance may be improved by scrupulously offsetting network depth, thickness, and resolution. They proposed a novel scaling technique, Efficientnet, that employs a straightforward though incredibly powerful coefficient to equally scale all depth, extent, and resolution parameters [41]. Tan et al. also presented the EfficientNetV2 family of convolutional networks [42], which exhibit faster training speeds and higher parameter efficiency than the original EfficientNet [41]. They demonstrated that EfficientNetV2 models can be trained in a shorter time than state-of-the-art models while also being up to 6.8 times smaller [42]. The authors noted that gradually increasing the image size during training can further speed up the process, but this may lead to a decrease in accuracy. To address this accuracy drop, they proposed a progressive learning strategy that dynamically adjusts regularization, such as data augmentation, along with image size [42].

Howard et al. [43] introduced the MobileNets family of efficient models for mobile and embedded computer vision applications. MobileNets use a simplified architecture that employs depth-wise separable convolutions to build compact deep neural networks [43]. The authors proposed two simple global hyper-parameters that effectively balance training time and accuracy [43], enabling model developers to choose an appropriately sized model for their specific application based on the problem's constraints. Sandler et al. [44] improved the performance of mobile models on various tasks and benchmarks by introducing MobileNetV2, which is designed to perform well across a range of model sizes.

Radosavovic et al. [45] utilized the quantized linear function to explain the widths and depths of successful networks and developed RegNet based on these parameters. Through an analysis of the RegNet design space, they uncovered some interesting results

that challenge the conventional approach to network design. The authors claim that RegNet models may outperform well-known EfficientNet models while being up to five times faster on GPUs with similar training settings [45].

AlexNet, proposed by Krizhevsky et al. [46], is a CNN architecture that was developed for object detection. Although obtaining access to large-scale databases is a significant obstacle in CNN training [47], AlexNet was trained on the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) database [48], along with augmentation techniques. On the other hand, Simonyan et al. [28] presented the VGG16 architecture for the object recognition task. VGG19, an enhanced version of VGG16, improves upon the limitations of AlexNet and enhances the system's accuracy.

He et al. introduced ResNet [29], a framework for residual learning that makes it easier to train deeper networks than those previously used. Instead of learning non-referenced functions, they rearranged the layers to learn residual functions based on the layer inputs. They presented extensive experimental evidence showing that these residual networks are simpler to tune and can benefit from much greater depth [29]. They evaluated residual networks up to 152 layers deep on the ImageNet dataset [49], which is eight times deeper than VGG networks while still being less complex [29]. The ResNet architecture was reviewed by Bello et al. [50] who proposed the ResNet-RS family of models. These models were designed to be faster than EfficientNets on cloud Tensor Processing Units (TPUs), achieving similar accuracies on the ImageNet dataset while utilizing advanced training and scaling techniques. According to the authors, the ResNet-RS models are 1.7×–2.7× faster than EfficientNets on TPUs [50].

In summary, each of these architectures has its own strengths and weaknesses. VGG is a deep learning architecture with a deep structure consisting of 16 or 19 convolutional layers [51], while ResNet, which introduced residual learning, is capable of being trained with a depth of up to 152 layers, making it 8 times deeper than the VGG network, while still having lower complexity. When evaluated on the ImageNet dataset, an ensemble of these ResNet models achieved an impressive error rate of 3.57% on the test set, winning first place in the ILSVRC 2015 classification task [29]. MobileNet is considered a lightweight deep neural network due to its fewer parameters and higher classification accuracy. To enhance its performance, MobileNet has integrated dense blocks from DenseNets, which further reduce the number of network parameters while improving classification accuracy [40]. Compared to MobileNet, DenseNet has a higher number of parameters and is more computationally intensive, making it less suitable for mobile and embedded devices. However, DenseNet can achieve state-of-the-art performance on many computer vision tasks and is often used in applications such as object detection and image segmentation. EfficientNet is a recent architecture that achieves state-of-the-art performance with fewer parameters and FLOPS compared to other models [41,52]. The choice of which architecture to use depends on the specific requirements of the application, such as accuracy, speed, memory, and hardware constraints.

2.2.2. Deep Transfer Learning Methods

A domain consists of two components: a feature space, denoted as $\mathcal{X}$, and a marginal probability distribution $P(X)$, where $X = \{x_1, \ldots, x_n\} \in \mathcal{X}$. For instance, consider a machine learning application for software module defect classification, where each software metric serves as a feature. Here, x_i represents the i-th feature vector, or instance, corresponding to the i-th software module. The total number of feature vectors is denoted as n, while $\mathcal{X}$ represents the space of all possible feature vectors, and a specific learning sample is represented by X [53]. In a given domain $\mathcal{D}$, a task $\mathcal{T}$ is defined by a label space $\mathcal{Y}$ and a predictive function $f(\cdot)$, which is learned from the pairs of feature vectors and labels $\{x_i, y_i\}$, where $x_i \in X$ and $y_i \in \mathcal{Y}$. In the software module defect classification application, for example, $\mathcal{Y}$ comprises true and false labels, with y_i taking on either of these values. The learner function $f(x)$ predicts the label value for a given software module x. The domain $\mathcal{D}$ is represented as $\mathcal{D} = \{\mathcal{X},\ P(X)\}$, while the task $\mathcal{T}$ is denoted as $\mathcal{T} = \{\mathcal{Y},\ f(\cdot)\}$ [53]. $\mathcal{D}_{\mathcal{S}}$ rep-

resents the source domain data, where $\mathcal{D}_S = \{(x_{s,1}, y_{s,1}), \ldots, (x_{s,n}, y_{s,n})\}$, with $x_{s,i} \in \mathcal{X}_S$ as the i-th data instance of $\mathcal{D}_S$, and $y_{s,i} \in \mathcal{Y}_S$ as the corresponding class label for xs_i. Similarly, $\mathcal{D}_T$ represents the target domain data, where $\mathcal{D}_T = \{(x_{T,1}, y_{T,1}), \ldots, (x_{T,n}, y_{T,n})\}$, with $x_{T,i} \in \mathcal{X}_T$ as the i-th data instance of $\mathcal{D}_T$, and $y_{T,i} \in \mathcal{Y}_T$ as the corresponding class label for $x_{T,i}$. The source task is represented as $\mathcal{T}_S$, the target task is represented as $\mathcal{T}_T$, the source predictive function is represented as $f_S(\cdot)$, and the target predictive function is represented as $f_T(\cdot)$ [53].

The formal definition of transfer learning states that it involves improving the predictive function $f(\cdot)$ for a target domain $\mathcal{D}_T$ and task $\mathcal{T}_T$ by leveraging related information from a source domain $\mathcal{D}_S$ and task $\mathcal{T}_S$, where either $\mathcal{D}_S \neq \mathcal{D}_T$ or $\mathcal{T}_S \neq \mathcal{T}_T$ [53]. It is possible to extend the single source domain to multiple source domains. When $\mathcal{D}_S = \{\mathcal{X}_S, P(X_S)\}$ and $\mathcal{D}_T = \{\mathcal{X}_T, P(X_T)\}$, the condition where $\mathcal{D}_S \neq \mathcal{D}_T$ means that $\mathcal{X}_S \neq \mathcal{X}_T$ and/or $P(X_S) \neq P(X_T)$ [53].

