Lightweight Remote Sensing Scene Classification on Edge Devices via Knowledge Distillation and Early-exit

Yang Zhao
Harbin Institute of Technology, Shenzhen
Shenzhen, Guangdong, China
yang.zhao@hit.edu.cn

Shusheng Li
Harbin Institute of Technology, Shenzhen
Shenzhen, Guangdong, China
lss11@foxmail.com

Xueshang Feng
Harbin Institute of Technology, Shenzhen
Shenzhen, Guangdong, China
fengxueshang@hit.edu.cn

Abstract

As the development of lightweight deep learning algorithms, various deep neural network (DNN) models have been proposed for the remote sensing scene classification (RSSC) application. However, it is still challenging for these RSSC models to achieve optimal performance among model accuracy, inference latency, and energy consumption on resource-constrained edge devices. In this paper, we propose a lightweight RSSC framework, which includes a distilled global filter network (GFNet) model and an early-exit mechanism designed for edge devices to achieve state-of-the-art performance. Specifically, we first apply frequency domain distillation on the GFNet model to reduce model size. Then we design a dynamic early-exit model tailored for DNN models on edge devices to further improve model inference efficiency. We evaluate our E3C model on three edge devices across four datasets. Extensive experimental results show that it achieves an average of 1.3x speedup on model inference and over 40% improvement on energy efficiency, while maintaining high classification accuracy.

CCS Concepts

• **Computing methodologies** → **Object recognition**; • **Computer systems organization** → *Embedded systems.*

Keywords

Remote sensing scene classification; early-exit; knowledge distillation; edge computing; DNN inference

ACM Reference Format:
Yang Zhao, Shusheng Li, and Xueshang Feng. 2025. Lightweight Remote Sensing Scene Classification on Edge Devices via Knowledge Distillation and Early-exit. In *Proceedings of the 33rd ACM International Conference on Multimedia (MM '25), October 27–31, 2025, Dublin, Ireland*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3746027.3755609

1 Introduction

As the development of deep neural network (DNN) and embedded system techniques, remote sensing scene classification (RSSC) on edge devices becomes more and more practical for land resource management, urban planning, traffic flow prediction and many other remote sensing applications [2, 4, 26]. For traditional RSSC, remote sensing imagery data are first downloaded from the sensor sources, such as RGB-infrared cameras on unmanned aerial vehicles (UAVs) or hyperspectral sensors on satellites. Then they are pre-processed and fed into DNN models, which are performed on high-performance servers [4]. However, the resolution and volume of remote sensing imagery data have been rapidly increasing as the development of sensor technology. It will create overwhelming pressure on data communication systems if large amounts of imagery data are transferred to a central server, especially for real-time applications [5, 12, 24]. Thus, it becomes a promising solution to perform DNN model inference on edge devices near data sources, without suffering from the communication bottleneck issue [7, 11, 29].

However, edge devices usually has limited resources, and embedded systems such as UAVs have strict requirements on size, weight and power consumption [20]. Although vision Transformers (ViT) and other DNN models can provide outstanding classification accuracy for RSSC [31], these models require substantial model parameters and floating-point operations (FLOPs) [33], and thus have high inference latency and energy consumption on edge devices [26, 34]. To address this challenge, recent studies have developed various lightweight deep learning techniques, such as knowledge distillation, to reduce model complexity and accelerate model inference [15, 20, 26, 34]. For example, the data-efficient image Transformer (DeiT) model introduces a token-based distillation strategy that significantly outperforms vanilla distillation methods [23]. Early-exiting dynamic neural networks accelerate DNN inference by allowing a model to make predictions from intermediate layers [19, 22]. On the model architecture side, a simple yet computationally efficient architecture called Global Filter Network (GFNet) replaces the self-attention sub-layer in vision Transformers (ViT) with 2D discrete Fourier transform modules and global filter modules that enjoy log-linear computation complexity [18]. We perform experiments to train these lightweight models, deploy and evaluate them on various edge devices, as shown in Fig. 1. However, we find the following issues, when comparing the performance of the GFNet model and vision Transformer models [8, 23].

First, we find that none of the GFNet, DeiT and ViT models can achieve low latency, e.g., <500 milliseconds, using popular RSSC datasets on resource-constrained edge devices. One example is shown in Fig. 2(c): the GFNet and ViT models exhibit high inference latency, exceeding one second across all three datasets on the resource-constrained Raspberry Pi device. We also find that although GFNet is more computationally efficient than ViT, a ViT model combined with early-exit mechanisms achieves even lower latency than GFNet on certain edge devices, since model inference can be completed early by exiting from an intermediate layer of a more sophisticated DNN model. Can we apply lightweight techniques

MM '25, Dublin, Ireland

ACM ISBN 979-8-4007-2035-2/2025/10
https://doi.org/10.1145/3746027.3755609

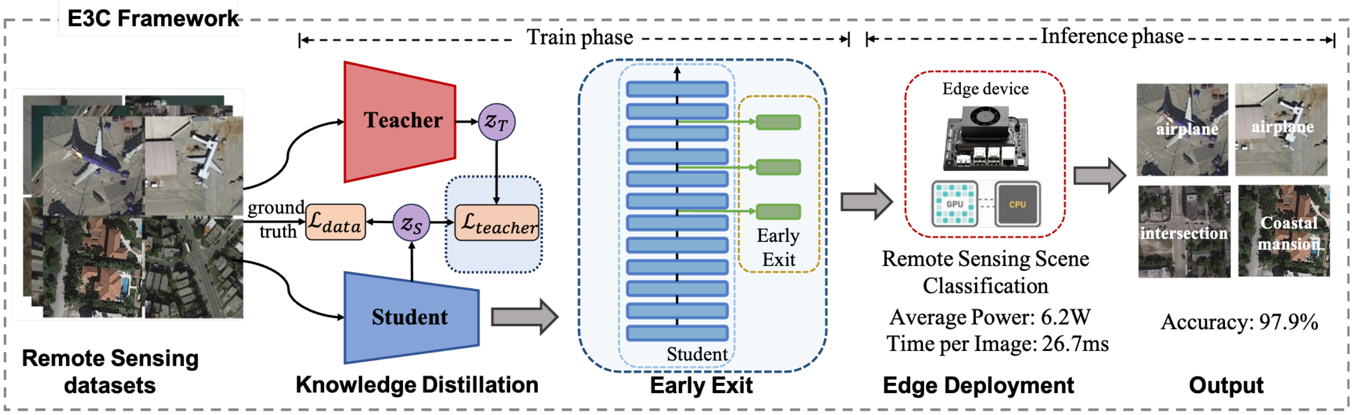

Figure 1: Overview of the E3C Framework with knowledge distillation and early-exit tailored for RSSC on edge devices.