Contrary to standard deep learning approaches, transfer learning techniques leverage knowledge accumulated from data in related domains (source domain) to enable predictive modeling comprising multiple data patterns in the target domain. To perform deep transfer learning, a pre-trained neural network is adjusted to tackle a different task by fine-tuning its parameters. The pre-trained model is usually trained on an extensive dataset such as ImageNet, and its knowledge can be applied to new tasks with similar characteristics. This process entails freezing the pre-trained layers' weights and training new layers to tailor the model to the new task [18,47]. A list of thirty-nine models used for transfer learning in this study:

- DenseNet121, DenseNet169, and DenseNet201 [40];
- EfficientNetB0–EfficientNetB7 [41];
- EfficientNetV2B0–EfficientNetV2B3 [42];
- InceptionV3 [30];
- MobileNetV1 [43] and MobileNetV2 [44];
- NASNetMobile [54];
- RegNetX002, RegNetX004, RegNetX006, RegNetX008, and RegNetX032 [45];
- ResNet50, ResNet101, and ResNet152 [29];
- ResNetRS50, ResNetRS101, ResNetRS152, ResNetRS200, ResNetRS270, ResNetRS350, and ResNetRS420 [50];
- ResNet50V2, ResNet101V2, and ResNet152V2 [55];
- VGG16 and VGG19 [28].

The workflow utilized to train the 39 pre-trained models is summarized in Figure 2. In each instance, we removed the top layer of our pre-trained models, kept the underlying layers frozen, and then trained the added layers, which will be discussed in Section 2.2.3, using the train set. These added layers were fine-tuned using the validation set.

2.2.3. Modifications in Network Architecture

All pre-trained models employed in this research were from the Tensorflow v2.10.0 library [56] which were pre-trained using the ImageNet dataset [49] on Google servers. The top layer of each of these previously trained models was removed while the underlying layers were frozen, and no training was performed on these layers. Then, we added a few new, trainable layers on top of the frozen layers. By training only these new layers, the models learned to turn the old features into predictions on the EuroSAT dataset. Table 1 highlights new layers added on top of the previously trained models' frozen layers. The first layer after the frozen layers is Global Average Pooling 2D, which involved the global average pooling calculation for spatial data [57]. The next layer, Flatten, squeezes the input data to a single dimension rather than two dimensions [56]. After this layer, there is the first Dense layer which is a fully connected layer [56] with ReLU activation [58]. The size of the dense layer before the output dense layer was chosen at 128, 256, 1024, and 4096. We tested all of these sizes on each model and kept the best one only in each case. The final

layer is another Dense layer which is used in conjunction with softmax activation [56,59] for the output of the classification.

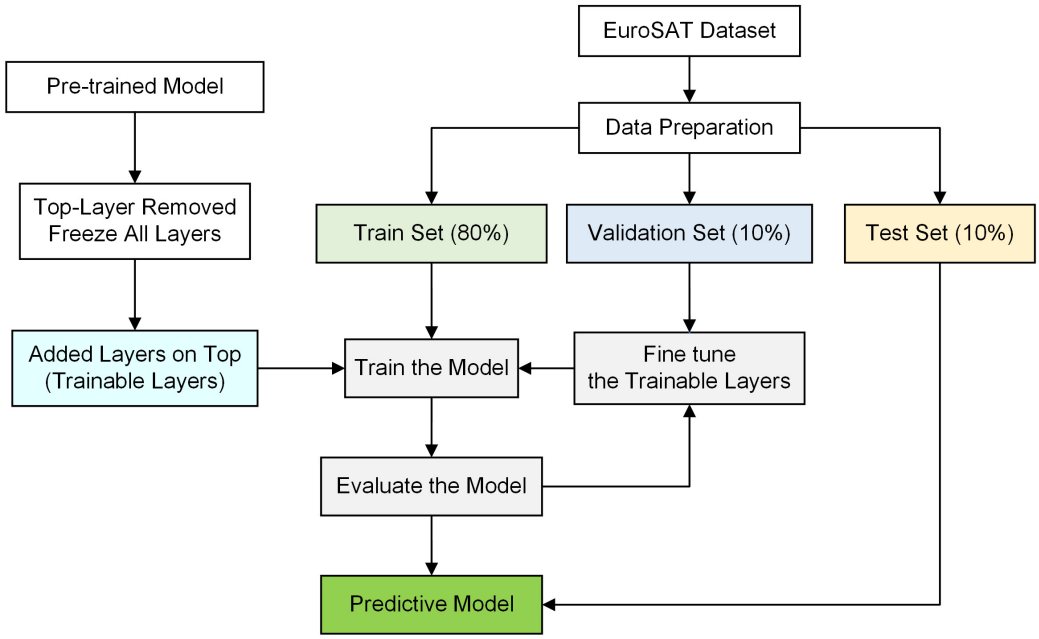

Figure 2. A summary of the process of deep transfer learning for the EuroSAT data.

Table 1. Additional layers structure for various pre-trained models with frozen layers.

Layer Name	Layer Type	Size of Feature Map	Activation Function
Global Avg Pooling 2D	Global Average Pooling 2D	Varies for all models	N/A
Flatten	Flatten	Varies for all models	N/A
Dense	Dense	Varies for all models	relu
Dense_output	Dense	10	softmax

2.2.4. Early Stopping

A regularization method for deep neural networks is known as early stopping which suspends training after a specific number of iterations once the model's accuracy on the validation dataset stops improving. In general, the optimal model weights are maintained and modified throughout training. After a specific number of iterations, when the accuracy metric on the validation set stops improving significantly, training is halted, and the most recent optimal weights are used (see Figure 3). This procedure can decrease overfitting and improve deep neural networks' capacity for adaptability [12].

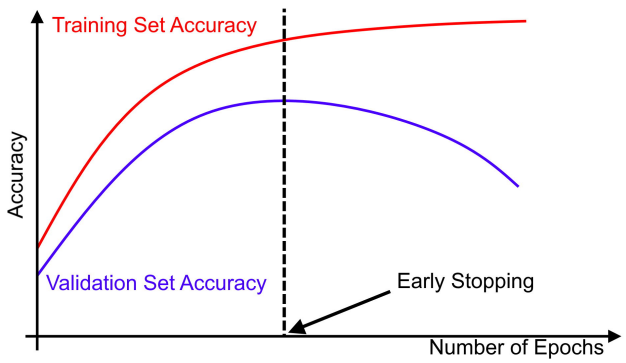

Figure 3. Early stopping: training is terminated when the accuracy on the validation set stops improving.

2.2.5. Categorical Cross-Entropy Loss Function

The cross-entropy loss function is commonly used in classification problems, and its mathematical formulation is expressed as [60]:

$$J(\Theta) = -\frac{1}{p}\sum_{i=1}^{p}\sum_{k=1}^{K} y_k^i \log(\hat{y}_k^i) \quad (1)$$

Here, y_k^i and $\hat{y}_k^i$ represent the true and predicted probabilities that the i-th instance belongs to class k, respectively. K denotes the total number of target classes, p represents the total number of instances, and Θ represents the hypothesis set as a function of weights (filter or kernel) and biases ($\Theta = f(w,\ b)$). It is worth noting that when $K = 2$, Equation (1) is called a binary cross-entropy or a logistic regression cost function, whereas for K, it is referred to as a categorical cross-entropy loss function. The loss function computes the negative log probability of the true class, where it equals zero if the predicted probability of the true class is one, and as the predicted probability approaches zero, the loss function approaches infinity [60]. The sparse categorical cross-entropy loss function from the Tensorflow library [56] was used, which calculates the cross-entropy loss between the labels and predictions [56,61].

2.2.6. Optimizations and Learning Rate

In our models, we used the Adam optimizer, which is a stochastic gradient descent method that employs a flexible estimate of first- and second-order moments [56]. According to Kingma et al. [62], the approach is computationally efficient, necessitates less memory, is not sensitive to gradient diagonal rescaling, and is preferable for enormous data/parameter problems.

Determining the learning rate may be complicated, since a too-small learning rate can lead to major training errors, while an overlarge learning rate can lead to an insufficient set of weights being used too quickly (without achieving the local minima) or an unsteady training process [12,63]. The ReduceLROnPlateau algorithm [56,64] was utilized to mitigate the challenges of selecting the learning rate. When learning becomes stagnant, models usually benefit from declining the learning rate by a factor of 2–10 [12,63]. For our numerical experiments, the ReduceLROnPlateau algorithm reduced the learning rate by 0.2 when there was no progress in the accuracy of the validations set for 3 iterations, and Adam was used as the optimizer, with a minimum learning rate of 0.001.

2.2.7. Accuracy Metrics

A metric of cross-agreement is Cohen's kappa [65,66]. The degree of the agreement formed between two annotators on classification procedures is known as Cohen's kappa, and it is defined as follows [65–67]:

$$\kappa = (p_o - p_e)/(1 - p_e) \quad (2)$$

where p_o is the observed likelihood of concurrence on each sample's label and p_e is the expected concurrence when both annotators assign labels at random. By applying a per-annotator empirical before the class labels, p_e is determined.

Let $\hat{y}_i$ represent the predicted value of the i-th sample, and let y_i be the affiliated true value; then, the fraction of correct predictions over n_{samples} can be expressed as follows [67],

$$\text{accuracy}(y, \hat{y}) = \frac{1}{n_{\text{samples}}}\sum_{i=0}^{n_{\text{samples}}-1} 1(\hat{y}_i = y_i) \quad (3)$$

where $1(x)$ represents the indicator function, which is defined as

$$1_A(x) = \begin{cases} 1 & \text{if } x \in A\,, \\ 0 & \text{if } x \notin A\,. \end{cases} \quad (4)$$

Furthermore, precision, recall, and f1-score accuracy metrics were employed [68,69]. These metrics are defined as follows:

$$\text{Precision} = \frac{TP}{TP + FP}, \tag{5}$$

$$\text{Recall} = \frac{TP}{TP + FN}, \tag{6}$$

$$\text{F1} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}, \tag{7}$$

where TP, TN, FN, and FP represent True Positives, True Negatives, False Negatives, and False Positives, respectively.

3. Results

All models were trained in accordance with the descriptions in Section 2.2. Each model was trained and validated using train and validation sets, respectively. The test set was placed aside for assessments. Then, using the train, validation, and test sets for each model, the cross-entropy loss and accuracy, as defined in Equation (3) were computed, and the average of these three values for each model can be seen in Table 2. ResNet50, EfficientNetV2B0, and ResNet152 demonstrated the highest overall accuracy metrics, as seen in Table 2. Moreover, all models were trained using TensorFlow-GPU v2.10.0 [56] on a Microsoft Windows 11 PC equipped with an AMD Ryzen 9 3900X 12-core processor, 32 GB RAM (DDR4-3600), NVIDIA RTX2080 Super GPU. The training time for each of the models utilized in this study is exhibited in Table 2. As can be observed, mobile networks typically require shorter training time [43]. On our test machine, ResNet152 took nearly three times as long to train as EfficientNetV2B0, while ResNet50 took nearly twice as long as EfficientNetV2B0. According to Tan et al. [42], some of the EfficientNetV2 models were expected to have a lower overall training time. During our experiments, EfficientNetV2 models outperformed EfficientNet models in terms of accuracy and training time.

Cohen's kappa, cross-entropy loss, and accuracy were calculated for the ResNet50, EfficientNetV2B0, and ResNet152 models. Unlike many previous transfer learning models developed on the EuroSAT dataset, our models never encountered the test set throughout the training and validation processes. This provides a key metric for evaluating these models. As seen in Table 3, ResNet50 has the highest accuracy metrics. On the Test set, ResNet50 achieved a kappa score of 0.9641 and an accuracy score of 0.9678.

For ResNet50, EfficientNetV2B0, and ResNet152 models, accuracy, recall, and f1-scores were computed by class and overall, as the average of all classes. Table 4 summarizes these metrics. On the train set, ResNet50 and ResNet152 have slightly higher overall f1-scores than EfficientNetV2B0. However, in the validation and test sets, ResNet50 and EfficientNetV2B0 narrowly outperform ResNet152 in terms of f1-score. Because the f1-score contains both accuracy and recall scores, it is an important metric to consider when evaluating classification performance. As Table 4 indicates, the f1-score on the test set for ResNet50, EfficientNetV2B0, and ResNet152 models are 0.967 ± 0.021, 0.963 ± 0.019, and 0.959 ± 0.024, respectively. ResNet50 achieved an overall f1-score of 0.967 on the test set with the lowest score in the Highway class and the highest in the Sea Lake class. According to Table 4, EfficientNetV2B0 had the highest f1-score in the Annual Crop, Highway, Permanent Crop, and River classes, whereas ResNet50 had the best f1-score in the Forest, Herbaceous Vegetation, Industrial, Pasture, and Sea Lake classes. ResNet152 was only the best in the Residential class.

Table 2. Epochs, training time (s), cross-entropy loss, and accuracy which was averaged over the train, validation, and test sets. Models were trained (separately) on an NVIDIA RTX2080S GPU.

Rank	Model Name	Epoch	Train Time (s)	Loss	Accuracy	Rank	Model Name	Epoch	Train Time (s)	Loss	Accuracy
1	ResNet50	20	644.5	0.091	0.975	21	RegNetX004	26	390.1	0.138	0.954
2	EfficientNetV2B0	20	367.5	0.087	0.972	22	RegNetX006	17	249.2	0.138	0.954
3	ResNet152	17	1265.2	0.087	0.971	23	DenseNet169	11	454.3	0.138	0.953
4	ResNetRS50	19	742.9	0.094	0.969	24	EfficientNetB4	15	784.9	0.139	0.953
5	ResNetRS152	17	1419.3	0.106	0.968	25	ResNetRS350	15	3049.3	0.150	0.951
6	ResNet101	16	858.7	0.103	0.967	26	MobileNet	12	152.7	0.146	0.950
7	RegNetX008	23	423.7	0.120	0.966	27	VGG19	14	658.6	0.156	0.949
8	ResNetRS200	16	1988.2	0.106	0.964	28	ResNet101V2	12	650.6	0.156	0.948
9	ResNetRS270	18	2991.4	0.109	0.964	29	ResNetRS420	14	3430.0	0.163	0.946
10	EfficientNetV2B1	15	340.2	0.116	0.963	30	VGG16	12	463.1	0.165	0.945
11	ResNetRS101	12	728.2	0.121	0.960	31	MobileNetV2	15	229.2	0.169	0.944
12	EfficientNetV2B2	12	308.6	0.132	0.957	32	EfficientNetB5	14	979.9	0.168	0.944
13	RegNetX032	13	534.7	0.125	0.957	33	EfficientNetB7	17	2106.8	0.167	0.943
14	EfficientNetV2B3	13	416.3	0.124	0.957	34	RegNetX002	19	194.4	0.165	0.943
15	EfficientNetB2	14	441.2	0.128	0.957	35	ResNet50V2	11	332.1	0.179	0.942
16	DenseNet201	11	558.2	0.128	0.956	36	EfficientNetB6	22	2054.6	0.172	0.942
17	EfficientNetB0	12	264.6	0.133	0.956	37	ResNet152V2	11	812.1	0.187	0.937
18	DenseNet121	13	419.4	0.133	0.955	38	NASNetMobile	13	324.0	0.224	0.925
19	EfficientNetB1	14	438.6	0.133	0.955	39	InceptionV3	14	278.4	0.256	0.918
20	EfficientNetB3	13	526.0	0.136	0.955						

Table 3. Metrics: train, validation, and test sets.