such as knowledge distillation and early-exit to a computationally efficient DNN model other than Transformers to further improve the performance of RSSC on edge devices? We perform the following investigations to find out the answer. First, on the knowledge distillation side, we adopt the hard-label distillation approach as in [23] to apply frequency domain distillation on the GFNet model. Specifically, we introduce a distillation token that captures global information by interacting with other tokens through the global filter layer in GFNet [18]. We find that the model size and inference latency of the distilled GFNet model can be significantly reduced.

Second, on the early-exit side, we have investigated directly applying the early-exit mechanisms used in JEI-DNN [19] to the GFNet model to further improve the RSSC performance. However, we find that applying the JET-DNN early-exit method to GFNet does not improve classification accuracy and energy consumption, and even increases latency on certain edge devices. After analyzing our data, it turns out that since we use a relatively lightweight GFNet model as the backbone model, the computational overhead introduced by gating mechanisms (GMs) and intermediate inference modules (IMs) as in [19] outweighs the benefits provided by the early-exit mechanisms, especially when some inputs are not "easy enough" to be determined by early classifiers. The JEI-DNN method optimizes a loss that jointly assesses accuracy and inference cost, but it also adds early-exit branches with GMs and IMs modules to each and every layer of the backbone model [19], which can cause additional computation overhead. Thus, we propose a new early-exit method tailored for lightweight models on edge devices, in which we start incorporating the early-exit branches from an intermediate layer every other layer of the backbone model. Since the shallow layers of a lightweight model, e.g., GFNet, usually cannot capture image features effectively, significant computational overhead can be saved by using this new method. For edge devices with heterogeneous processors, we propose to use CPUs to compute early-exit model parameters, while loading other model parameters on GPUs, to better utilize heterogeneous resources.

Finally, we design and implement a remote sensing scene classification framework called E3C, which includes knowledge distillation and early-exit mechanisms tailored for the GFNet model on edge devices. We perform extensive evaluations across four RSSC datasets on three different edge devices. Experimental results show that compared with state-of-the-art baselines, E3C achieves an average of 1.3x model inference speedup, and over 40% energy efficiency improvement, while maintaining high accuracy.

To summarize, this paper aims to fill current research gaps by investigating lightweight techniques including knowledge distillation and early-exit on a computationally efficient DNN model for the RSSC problem on resource-constrained edge devices. Through knowledge distillation, the distilled GFNet model achieves high model accuracy with model size and inference latency significantly reduced. A new early-exit model is designed for RSSC model inference on edge devices, which further reduces model latency and energy consumption. The contributions of this paper are as follows:

- We investigate lightweight DNN techniques and models for the RSSC problem, and find the root causes for the high inference latency issue on resource-constrained edge devices.
- We propose frequency distillation on the GFNet model and a dynamic early-exit model tailored for DNN models on edge devices. We find that knowledge distillation and early-exit mechanisms complement each other in enhancing model efficiency.
- We design a lightweight RSSC framework by integrating proposed DNN models and optimizing the use of heterogeneous computing resources. Extensive evaluation across four datasets on three edge devices show that our E3C model achieves state-of-the-art trade-off performance in terms of accuracy, inference latency and energy efficiency.

2 Related Work

Remote Sensing Scene Classification. Remote sensing scene classification (RSSC) is a fundamental task of interpreting remote sensing imagery data to label them with correct semantic categories. The key to RSSC is to extract informative and discriminative features [17]. DNN-based RSSC approaches have the capability of extracting discriminative and semantic features more effectively, compared with traditional handcrafted feature extractors. For example, a multiscale spatial-frequency Transformer model uses self-attention to better capture the global dependence of the spatial and frequency multiscale features for RSSC [31]. CGINet proposes a light context-aware attention block to explicitly model the global context to obtain larger receptive fields and more contextual information, so that global context and part-level discriminative features can be combined within a unified framework for accurate RSSC [35]. The RSSC models mentioned above are accompanied by huge computational costs. Thus, recent studies have investigated various lightweight DNN techniques to improve the efficiency of these RSSC models [17]. For example, SDT2Net incorporates an efficient attention block and a differentiable token compression mechanism, which adaptively prunes tokens under a given FLOPS constraint to build an efficient RSSC Transformer model [16]. EFPF is an energy-based filter pruning framework designed to reduce model size, which calculates layer energy using the eigenvalues of each weight tensor through singular value decomposition [15]. While these lightweight DNN models can reduce model size and FLOPS requirements, little work focuses on optimizing and deploying DNN models on resource-constrained edge devices for real-time RSSC model inference.

Knowledge Distillation. Knowledge distillation is an effective model compression technique to improve the efficiency of DNN models without increasing the number of parameters or computational cost. The application of knowledge distillation is pioneered by [10], and it has becomes a popular technique for transferring knowledge from large teacher models to smaller student models. Nowadays, it is primarily categorized into response-based methods, feature-based methods and relation-based methods [30]. In this paper, we adopt the response-based method, also known as the logit-based method, which involves transferring knowledge through the final classification layer or logits of the model. Recently, methods have been proposed to compress remote sensing scene classifiers using knowledge distillation. For example, a target-aware knowledge distillation method introduces a target extraction module, and then a target-aware loss is designed to enable the transfer of knowledge in the target regions from the teacher model [26]. This method can reduce RSSC model size significantly, but it also adds an additional target extraction module, which increases computational overhead.