Model Name	Set	Kappa	Loss	Accuracy
ResNet50	Train	0.9900	0.0304	0.9910
	Validation	0.9629	0.1226	0.9667
	Test	0.9641	0.1189	0.9678
EfficientNetV2B0	Train	0.9789	0.0572	0.9811
	Validation	0.9662	0.0969	0.9696
	Test	0.9600	0.1064	0.9641
ResNet152	Train	0.9864	0.0395	0.9878
	Validation	0.9613	0.1063	0.9652
	Test	0.9559	0.1153	0.9604

The test set confusion matrices for ResNet50, EfficientNetV2B0, and ResNet152 are illustrated in Figure 4. As seen in Figure 4, the Highway class was the most confusing for ResNet50 in the test set, whereas Sea Lake was the least puzzling. EfficientNetV2B0 had the least confusion in classifying the Permanent Crop class, whereas ResNet152 performed the best in the Herbaceous Vegetation, Highway, and Residential classes. ResNet50 excelled in the classifications of Annual Crop, Forest, Industrial, Pasture, and River classes.

A random subset of the test set was sampled, and predictions were made using the ResNet50 model. Figure 5 depicts these predictions, and Table 5 additionally includes the probability for predicted classes. The ResNet50 model erroneously predicted an image from the Pasture class as an image from the Highway class. As shown in Table 5, the image was predicted to be from the Highway class with a probability of 0.946, the Herbaceous Vegetation class with a probability of 0.014, and the Pasture class with a probability of 0.025. As it appears from Figure 5, one potential explanation for this error might be the presence of lines that could be mistaken for roads.

Table 4. Precision and recall by class and overall for (a) ResNet50, (b) EfficientNetV2B0, and (c) ResNet152.

(a) ResNet50									
	Train			**Validation**			**Test**		
Class	**Precision**	**Recall**	**F1-Score**	**Precision**	**Recall**	**F1-Score**	**Precision**	**Recall**	**F1-Score**
Annual Crop	0.996	0.989	0.992	0.987	0.954	0.970	0.967	0.964	0.966
Forest	0.995	0.997	0.996	0.990	0.983	0.986	0.986	0.993	0.989
Herbaceous Vegetation	0.990	0.983	0.986	0.980	0.964	0.972	0.975	0.954	0.964
Highway	0.986	0.982	0.984	0.933	0.925	0.929	0.948	0.916	0.932
Industrial	0.996	0.993	0.995	0.992	0.964	0.978	0.988	0.981	0.984
Pasture	0.981	0.989	0.985	0.951	0.947	0.949	0.974	0.964	0.969
Permanent Crop	0.983	0.989	0.986	0.920	0.982	0.950	0.933	0.950	0.941
Residential	0.996	1.000	0.998	0.980	0.997	0.988	0.987	0.984	0.986
River	0.980	0.988	0.984	0.929	0.943	0.936	0.918	0.966	0.941
Sea Lake	0.999	0.998	0.999	0.987	0.997	0.992	0.990	1.000	0.995
Accuracy	0.991	0.991	0.991	0.967	0.967	0.967	0.968	0.968	0.968
Macro avg	0.990	0.991	0.991	0.965	0.966	0.965	0.967	0.967	0.967
Weighted avg	0.991	0.991	0.991	0.967	0.967	0.967	0.968	0.968	0.968
(b) EfficientNetV2B0									
	Train			**Validation**			**Test**		
Class	**Precision**	**Recall**	**F1-Score**	**Precision**	**Recall**	**F1-Score**	**Precision**	**Recall**	**F1-Score**
Annual Crop	0.981	0.988	0.984	0.973	0.980	0.976	0.989	0.951	0.970
Forest	0.983	0.991	0.987	0.983	0.986	0.984	0.976	0.976	0.976
Herbaceous Vegetation	0.989	0.939	0.964	0.984	0.940	0.961	0.975	0.920	0.946
Highway	0.975	0.972	0.973	0.944	0.944	0.944	0.944	0.933	0.939
Industrial	0.992	0.995	0.993	0.975	0.988	0.981	0.977	0.984	0.980
Pasture	0.965	0.968	0.967	0.966	0.952	0.959	0.948	0.948	0.948
Permanent Crop	0.953	0.983	0.968	0.932	0.982	0.957	0.903	0.984	0.941
Residential	0.992	0.997	0.995	0.993	0.993	0.993	0.990	0.990	0.990
River	0.977	0.981	0.979	0.936	0.933	0.934	0.947	0.955	0.951
Sea Lake	0.994	0.996	0.995	0.993	0.993	0.993	0.977	1.000	0.988
Accuracy	0.981	0.981	0.981	0.970	0.970	0.970	0.964	0.964	0.964
Macro avg	0.980	0.981	0.980	0.968	0.969	0.968	0.962	0.964	0.963
Weighted avg	0.981	0.981	0.981	0.970	0.970	0.970	0.965	0.964	0.964
(c) ResNet152									
	Train			**Validation**			**Test**		
Class	**Precision**	**Recall**	**F1-Score**	**Precision**	**Recall**	**F1-Score**	**Precision**	**Recall**	**F1-Score**
Annual Crop	0.989	0.987	0.988	0.956	0.966	0.961	0.953	0.943	0.948
Forest	0.994	0.996	0.995	0.976	0.989	0.983	0.990	0.979	0.984
Herbaceous Vegetation	0.981	0.984	0.983	0.967	0.964	0.966	0.950	0.962	0.956
Highway	0.978	0.981	0.979	0.937	0.915	0.925	0.933	0.940	0.936
Industrial	0.997	0.984	0.990	0.992	0.972	0.982	0.988	0.966	0.977
Pasture	0.991	0.983	0.987	0.956	0.952	0.954	0.932	0.962	0.947
Permanent Crop	0.979	0.977	0.978	0.958	0.958	0.958	0.929	0.939	0.934
Residential	0.988	0.999	0.993	0.983	0.997	0.990	0.987	0.997	0.992
River	0.986	0.981	0.984	0.936	0.936	0.936	0.943	0.906	0.924
Sea Lake	0.994	0.999	0.996	0.984	0.990	0.987	0.983	0.997	0.990
Accuracy	0.988	0.988	0.988	0.965	0.965	0.965	0.960	0.960	0.960
Macro avg	0.988	0.987	0.987	0.964	0.964	0.964	0.959	0.959	0.959
Weighted avg	0.988	0.988	0.988	0.965	0.965	0.965	0.960	0.960	0.960

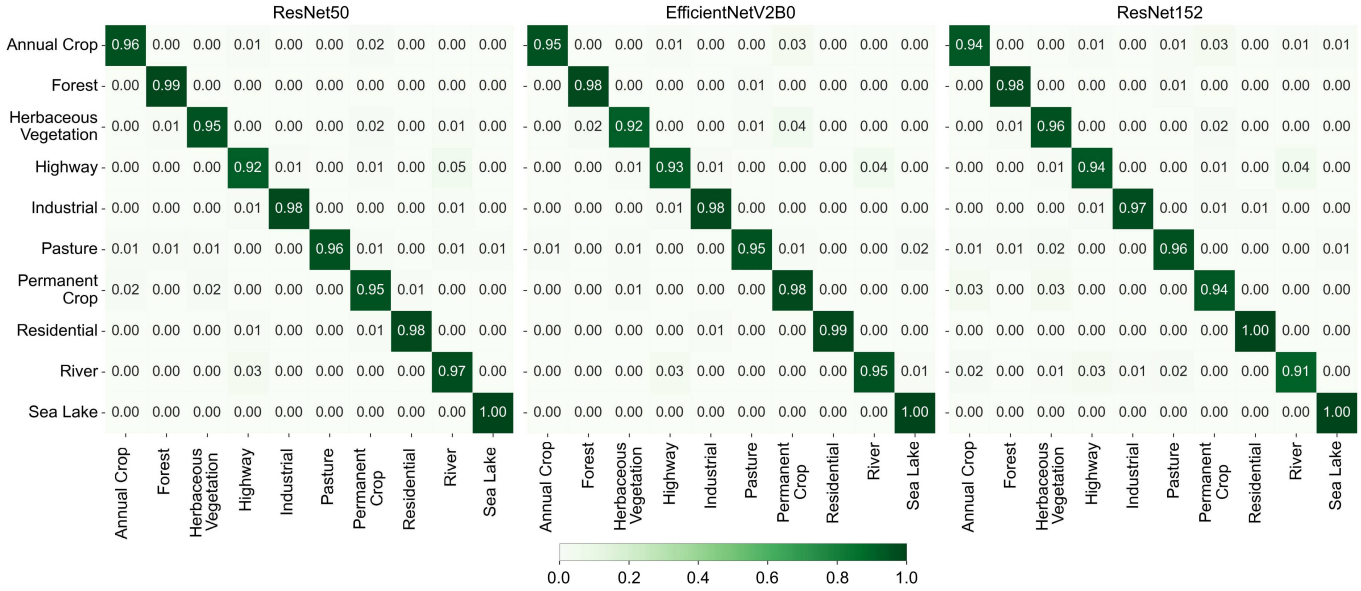

Figure 4. Normalized confusion matrix for ResNet50 (**left**), EfficientNetV2B0 (**center**), and ResNet152 (**right**) on the test set.

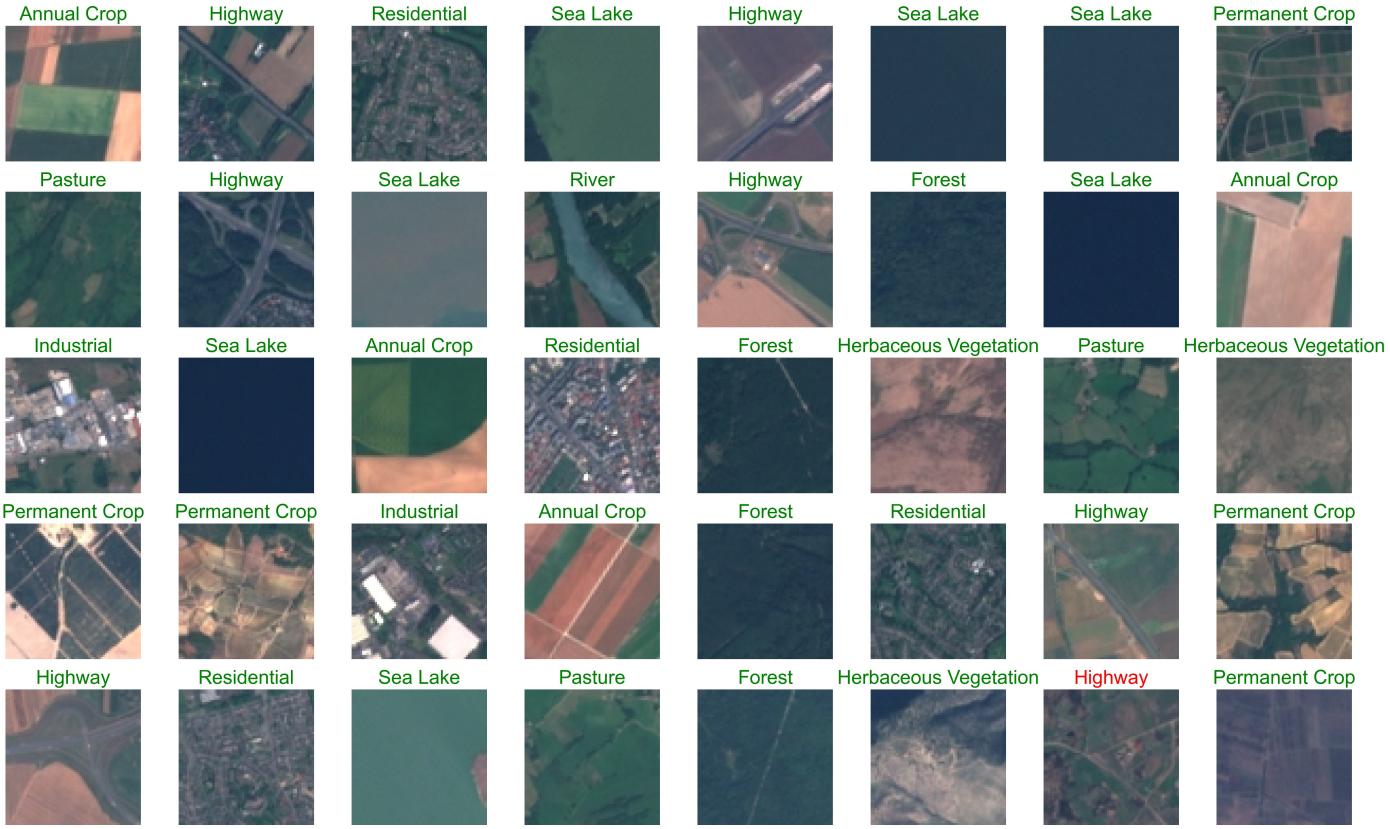

Figure 5. ResNet50 is used to predict a sample from the test set. Correctly predicted labels are displayed in green text, whereas incorrectly predicted labels are indicated in red text. The Pasture class was incorrectly identified as the Highway class.

Table 5. Predicted probability for a sample subset of the test set using ResNet50. The values were rounded off to four decimal places, and if 0.0000 or 1.0000 appeared, they were, respectively, simplified as 0 and 1. The highlighted row indicates that an image from the Pasture class was incorrectly classified as an image from the Highway class.