Early-exit DNN Models. The early-exit DNN models belong to a broader category of dynamic neural networks [9], in which the model depth is dynamic by the early-exit mechanism to allowing "easy" samples to be output at shallow exits without executing deeper layers. The goal of the early-exit mechanism is to augment the backbone model with additional trainable components so as to obtain a final result at a reduced inference cost, and BranchyNet is among the first initiatives to propose adding early-exit branches to DNN models to reduce model inference time [22]. For time-sensitive edge computing applications, various early-exit methods have been proposed to achieve better inference performance [13, 14]. For example, TLEE proposes to apply the exit-early mechanism to video recognition, which automatically determines when and where to exit the inference process based on complexity and performance requirements of the input video [25]. In addition, E4 proposes an energy-efficient DNN inference framework for edge video analysis, which achieves state-of-the-art DNN inference performance by integrating a early-exit module with a DVFS regulator to adjust CPU and GPU clock frequencies optimally based on the DNN exit points [34]. However, we have found little work on integrating early-exit mechanisms with other lightweight techniques for real-time RSSC on edge devices. In this paper, our early-exit model is built upon JEI-DNN, which enhances the backbone model with jointly optimized gating and intermediate inference modules [19].

3 Methodology

The overview of our lightweight RSSC framework is presented, followed by detailed introduction of two key components.

3.1 Overview

The overview of our E3C Framework is shown in Fig 1, which includes knowledge distillation and **E**arly-**E**xit for **E**dge **C**omputing-oriented (E3C) RSSC. The framework uses a computationally efficient GFNet as the backbone model, which learns long-range spatial dependencies in the frequency domain with a computational complexity of $O(nlogn)$. In our model training phase, we first train a distilled GFNet model using the logit-based knowledge distillation method in the frequency domain. Then, we train the gating and intermediate inference modules in the early-exit model in an alternating way [19]. Finally, we deploy trained models on edge devices to perform model inference. For edge devices with heterogeneous CPU-GPU processors, we separate the compute of model parameters on different processors to better utilize the computational resources. Our E3C model can achieve real-time RSSC inference with high classification accuracy on various edge devices.

3.2 Knowledge Distillation on GFNet Model

Now we introduce the GFNet model and describe how we apply knowledge distillation on it to further improve computation efficiency. Given an input image $\mathbf{x} \in \mathbb{R}^{H\times W\times C}$, a 2D discrete Fourier transform $\mathcal{F}[\cdot]$ can be performed to convert the input spatial features to the frequency domain: $\mathbf{X} = \mathcal{F}[\mathbf{x}] \in \mathbb{C}^{H\times W\times C}$. Then the spectrum $\mathbf{X}$ in the frequency domain can be modulated by multiplying a learnable global filter $\mathbf{K} \in \mathbb{C}^{H\times W\times C}$:

$$\tilde{\mathbf{X}} = \mathbf{K} \odot \mathbf{X}, \tag{1}$$

where $\odot$ is the element-wise multiplication, also known as the Hadamard product [21]. Finally, the modulated spectrum $\tilde{\mathbf{X}}$ can be transformed back to the spatial domain using a 2D inverse Fourier transform. As shown in [18], the global filter $\mathbf{K}$ is equivalent to a depthwise global circular convolution with a complexity of $O(CS^2)$, where $S = HW$, and can learn features effectively in the frequency domain. Since the 2D Fourier and inverse Fourier transforms have fast Fourier transforms (FFT) algorithm implementations, and the global filter in the frequency domain enjoys a complexity of $O(CS\log S)$, GFNet is a more efficient DNN model than ViT models.

As mentioned in Section 1, to further improve the computational efficiency, we apply knowledge distillation on the GFNet model. Specifically, we use the effective hard-label distillation loss from the Data-efficient image Transformer (DeiT) model [23] as the total loss for our distillation: $\mathcal{L}_{\text{hardDistill}} = \frac{1}{2}\mathcal{L}_{CE}^{y} + \frac{1}{2}\mathcal{L}_{CE}^{teacher}$, where $\mathcal{L}_{CE}^{y}$ denotes the cross entropy loss from the ground-truth y, and $\mathcal{L}_{CE}^{teacher}$ denotes the cross entropy loss from the GFNet teacher model. Since GFNet uses FFT to transform inputs into the frequency domain, we propose to perform frequency domain distillation using

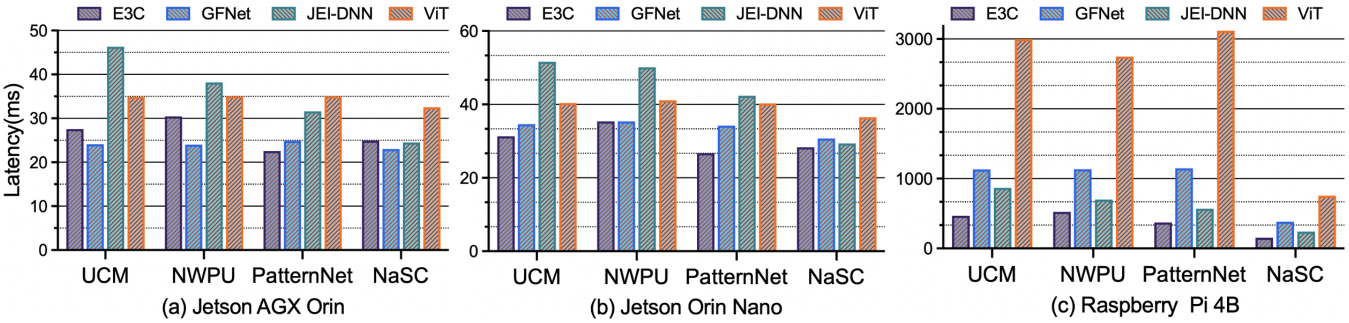

Figure 2: Comparison of model inference latency on three edge devices across four RSSC datasets.

a distillation token and a class token. The class token can learn the inherent features and distribution of the data [6], while the distillation token learns the network information of the teacher model [23]. Note that our goal is to investigate the idea of using knowledge distillation methods to further improve the model inference efficiency of the GFNet model on edge devices. We choose the hard-label distillation method, since it is parameter-free and simpler compared with the soft distillation method, but other distillation methods can also be used within our framework.

3.3 Early-exit Models

For the early-exit model, we first introduce a state-of-the-art early-exit dynamic networks called JEI-DNN, then describe how we build upon it to design a new early-exit mechanism for lightweight model inference on edge devices.