	Predicted Probability									
Actual Class	**Annual Crop**	**Forest**	**Herbaceous Vegetation**	**Highway**	**Industrial**	**Pasture**	**Permanent Crop**	**Residential**	**River**	**Sea Lake**
Annual Crop	1	0	0	0	0	0	0	0	0	0
Highway	0	0	0	1	0	0	0	0	0	0
Residential	0	0	0	0	0	0	0	1	0	0
Sea Lake	0	0	0	0	0	0.0633	0	0	0	0.9367
Highway	0.0050	0	0	0.9709	0.0140	0	0	0	0.0101	0
Sea Lake	0	0	0	0	0	0	0	0	0	1
Sea Lake	0	0	0	0	0	0	0	0	0	1
Permanent Crop	0	0	0	0.0411	0	0	0.9589	0	0	0
Pasture	0	0.0036	0.0026	0.0001	0	0.9914	0	0	0.0022	0
Highway	0	0	0	1	0	0	0	0	0	0
Sea Lake	0	0	0	0	0	0	0	0	0	1
River	0	0	0	0.0035	0	0	0	0	0.9965	0
Highway	0	0	0	1	0	0	0	0	0	0
Forest	0	1	0	0	0	0	0	0	0	0
Sea Lake	0	0	0	0	0	0	0	0	0	1
Annual Crop	1	0	0	0	0	0	0	0	0	0
Industrial	0	0	0	0	1	0	0	0	0	0
Sea Lake	0	0	0	0	0	0	0	0	0	1
Annual Crop	1	0	0	0	0	0	0	0	0	0
Residential	0	0	0	0	0	0	0	1	0	0
Forest	0	0.9988	0.0004	0.0003	0	0.0005	0	0	0	0
Herbaceous Vegetation	0	0	0.9998	0	0	0	0.0002	0	0	0
Pasture	0	0	0	0	0	1	0	0	0	0
Herbaceous Vegetation	0	0	0.9998	0	0	0.0001	0.0001	0	0	0
Permanent Crop	0	0	0	0	0	0	1	0	0	0
Permanent Crop	0	0	0	0	0	0	1	0	0	0
Industrial	0	0	0	0	1	0	0	0	0	0
Annual Crop	1	0	0	0	0	0	0	0	0	0
Forest	0	1	0	0	0	0	0	0	0	0
Residential	0	0	0	0	0	0	0	1	0	0
Highway	0	0	0	0.9996	0	0	0	0	0.0004	0
Permanent Crop	0	0	0	0	0	0	1	0	0	0
Highway	0	0	0	1	0	0	0	0	0	0
Residential	0	0	0	0	0	0	0	1	0	0
Sea Lake	0	0	0	0	0	0	0	0	0	1
Pasture	0	0	0	0	0	1	0	0	0	0
Forest	0	0.9934	0.0066	0	0	0	0	0	0	0
Herbaceous Vegetation	0	0	1	0	0	0	0	0	0	0
Pasture	0	0	0.0143	0.9465	0.0008	0.0247	0.0047	0.0090	0	0
Permanent Crop	0.0120	0	0.0196	0.0002	0.0001	0.0039	0.9642	0	0	0

4. Discussion

In our testing, MobileNet achieved a high accuracy score of 0.9500 within just 152.71 s of training on our machine, making it the fastest among the 39 models evaluated. However, it ranked only 26th in terms of accuracy. MobileNet is a lightweight deep neural network intentionally designed with fewer parameters than other models, yet it still maintains high accuracy in image classification tasks. This feature has made it a popular choice for real-time image recognition applications on devices with limited computational resources, such as mobile phones or embedded systems. The architecture of MobileNet includes depthwise separable convolutions, which reduce the computational cost of convolutions used during the classification process. This feature allows the network to process large amounts of data efficiently while maintaining high accuracy. While ResNet50 had the highest overall accuracy score, it ranked 23rd in terms of training time. ResNet152 achieved the third highest accuracy score, but it ranked 32nd in training time. Both models belong to the ResNet family of deep neural networks. ResNet50 has 50 convolutional layers and the ability to learn intricate patterns while overcoming vanishing gradients. On the other hand, ResNet152 has 152 layers and uses skip connections to directly transmit information between layers, solving the vanishing gradient problem during deep neural network training. EfficientNetV2B0 achieved a high accuracy score of 0.9716, ranking second overall, but it required 2.4 times longer training time than MobileNet. EfficientNetV2B0 is an upgraded version of the original EfficientNet model, which is designed to be computationally efficient while maintaining high accuracy in image classification tasks. Overall, the EfficientNetV2 and RegNet models showed relatively high kappa values in relation to training time compared to the other 39 models evaluated.

Table 3 provided Cohen's kappa, cross-entropy loss, and accuracy metrics for the ResNet50, EfficientNetV2B0, and ResNet152 models. Among these models, the maximum difference in kappa scores on the test set was 0.0082, and the maximum difference in accuracy scores was 0.0074. However, since the EuroSAT dataset is not well balanced across all categories, the f1-score might be a better performance indicator than accuracy [70]. According to Table 4, EfficientNetV2B0 achieved the highest f1-scores in the Annual Crop, Highway, Permanent Crop, and River categories. ResNet50 performed best in the Forest, Herbaceous Vegetation, Industrial, Pasture, and Sea Lake classes, while ResNet152 achieved the highest f1-score only in the Residential category. Moreover, since EfficientNetV2B0 required less training time than ResNet50, it achieved impressive f1 scores.

The performance of the models in different categories was further validated by the confusion matrices on the test set. The results were consistent with the f1-scores reported in Table 4. For instance, EfficientNetV2B0 showed the least confusion in classifying the Permanent Crop category, while ResNet152 performed best in the Herbaceous Vegetation, Highway, and Residential classes. ResNet50 exhibited strong performance in the Annual Crop, Forest, Industrial, Pasture, and River categories.

By splitting the EuroSAT dataset into train and test sets with 80% and 20% distributions, respectively, Li et al. [35] obtained an overall accuracy of 0.9874. Helber et al. [31] achieved a total accuracy of 0.9857 by dividing the EuroSAT dataset into a training (80%) set and a test (20%) set. Naushad et al. [12] segmented the EuroSAT into two sets, the train set (75%) and the test set (25%), which was used for both validations and testing, and obtained an overall accuracy of 0.9917. However, we divided the EuroSAT dataset into three subsets, train (80%), validation (10%), and test (10%), and kept the test set unseen by models for testing purposes. Our best overall accuracy was obtained using ResNet50, 0.9751, which is the lowest of any of the research listed. The variation in overall accuracy might be attributed to how we distributed the dataset into three subsets instead of two sets with one of them, the test set, not being involved during the training and validation processes. Our total accuracy was calculated by averaging the accuracy scores from the train, validation, and test sets. Our test set was at least 50% smaller than any of the aforementioned research, which could contribute to this slightly lower overall accuracy level. On the other hand, our test set which was independent of the validation set played a substantial role in our analysis. In

several of the previous studies, it was unclear which set was utilized for validation, as their dataset was only divided into train and test sets. The test set should ideally be a dataset obtained separately. The presence of a large number of independent test sets validates the generalization argument [39]. Nevertheless, the key objective of this study was to evaluate deep transfer learning models in terms of accuracy metrics and training time, and the present distribution of train, validation, and test sets was paramount for the analyses.

While we were modeling, we faced a hurdle due to the insufficient quantity of labeled data available in the EuroSAT datasets. Specifically, there were only 27,000 labeled images available for 10 classes. To address this challenge, we utilized data augmentation techniques to create additional data samples by introducing variations to the existing data samples. These variations were generated with the aid of algorithms that could automatically create new data samples.

Deep transfer learning presents a broad range of models suitable for training in various applications. Although some pre-trained models achieved exceptional accuracy metrics in other domains through deep transfer learning, they did not rank among our top models for LULC classifications. The choice of the most appropriate model for a specific application hinges on various factors. For example, the size of the dataset and the complexity of the problem are key considerations when selecting a model. In addition, the availability of computational resources, such as GPU capacity, can also affect the model selection process. Ultimately, the ideal model for a given application should strike a balance between performance and practical constraints. After conducting our research, we propose that EfficientNetV2 is the optimal choice for transfer learning-based classification of LULC image classifications. Nonetheless, if there are constraints on computational resources, MobileNet could be a viable alternative.

5. Conclusions

In this article, we leveraged Tensorflow's pre-trained models to develop 39 deep transfer learning models and compare their accuracy and training time. The EuroSAT was divided into three subgroups (80% for training, 10% for validation, and 10% for testing). ResNet50, EfficientNetV2B0, and ResNet152 models were the best three in terms of kappa and accuracy scores among our 39 deep transfer learning models. ResNet152 took about three times as long to train as EfficientNetV2B0 on our test computer, while ResNet50 took roughly twice as long. On the test set, ResNet50 obtained an overall f1-score of 0.967, with the lowest score in the Highway class and the greatest in the Sea Lake class. In the Annual Crop, Highway, Permanent Crop, and River classes, EfficientNetV2B0 had the highest f1-score, while ResNet50 had the best f1-score in the Forest, Herbaceous Vegetation, Industrial, Pasture, and Sea Lake classes. ResNet152 had undoubtedly the strongest f1-score in the Residential class. The Highway class was the most confusing for ResNet50 in the test set, while Sea Lake was the least confusing. EfficientNetV2B0 performed best in the Permanent Crop class, whereas ResNet152 performed best in the Herbaceous Vegetation, Highway, and Residential classes. ResNet50 excelled in the Annual Crop, Forest, Industrial, Pasture, and River categories. Based on these results, we suggest that EfficientNetV2 is the optimal choice for transfer learning-based land cover and land use image classification.

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

Funding: This study was partially funded by a Discovery Grant from the Natural Sciences and Engineering Research Council of Canada to Q.K.H.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data used in this research are available in the public domain.

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

References

1. Dastour, H.; Ghaderpour, E.; Zaghloul, M.S.; Farjad, B.; Gupta, A.; Eum, H.; Achari, G.; Hassan, Q.K. Wavelet-based spatiotemporal analyses of climate and vegetation for the Athabasca river basin in Canada. *Int. J. Appl. Earth Obs. Geoinf.* **2022**, *114*, 103044. [CrossRef]
2. Afrin, S.; Gupta, A.; Farjad, B.; Ahmed, M.R.; Achari, G.; Hassan, Q.K. Development of land-use/land-cover maps using Landsat-8 and MODIS data, and their integration for hydro-ecological applications. *Sensors* **2019**, *19*, 4891. [CrossRef] [PubMed]
3. Akbar, T.A.; Hassan, Q.K.; Achari, G. Clusterization of Surface Water Quality and Its Relation to Climate and Land Use/Cover. *J. Environ. Prot.* **2013**, *4*, 333–343. [CrossRef]
4. Abdullah, A.Y.M.; Masrur, A.; Adnan, M.S.G.; Baky, M.; Al, A.; Hassan, Q.K.; Dewan, A. Spatio-temporal patterns of land use/land cover change in the heterogeneous coastal region of Bangladesh between 1990 and 2017. *Remote Sens.* **2019**, *11*, 790. [CrossRef]
5. Liu, X.; He, J.; Yao, Y.; Zhang, J.; Liang, H.; Wang, H.; Hong, Y. Classifying urban land use by integrating remote sensing and social media data. *Int. J. Geogr. Inf. Sci.* **2017**, *31*, 1675–1696. [CrossRef]
6. Huang, Z.; Dumitru, C.O.; Pan, Z.; Lei, B.; Datcu, M. Classification of large-scale high-resolution SAR images with deep transfer learning. *IEEE Geosci. Remote Sens. Lett.* **2020**, *18*, 107–111. [CrossRef]
7. Mc Cutchan, M.; Comber, A.J.; Giannopoulos, I.; Canestrini, M. Semantic boosting: Enhancing deep learning based LULC classification. *Remote Sens.* **2021**, *13*, 3197. [CrossRef]
8. Phan, T.N.; Kuch, V.; Lehnert, L.W. Land cover classification using Google Earth Engine and random forest classifier—The role of image composition. *Remote Sens.* **2020**, *12*, 2411. [CrossRef]
9. Moharram, M.A.; Sundaram, D.M. Land Use and Land Cover Classification with Hyperspectral Data: A comprehensive review of methods, challenges and future directions. *Neurocomputing* **2023**, *536*, 90–113. [CrossRef]
10. Zhang, J.; Wang, H.; Wang, Y.; Zhou, Q.; Li, Y. Deep network based on up and down blocks using wavelet transform and successive multi-scale spatial attention for cloud detection. *Remote Sens. Environ.* **2021**, *261*, 112483. [CrossRef]
11. Gómez, P.; Meoni, G. MSMatch: Semisupervised Multispectral Scene Classification With Few Labels. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2021**, *14*, 11643–11654. [CrossRef]
12. Naushad, R.; Kaur, T.; Ghaderpour, E. Deep transfer learning for land use and land cover classification: A comparative study. *Sensors* **2021**, *21*, 8083. [CrossRef]
13. Günen, M.A. Performance comparison of deep learning and machine learning methods in determining wetland water areas using EuroSAT dataset. *Environ. Sci. Pollut. Res.* **2022**, *29*, 21092–21106. [CrossRef]
14. Qi, K.; Wu, H.; Shen, C.; Gong, J. Land-use scene classification in high-resolution remote sensing images using improved correlatons. *IEEE Geosci. Remote Sens. Lett.* **2015**, *12*, 2403–2407.
15. Marmanis, D.; Datcu, M.; Esch, T.; Stilla, U. Deep learning earth observation classification using ImageNet pretrained networks. *IEEE Geosci. Remote Sens. Lett.* **2015**, *13*, 105–109. [CrossRef]
16. Das, A.; Giri, R.; Chourasia, G.; Bala, A.A. Classification of retinal diseases using transfer learning approach. In Proceedings of the 2019 International Conference on Communication and Electronics Systems (ICCES), Coimbatore, India, 17–19 July 2019; pp. 2080–2084.
17. Pan, S.J.; Yang, Q. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.* **2009**, *22*, 1345–1359. [CrossRef]
18. Tan, C.; Sun, F.; Kong, T.; Zhang, W.; Yang, C.; Liu, C. A survey on deep transfer learning. In Proceedings of the International Conference on Artificial Neural Networks, Rhodes, Greece, 4–7 October 2018; pp. 270–279.
19. Zhuang, F.; Qi, Z.; Duan, K.; Xi, D.; Zhu, Y.; Zhu, H.; Xiong, H.; He, Q. A comprehensive survey on transfer learning. *Proc. IEEE* **2020**, *109*, 43–76. [CrossRef]
20. Alem, A.; Kumar, S. Transfer learning models for land cover and land use classification in remote sensing image. *Appl. Artif. Intell.* **2022**, *36*, 2014192. [CrossRef]
21. Tseng, H.H.; Yang, M.D.; Saminathan, R.; Hsu, Y.C.; Yang, C.Y.; Wu, D.H. Rice Seedling Detection in UAV Images Using Transfer Learning and Machine Learning. *Remote Sens.* **2022**, *14*, 2837. [CrossRef]
22. Chen, J.; Sun, J.; Li, Y.; Hou, C. Object detection in remote sensing images based on deep transfer learning. *Multimed. Tools Appl.* **2022**, *81*, 12093–12109.
23. Hilal, A.M.; Al-Wesabi, F.N.; Alzahrani, K.J.; Al Duhayyim, M.; Ahmed Hamza, M.; Rizwanullah, M.; García Díaz, V. Deep transfer learning based fusion model for environmental remote sensing image classification model. *Eur. J. Remote Sens.* **2022**, *55*, 12–23. [CrossRef]
24. Song, H.; Yang, W. GSCCTL: A general semi-supervised scene classification method for remote sensing images based on clustering and transfer learning. *Int. J. Remote Sens.* **2022**, *43*, 5976–6000. [CrossRef]
25. Dumitru, C.O.; Schwarz, G.; Datcu, M. Land cover semantic annotation derived from high-resolution SAR images. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2016**, *9*, 2215–2232. [CrossRef]