3.3.1 JEI-DNN Model. Traditional early-exit models add two additional components to the backbone model: 1) inference modules (IMs), also known as intermediate classifiers to generate results at particular network depths, 2) gating mechanisms (GMs) for determining which IMs should be used to derive the final result [9]. However, in most early-exit dynamic networks, the IMs and GMs are decoupled in training, since simple threshold-based gating mechanisms are used with the GMs treated as a post-training add-on component. In addition, for edge computing-oriented settings where DNN models can produce different outputs with varying computational resources, confidence measures are essential but are missing from traditional early-exit models. Thus, the state-of-the-art early-exit model JEI-DNN introduces an approach for modeling the probability of exiting at a particular inference module. It also proposes to avoid train-test mismatch and provide good uncertainty characterization by optimizing a loss that jointly assesses accuracy and inference cost [19]:

$$\mathcal{L}_b = \mathcal{L}_{CE}(p_b, y) + \lambda IC_b, \tag{2}$$

where b represents an early exit branch point, i.e., exit at layer b, p_b is the prediction probability, IC_b is the inference cost when the model chooses to exit at layer b, λ is the trade-off coefficient between inference cost and model accuracy representing the importance of inference cost.

However, directly optimizing (2) is challenging, and JEI-DNN adopts an alternating training strategy for updating the parameters of the GMs and IMs. Specifically, four uncertainty statistics are used as the feature inputs for GMs: the maximum predicted probability $\max(p_b)$, the entropy of the prediction $\sum_k p_b^k \log(p_b^k)$, the entropy scaled by a temperature $\sum_k \tilde{p}_b^k \log(\tilde{p}_b^k)$ and the difference between the two most confident predictions [19]. For IMs, a single layer neural network is used to allow inserting exit branches at every layer. Note that while the GMs and IMs modules are trained alternately, our GFNet-based backbone model is trained separately. That is, the backbone model parameters are frozen, while the parameters of the gate modules and the intermediate classifiers are updating. Also note that the GMs have a gating threshold parameter, whose impact is investigated in the evaluation section.

3.3.2 Lightweight Early-exit Model. As mentioned in Section 1, we find that the JEI-DNN model does not always improve DNN inference performance on resource-constrained edge devices. The reason lies in the fact that although the intermediate classifier is a lightweight single neural network, adding IMs and GMs at each and every layer of the backbone model can still increase computational overhead, especially when "early classifiers" cannot complete the inference tasks for certain difficult inputs. Thus, we propose a new early-exit model tailored for real-time RSSC on edge devices by improving the JEI-DNN model in the following way.

First, since we use a distilled GFNet model as the backbone of our framework, adding early exit branches to the shallow layers of the distilled model will increase computational redundancy during model inference. Thus, we propose to start incorporating early-exit branches from an intermediate layer with an index of l_m. In addition, instead of adding early-exit branches on each and every layer of the model, we propose to incorporate early-exit branches every M layers, so as to reduce computational redundancy while ensuring high accuracy:

$$\mathcal{B} = \{b \ ; \ l_m \le b < L, \ (b - l_m)\%M = 0\}, \tag{3}$$

where $\mathcal{B}$ is the set of the early-exit insertion points, and the total number of GFNet blocks is $L - 1$. Note that we investigate the impacts of the parameters l_m and M in our evaluation section.

Finally, many edge computing devices have heterogeneous processors of CPUs and GPUs. We propose the following mechanism for the early-exit model to better utilize the CPU-GPU heterogeneous computational resources. As mentioned in Section 3.3.1, uncertainty statistics with logarithmic operations need to be computed for the gating mechanisms of the early-exit model. Thus, we load the computation of the early-exit model parameters on CPUs, while keeping other model parameters on GPUs during model inference. This design mechanism further reduces inference latency by better utilizing heterogeneous processors on edge devices.

4 Experiments and Results

This section presents the E3C implementation details, followed by experimental settings and performance evaluation.

4.1 Implementation Details

As shown in Fig 1, the E3C framework has a training phase and an inference phase. In the training phase, the distillation procedures in DeiT [23] are used to perform frequency domain distillation on the GFNet model to train a distilled GFNet model, during which the learning rate is set to be 0.01 and the number of epochs to be 300. Then, for training the early-exit model, the distilled GFNet backbone network is frozen, while the IMs and GMs modules are trained as discussed in Section 3.3.1. We start by training all IMs on the full dataset in a warm-up stage, which ensures that IMs are performing reasonably well when we start training the GMs modules [19]. We adopt an alternating training strategy for updating the parameters of the IMs and GMs, during which the learning rate is set to be 0.01 with 200 training iterations. Four Nvidia RTX 4090 GPUs are used in the training phase, after which the trained E3C model is deployed on edge devices to perform model inference. Note that DNN model inference optimization techniques, such as TVM [3], are orthogonal to the techniques used in our E3C model, and can be used to further improve model performance in future.

For the early-exit model parameter determination, we use the following statistics from the training data. We define exit rate as the proportion of samples exiting at a particular layer relative to the total number of input samples, and define exit accuracy as the proportion of correctly classified samples among those exiting at that layer. We can derive these statistics during our model training, with examples shown in Fig. 3 and our supplementary material. We see that the histograms of the exit rate metric with high exit accuracy starts from the 5th layer of the model for most of the training datasets. Since the JEI-DNN model is proposed to avoid train-test mismatch [19], we use the statistics from the training dataset to set the early-exit model parameter $l_m = 4$, i.e., use the 5th layer as the starting point of inserting early-exit branches. Similarly for the layer interval parameter M, we find that the best latency and energy efficiency can be achieved by setting $M = 2$. If setting otherwise, e.g., $M = 3$, adding early-exit branches every three layers, it would produce insufficient exit points, especially considering the fact that we incorporate early exit branches from an intermediate layer. Note that for simplicity, we use the same early-exit configuration parameters l_m and M for all four RSSC datasets in our evaluation. However, the early-exit configurations can be made adaptively based on dataset complexity to further improve the performance, which we leave as future work.