26. Li, W.; Wang, Z.; Wang, Y.; Wu, J.; Wang, J.; Jia, Y.; Gui, G. Classification of high-spatial-resolution remote sensing scenes method using transfer learning and deep convolutional neural network. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2020**, *13*, 1986–1995. [CrossRef]
27. Zhong, Y.; Han, X.; Zhang, L. Multi-class geospatial object detection based on a position-sensitive balancing framework for high spatial resolution remote sensing imagery. *ISPRS J. Photogramm. Remote Sens.* **2018**, *138*, 281–294. [CrossRef]
28. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
29. 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.
30. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the inception architecture for computer vision. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 2818–2826.
31. 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]
32. Helber, P. EuroSAT: Land Use and Land Cover Classification with Sentinel-2. 2019. Available online: https://github.com/phelber/EuroSAT (accessed 1 October 2022).
33. 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; pp. 204–207.
34. Al-Qizwini, M.; Barjasteh, I.; Al-Qassab, H.; Radha, H. Deep learning algorithm for autonomous driving using GoogLeNet. In Proceedings of the 2017 IEEE Intelligent Vehicles Symposium (IV), Los Angeles, CA, USA, 11–14 June 2017; pp. 89–96.
35. Li, J.; Lin, D.; Wang, Y.; Xu, G.; Zhang, Y.; Ding, C.; Zhou, Y. Deep discriminative representation learning with attention map for scene classification. *Remote Sens.* **2020**, *12*, 1366. [CrossRef]
36. Yassine, H.; Tout, K.; Jaber, M. Improving LULC Classification from Satellite Imagery Using Deep Learning–EuroSAT Dataset. *ISPRS-Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2021**, *43*, 369–376. [CrossRef]
37. Mikołajczyk, A.; Grochowski, M. Data augmentation for improving deep learning in image classification problem. In Proceedings of the 2018 International Interdisciplinary PhD Workshop (IIPhDW), Swinoujscie, Poland, 9–12 May 2018; pp. 117–122.
38. Shorten, C.; Khoshgoftaar, T.M. A survey on image data augmentation for deep learning. *J. Big Data* **2019**, *6*, 1–48.
39. Eelbode, T.; Sinonquel, P.; Maes, F.; Bisschops, R. Pitfalls in training and validation of deep learning systems. *Best Pract. Res. Clin. Gastroenterol.* **2021**, *52*, 101712. [CrossRef]
40. Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely connected convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA , 21–26 July 2017; pp. 4700–4708.
41. Tan, M.; Le, Q. EfficientNet: Rethinking model scaling for convolutional neural networks. In Proceedings of the International Conference on Machine Learning, PMLR, Long Beach, CA, USA, 9–15 June 2019; pp. 6105–6114.
42. Tan, M.; Le, Q. EfficientNetV2: Smaller models and faster training. In Proceedings of the 38th International Conference on Machine Learning, Virtual, 18–24 July 2021; pp. 10096–10106.
43. 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.
44. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.C. MobileNetV2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 4510–4520.
45. Radosavovic, I.; Kosaraju, R.P.; Girshick, R.; He, K.; Dollár, P. Designing network design spaces. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 10428–10436.
46. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet classification with deep convolutional neural networks. *Commun. ACM* **2017**, *60*, 84–90. [CrossRef]
47. Shaha, M.; Pawar, M. Transfer learning for image classification. In Proceedings of the 2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA), Coimbatore, India, 29–31 March 2018; pp. 656–660.
48. Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. ImageNet Large Scale Visual Recognition Challenge. *Int. J. Comput. Vis. (IJCV)* **2015**, *115*, 211–252. [CrossRef]
49. Deng, J.; Dong, W.; Socher, R.; Li, L.J.; 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.
50. Bello, I.; Fedus, W.; Du, X.; Cubuk, E.D.; Srinivas, A.; Lin, T.Y.; Shlens, J.; Zoph, B. Revisiting ResNets: Improved training and scaling strategies. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 22614–22627.
51. Gong, Z.; Zhong, P.; Yu, Y.; Hu, W. Diversity-promoting deep structural metric learning for remote sensing scene classification. *IEEE Trans. Geosci. Remote Sens.* **2017**, *56*, 371–390. [CrossRef]
52. Agarwal, A.; Vatsa, M.; Singh, R.; Ratha, N. Intelligent and adaptive mixup technique for adversarial robustness. In Proceedings of the 2021 IEEE International Conference on Image Processing (ICIP), Anchorage, AK, USA, 19–22 September 2021; pp. 824–828.
53. Weiss, K.; Khoshgoftaar, T.M.; Wang, D. A survey of transfer learning. *J. Big Data* **2016**, *3*, 1–40. [CrossRef]
54. Zoph, B.; Vasudevan, V.; Shlens, J.; Le, Q.V. Learning transferable architectures for scalable image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 8697–8710.

55. He, K.; Zhang, X.; Ren, S.; Sun, J. Identity mappings in deep residual networks. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; pp. 630–645.
56. Abadi, M.; Barham, P.; Chen, J.; Chen, Z.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Irving, G.; Isard, M.; et al. TensorFlow: A system for Large-Scale machine learning. In Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16), Savannah, GA, USA, 2–4 November 2016; pp. 265–283.
57. Hsiao, T.Y.; Chang, Y.C.; Chou, H.H.; Chiu, C.T. Filter-based deep-compression with global average pooling for convolutional networks. *J. Syst. Archit.* **2019**, *95*, 9–18.
58. Li, Y.; Yuan, Y. Convergence analysis of two-layer neural networks with ReLU activation. In Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA, 4–9 December 2017.
59. Dunne, R.A.; Campbell, N.A. On the pairing of the softmax activation and cross-entropy penalty functions and the derivation of the softmax activation function. In Proceedings of the 8th Australasian Conference on Neural Networks, Melbourne, Australia, 3–4 April 1997; Volume 181, p. 185.
60. Sattarifar, A.; Nestorović, T. Damage localization and characterization using one-dimensional convolutional neural network and a sparse network of transducers. *Eng. Appl. Artif. Intell.* **2022**, *115*, 105273. [CrossRef]
61. Milanova, I.; Sarvanoska, K.; Srbinoski, V.; Gjoreski, H. Automatic text generation in Macedonian using recurrent neural networks. In Proceedings of the International Conference on ICT Innovations, Ohrid, North Macedonia, 17–19 October 2019; pp. 1–12.
62. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
63. Yu, X.H.; Chen, G.A.; Cheng, S.X. Dynamic learning rate optimization of the backpropagation algorithm. *IEEE Trans. Neural Netw.* **1995**, *6*, 669–677. [PubMed]
64. Zaheer, M.; Reddi, S.; Sachan, D.; Kale, S.; Kumar, S. Adaptive methods for nonconvex optimization. In Proceedings of the 32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, QC, Canada, 3–8 December 2018.
65. Cohen, J. A coefficient of agreement for nominal scales. *Educ. Psychol. Meas.* **1960**, *20*, 37–46. [CrossRef]
66. Artstein, R.; Poesio, M. Inter-coder agreement for computational linguistics. *Comput. Linguist.* **2008**, *34*, 555–596. [CrossRef]
67. Buitinck, L.; Louppe, G.; Blondel, M.; Pedregosa, F.; Mueller, A.; Grisel, O.; Niculae, V.; Prettenhofer, P.; Gramfort, A.; Grobler, J.; et al. API design for machine learning software: Experiences from the scikit-learn project. In Proceedings of the ECML PKDD Workshop: Languages for Data Mining and Machine Learning, Prague, Czech Republic, 23–27 September 2013; pp. 108–122.
68. Warrens, M.J. Cohen's kappa is a weighted average. *Stat. Methodol.* **2011**, *8*, 473–484. [CrossRef]
69. Powers, D. Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation. *J. Mach. Learn. Technol.* **2011**, *2*, 37–63.
70. Zou, Q.; Xie, S.; Lin, Z.; Wu, M.; Ju, Y. Finding the best classification threshold in imbalanced classification. *Big Data Res.* **2016**, *5*, 2–8. [CrossRef]