In the model inference phase, we deploy the E3C model combining the strengths of a distilled GFNet model and an early-exit model on three different edge computing devices: NVIDIA Jetson AGX Orin, NVIDIA Jetson Orin Nano and Raspberry Pi 4B. The specifications of these edge devices are listed in Table 1. For measuring the power consumption of the edge devices during inference, we use jetson-stats [1] and Power-Z KT002, which is a portable instrument that measures the current and voltage of edge devices. From Table 1, we see that the Jetson AGX Orin edge device has high-performance CPU-GPU modules with 64GB RAM delivering up to 275 TOPS of compute performance. Jetson Orin Nano also has heterogeneous CPU-GPU modules, but only has 20 TOPS and 8GB RAM, while Raspberry Pi 4B only has 4GB RAM without GPUs, the fewest computational resources among the three edge devices. In Section 4, we show that our E3C model outperforms existing state-of-the-art models on these edge devices. Python and the PyTorch library are used in our implementation, and our code and data are publicly available at https://github.com/KaHim-Lo/GFNet-Dynn.

Table 1: Edge devices used in this study.

Edge Device	Specs.
Jetson Orin Nano	CPU: 6xCortex-A78AE@1.5GHz GPU: 512xAmpere@0.6GHz
Jetson AGX Orin	CPU: 12xCortex-A78AE@2.2GHz GPU: 2048xAmpere@1.3GHz
Raspberry Pi 4B	CPU: 4xCortex-A72@1.5GHz

4.2 Experimental Settings

Datasets. We evaluate our E3C model using four RSSC datasets: NaSC-TG2 [37], PatternNet [36], UC Merced Land-Use [32], and NWPU-RESISC45 [4]. NaSC-TG2 is an RSSC dataset from the Tiangong-2 space station, which comprises 20,000 remote sensing images, divided into 10 natural scenes, with each scene containing 2,000 images of 128×128 pixels [37]. PatternNet is a large-scale high-resolution remote sensing dataset collected for remote sensing image retrieval. There are 38 classes and each class has 800 images of size 256×256 pixels [36]. The UC Merced (UCM) dataset is one of the most widely used remote sensing benchmarks, which contains 21 scene categories, with 100 images of 256×256 pixels per category [32]. The NWPU-RESISC45 dataset contains 45 scene categories with variations in spatial resolution, object pose, translation, illumination, viewpoint, occlusion, and background [4]. While other aerial scene classification datasets, such as AID [28] can also be used in the evaluation, the dataset diversities mentioned above validate generalizability across common RSSC challenges. For example, in terms of diversity in spatial resolution, PatternNet has the highest resolution ranging from 0.062m to 4.693m, while that of NWPU-RESISC45 ranges from 0.2m to 30m. Note that to ensure consistent input resolution across all datasets, all images were resized to a uniform resolution of 256×256 pixels in our evaluation. We also randomly split each of the datasets into training sets with the size of 0.8 and testing sets with the size of 0.2 in our evaluation. The seeds and partition indices can be found on our GitHub repository.

Existing alternatives. We compare our E3C model with the following state-of-the-art alternatives.

- **JEI-DNN** is a state-of-the-art dynamic neural network model with early-exit branches, which enhances the backbone Transformer model with jointly optimized lightweight and trainable gates and intermediate classifier modules [19].

Table 2: Comparison with baselines and ablation studies of distillation and early exit across four datasets on three edge devices.

Dataset	Distillation	Early Exit	Params (MB)	Accuracy (%)	Jetson AGX Orin		Jetson Orin Nano		Raspberry Pi 4B	
					Energy (mJ)	Improve (%)	Energy (mJ)	Improve (%)	Energy (mJ)	Improve (%)
UCM	✗	✗	15.92	99.3	144.8	N/A	96.9	N/A	1233.5	N/A
	✓	✗	4.55	98.6	75.3	48.0	49.8	48.6	547.2	55.6
	✗	✓	15.95	99.3	126.0	13.0	86.3	10.9	1105.3	10.4
	✓	✓	**4.57**	**97.9**	**68.8**	**52.5**	**40.7**	**58.0**	**432.2**	**65.0**
NWPU	✗	✗	15.93	96.4	141.6	N/A	99.1	N/A	1237.8	N/A
	✓	✗	4.56	95.0	73.2	48.3	51.0	48.5	538.7	56.4
	✗	✓	16.00	96.3	108.8	23.1	73.1	26.2	948.0	23.4
	✓	✓	**4.60**	**94.7**	**88.2**	**41.9**	**46.0**	**53.6**	**480.7**	**61.2**
PatternNet	✗	✗	15.93	99.7	144.4	N/A	95.8	N/A	1378.9	N/A
	✓	✗	4.56	98.9	73.2	49.3	48.2	50.0	500.9	63.7
	✗	✓	15.99	99.5	101.5	29.7	68.3	28.7	903.8	34.5
	✓	✓	**4.59**	**97.9**	**58.5**	**59.5**	**34.7**	**63.8**	**340.6**	**75.3**
NaSC	✗	✗	14.89	99.1	71.3	N/A	39.9	N/A	412.1	N/A
	✓	✗	3.92	98.3	51.7	27.5	28.6	28.3	151.8	63.2
	✗	✓	14.90	98.8	49.9	30.0	29.4	26.3	256.5	37.8
	✓	✓	**3.93**	**97.4**	**49.8**	**30.2**	**22.6**	**43.3**	**140.2**	**66.0**

- **GFNet** is a conceptually simple yet computationally efficient model, which learns long-term spatial dependencies in the frequency domain with log-linear complexity [18]. We have more detailed introduction in Section 3.2.
- **ViT** processes images as sequences of patches using Transformer-based self-attention, and serves as the backbone model for many variants [6]. In this paper, we choose the TinyViT variant, in which an efficient small vision transformer is obtained by transferring knowledge from a large pretrained model [27].

4.3 Performance Evaluation

The performance of E3C will be compared with the aforementioned alternatives, followed by ablation study and sensitivity test.

4.3.1 Model comparison. Table 2 shows the comparison of our E3C model with state-of-the-art DNN models including GFNet, TinyVit, JEI-DNN in terms of model size, classification accuracy, and inference energy consumption. For each dataset in Table 2, the first row contains results from GFNet, the second row is from TinyViT, the third row is from JEI-DNN, and the last one (with distillation and early-exit both checked) is from E3C.

Comparison of model accuracy. Similar to other lightweight DNN models, our E3C model suffers from accuracy loss due to the knowledge distillation and early-exit mechanisms. However, E3C achieves over 94.7% accuracy across four datasets, with accuracy loss less than 1.7%, compared with the state-of-the-art baselines, as shown in Table 2. For real-time RSSC applications, E3C may decrease model accuracy slightly, but can significantly improve inference latency, model size, and energy consumption, as we discuss next. In addition, the E3C framework is backbone-agnostic, allowing it to be applied to more advanced DNN models beyond GFNet, potentially achieving even higher accuracy in the future.

Comparison of DNN model size. Since we use different datasets to train our distillation and early-exit models, we report the model size, i.e., the number of model parameters of the E3C model as well as baseline models, for each and every dataset, as shown in Table 2. We see that the model sizes of E3C are all below 5MB across four datasets, significantly lower than the GFNet and JEI-DNN models, which have similar model sizes, since no distillation is applied. On average, E3C achieves a 71.8% reduction in model size compared to GFNet, making it highly suitable for deploying RSSC models on resource constrained edge devices. We also see that the E3C model has larger mode sizes than the TinyViT model with distillation applied to the ViT model. This is due to the fact that E3C has additional early-exit network modules, e.g., the IMs and GMs modules. Note that the model size of E3C is only slightly larger than that of TinyViT, e.g., 4.57MB vs. 4.55MB for the UCM dataset. However, a slight increase in model size leads to a significant improvement in energy efficiency, as we discuss next.

Comparison of energy consumption. For a given dataset, the same RSSC model exhibits varying energy consumption across different edge devices. Thus, we show the energy consumptions from the E3C and baseline models on three different edge devices in Table 2. GFNet shows the highest inference energy consumption among the RSSC models, as it employs neither distillation nor early-exit mechanisms. The E3C model demonstrates significant energy savings on the Jetson AGX Orin device, achieving an average reduction in energy consumption of 47.2% across four datasets compared to the GFNet model. Similarly, E3C achieves an average energy saving of 56.6% on the Jetson Orin Nano device, compared to GFNet. For the Raspberry Pi edge device, E3C achieves an average energy saving of 66.9%, compared to GFNet. Compared to JEI-DNN, E3C achieves an average energy consumption reduction of 41.3%

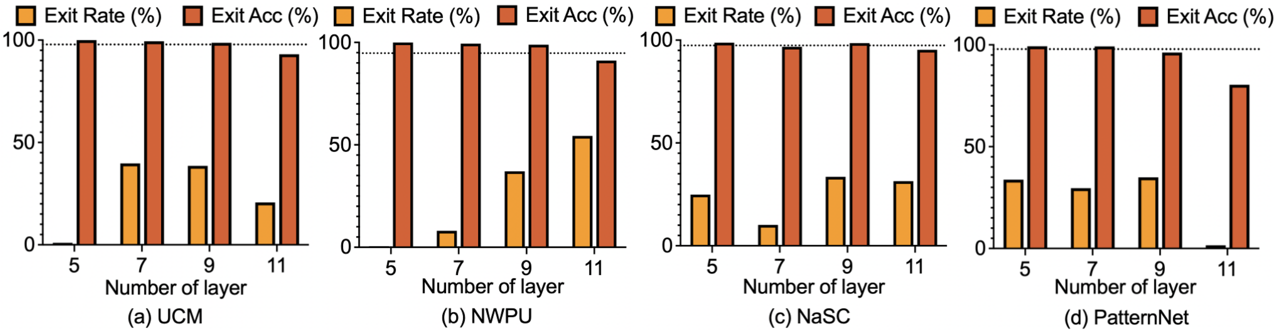

Figure 3: Exit rate and exit accuracy of E3C in four different datasets. The dotted line represents the overall model accuracy.

to 45.5% on two heterogeneous edge devices, and a 48.7% reduction on the CPU-only Raspberry Pi device. Thus, based on the results in Table 2, we see that the energy savings achieved by E3C stem from its early-exit module, which enables input images to terminate inference early, as well as from the distillation mechanism, since the smaller model size helps reduce both computational cost and redundancy. More detailed comparison of the energy efficiency performance of the E3C and other RSSC models are available in the supplementary material.

Comparison of model inference latency. For the real-time RSSC application on edge devices, DNN model inference latency is also an important performance metrics. Fig. 2 shows the inference latency results for E3C and baseline models. From Fig. 2(a), we see that E3C achieves up to 1.7x inference speedup compared to the JEI-DNN model on the Jetson AGX Orin device. However, the GFNet model has slightly better performance than E3C for the UCM, NWPU and NaSC datasets on Jetson AGX Orin. This is primarily because the Jetson AGX Orin features high-performance CPU-GPU modules, whereas the Jetson Orin Nano and Raspberry Pi have limited computational resources. As a result, the early-exit mechanism offers limited benefit for high-performance processors. We further analyze the effects of early-exit on different datasets in Section 4.3.3. As shown in Fig. 2(b), on the Jetson Orin Nano device, E3C demonstrates superior inference latency performance compared to alternatives. Compared to GFNet, E3C achieves up to a 1.6x inference speedup on average. Finally, for Raspberry Pi with only CPUs, the inference latencies of all models are significantly higher than those from Jetson AGX Orin and Orin Nano. This is expected since Raspberry Pi has the most limited computational resources among the three edge devices. However, the model inference speedup from our E3C model is also the most significant, compared with the other edge devices. For example, the inference latency from E3C is below 450 milliseconds, while the latencies from alternatives are all above 750 milliseconds for the UCM dataset. Compared to GFNet, E3C achieves up to 3.1× speedup and compared to JEI-DNN, E3C achieves up to 1.9× speedup on resource constrained devices. To summarize, while the inference latency of GFNet is comparable to that of E3C on the NWPU dataset, our E3C model achieves 1.3× speedup compared to the state-of-the-art JEI-DNN model on average, and significantly outperforms all alternative models including GFNet, JEI-DNN and ViT on Jetson Orin Nano and Raspberry Pi edge devices for all the other datasets.

4.3.2 *Ablation Study.*

We perform ablation study and investigate the effects of different parameters on the E3C performance.

Impacts of distillation and early-exit modules. We conduct ablation experiments to study the impacts of frequency domain distillation and early exit on model size, accuracy and energy efficiency. From Table 2 we see that the reduction in model size mainly comes from the knowledge distillation module. For example, the number of parameters of the distilled GFNet model is 4.57MB for the UCM dataset, a 71.4% reduction from 15.92MB of the original GFNet model. In terms of model accuracy, the early-exit module does not cause much degradation, with the accuracy loss within 0.3%, while the knowledge distillation module keeps the accuracy loss within 1.4% across all four datasets.

In addition, we see that the reduction in energy consumption during model inference comes from both distillation and early-exit. For example, knowledge distillation brings energy consumption on Jetson AGX Orin down from 144.8mJ to 75.3mJ, a 48% reduction for the UCM dataset, while early-exit reduces energy consumption by 13%. Adding early-exit modules increases model sizes slightly, but it can further reduce energy consumption, without sacrificing much accuracy. The energy consumption reduction rate for early exit ranges from 10.4% to 37.8%, while it ranges from 27.5% to 63.7% for distillation. Although adding distillation has more improvement on energy efficiency than adding early-exit modules on the UCM, NWPU and PatternNet datasets, the model accuracies of DNN models with early-exit modules are higher than those of models with distillation modules across all four datasets. Compared with the original GFNet model without distillation, E3C has the most significant performance improvement, achieving energy savings of 30.2% to 75.3%. It reveals the effective complementarity of frequency domain distillation and early-exit methods in reducing inference energy consumption. Note that the Raspberry Pi edge device has the fewest computational resources among three edge devices, and it consumes more energy running RSSC models. For the same UCM dataset, our E3C model running on Raspberry Pi consumes 432.2 mJ, while it only consumes 40.7 mJ on Jetson Orin Nano. However, it is on the resource constrained Raspberry Pi device that E3C has the most significant improvement on energy efficiency, with over 60% improvement upon the original GFNet model across all datasets.

Another interesting observation is that for the NaSC dataset, the DNN model with early-exit outperforms the DNN model with distillation in terms of both model accuracy and energy efficiency on Jetson AGX Orin. Since the NaSC dataset primarily comprises of "easy" imagery data, the DNN model with early-exit can make a significant portion of input images exit at shallow layers, thus significantly reducing inference latency and energy consumption, while maintaining high accuracy. Our data analysis shows that a significant amount of input images can exit within the first few early-exit branches for the NaSC dataset. We analyze the impacts of exit rate and datasets next.

Impacts of datasets and exit rate. As mentioned above, the early-exit model shows different performance on different datasets, since certain data inputs may not be "easy enough" to be determined by early classifiers. For different datasets, Fig. 3 shows the exit rate and corresponding exit accuracy statistics defined in Section 4.2. We see that a significant amount of input images exit in the range of 6th to 11th layers. For example, for the UCM and NWPU datasets shown in Fig. 3(a) and (b), almost no samples exit at the 5th layer of the E3C model. If we add early-exit modules to each and every layer of the backbone model as in the JEI-DNN model, it will increase computational redundancy and cause relatively high inference latency. However, for the NaSC dataset statistics shown in Fig. 3(c) and our supplementary material, more input imagery data can exit from the 5th layer or even earlier layers with high exit accuracies. It explains the reason why the early-exit models behave differently for different datasets.

4.3.3 *Sensitivity Test.*

We perform sensitivity test on the early-exit model parameters next.

Impact of the early-exit threshold parameter. As mentioned in Section 3.3, a threshold parameter is used in the gating mechanism of the early-exit module. As shown in Fig. 4, we report the accuracy of E3C under various threshold values. We see that as the threshold value increases, the model accuracy also increases in general. It makes sense since the larger the threshold value is, the more strict the conditions are for early exit, which can lead to higher accuracy with higher computational cost. In addition, for most datasets, the model accuracy is not sensitive to the threshold value in a wide range from 0.4 to 0.7. For the NaSC dataset, since more input imagery data can exit from early layers with lower accuracy, as shown in the supplementary material, increasing the threshold value can move the exit points to later layers, thus leading to higher accuracy. However, for threshold greater than 0.7, model accuracy has diminishing returns for all datasets.

Impact of the early-exit starting point parameter. As mentioned in Section 3.3.2, we start incorporating early-exit branches from an intermediate layer of an index of l_m to reduce the early-exit computational overhead. Based on our statistic analysis of exit rate in Section 4.3.2, we find that for our backbone model GFNet with 12 layers, a significant amount of data exit after the 5th layer with high classification accuracy, as shown in Fig. 3. Thus, we set the model parameter $l_m = 4$ based on the statistics. Here, we show the effect of this parameter on the model latency and energy consumption. We set l_m in the range of 0 to 6 representing the first seven layers, as the starting point of adding early-exit branches. We run our E3C model using the NWPU dataset on the Jetson AGX Orin edge device, and measure the model inference latency and energy consumption for different starting points. As shown in Fig. 5, both energy consumption and latency reach their minimum values when $l_m = 4$, which is consistent with our statistic analysis. In fact, when we set the parameter l_m too low, the model may struggle to capture the features of certain input images with shallow layers, producing high inference latency and energy consumption. If the parameter l_m is set too high, the early-exit branches will be shifted too far back, without much benefit of early-exit. Thus, this parameter is a trade-off between model accuracy and efficiency, which should be determined from training data statistics, as discussed in Section 4.2.

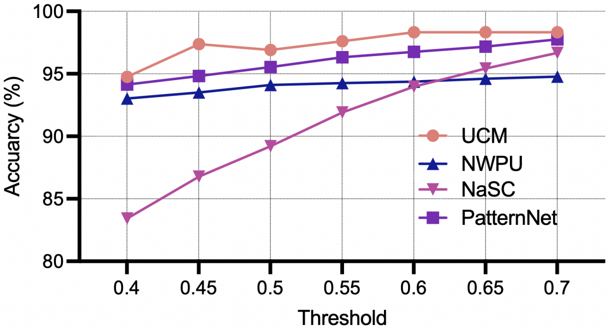

Figure 4: The classification accuracy of E3C under different thresholds across four datasets.

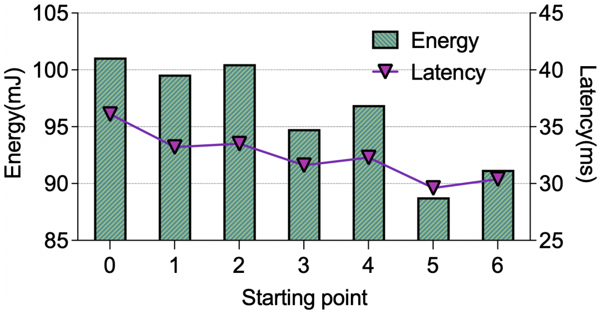

Figure 5: Comparison of inference energy consumption and inference latency for different starting points using the NWPU dataset on Jetson AGX Orin.

5 Conclusion

This work investigates the complementary benefits of the knowledge distillation and early-exit methods for lightweight RSSC DNN models on edge devices. We first perform frequency domain distillation on the GFNet model, and then design a new early-exit model and integrate it with the distilled GFNet model. For edge devices with CPUs and GPUs, we distribute the compute-intensive task of early-exit model parameter computation to CPUs, while keeping other model parameters on GPUs to better utilize heterogeneous computing resources. Extensive experiments with four RSSC datasets on three edge devices show that our E3C model outperforms state-of-the-art alternatives in terms of model size, inference latency, and energy efficiency, without sacrificing much model accuracy. Finally, the E3C framework is backbone-agnostic. Thus, this work can pave the way for further study of applying lightweight DNN models to other applications on edge devices.

Acknowledgments

This work is jointly supported by the National Key R & D Program of China (Grant No. 2022YFF0503900) and the Specialized Research Fund for State Key Laboratory of Solar Activity and Space Weather. We acknowledge the contributions of Zhaowei He, Weijie Xu, and Jiaqian Lu, who assisted in the experiments. We also thank the anonymous reviewers for their comments and suggestions.

References

[1] R Bonghi. 2022. jetson-stats. https://rnext.it/jetson_stats/. Accessed: 2025-04-10.

[2] Si-Bao Chen, Qing-Song Wei, Wen-Zhong Wang, Jin Tang, Bin Luo, and Zu-Yuan Wang. 2022. Remote Sensing Scene Classification via Multi-Branch Local Attention Network. *IEEE Transactions on Image Processing* 31 (2022), 99–109.

[3] Tianqi Chen, Thierry Moreau, Ziheng Jiang, Lianmin Zheng, Eddie Yan, Haichen Shen, Meghan Cowan, Leyuan Wang, Yuwei Hu, Luis Ceze, et al. 2018. {TVM}: An automated {End-to-End} optimizing compiler for deep learning. In *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*. 578–594.

[4] Gong Cheng, Junwei Han, and Xiaoqiang Lu. 2017. Remote sensing image scene classification: Benchmark and state of the art. *Proc. IEEE* 105, 10 (2017), 1865–1883.

[5] Gong Cheng, Xingxing Xie, Junwei Han, Lei Guo, and Gui-Song Xia. 2020. Remote sensing image scene classification meets deep learning: Challenges, methods, benchmarks, and opportunities. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 13 (2020), 3735–3756.

[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929* (2020).

[7] Gianluca Giuffrida, Luca Fanucci, Gabriele Meoni, Matej Batič, Léonie Buckley, Aubrey Dunne, Chris Van Dijk, Marco Esposito, John Hefele, Nathan Vercruyssen, et al. 2021. The Φ-Sat-1 mission: The first on-board deep neural network demonstrator for satellite earth observation. *IEEE Transactions on Geoscience and Remote Sensing* 60 (2021), 1–14.

[8] Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, Chunjing Xu, Yixing Xu, et al. 2022. A survey on vision transformer. *IEEE transactions on pattern analysis and machine intelligence* 45, 1 (2022), 87–110.

[9] Yizeng Han, Gao Huang, Shiji Song, Le Yang, Honghui Wang, and Yulin Wang. 2021. Dynamic neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence* 44, 11 (2021), 7436–7456.

[10] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* (2015).

[11] Qiangqiang Jiang, Lujie Zheng, Yu Zhou, Hao Liu, Qinglei Kong, Yamin Zhang, and Bo Chen. 2025. Efficient On-Orbit Remote Sensing Imagery Processing via Satellite Edge Computing Resource Scheduling Optimization. *IEEE Transactions on Geoscience and Remote Sensing* (2025).

[12] Jing Jin and Feng Wang. 2025. Fed-RSSC: A Semi-Decentralized Federated Framework for Remote Sensing Scene Classification. *IEEE Geoscience and Remote Sensing Letters* (2025).

[13] Xiangjie Li, Chenfei Lou, Yuchi Chen, Zhengping Zhu, Yingtao Shen, Yehan Ma, and An Zou. 2023. Predictive exit: Prediction of fine-grained early exits for computation-and energy-efficient inference. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 37. 8657–8665.

[14] Zhiyan Liu, Qiao Lan, and Kaibin Huang. 2023. Resource allocation for multiuser edge inference with batching and early exiting. *IEEE Journal on Selected Areas in Communications* 41, 4 (2023), 1186–1200.

[15] Yiheng Lu, Maoguo Gong, Zhuping Hu, Wei Zhao, Ziyu Guan, and Mingyang Zhang. 2023. Energy-based CNN pruning for remote sensing scene classification. *IEEE Transactions on Geoscience and Remote Sensing* 61 (2023), 1–14.

[16] Kang Ni, Qianqian Wu, Sichan Li, Zhizhong Zheng, and Peng Wang. 2024. Remote Sensing Scene Classification via Second-order Differentiable Token Transformer Network. *IEEE Transactions on Geoscience and Remote Sensing* (2024).

[17] Cheng Peng, Yangyang Li, Licheng Jiao, and Ronghua Shang. 2020. Efficient convolutional neural architecture search for remote sensing image scene classification. *IEEE Transactions on Geoscience and Remote Sensing* 59, 7 (2020), 6092–6105.

[18] Yongming Rao, Wenliang Zhao, Zheng Zhu, Jie Zhou, and Jiwen Lu. 2023. GFNet: Global filter networks for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 9 (2023), 10960–10973.

[19] Florence Regol, Joud Chataoui, and Mark Coates. 2024. Jointly-learned exit and inference for a dynamic neural network: Jei-dnn. In *International Conference on Learning Representations*.

[20] Md Maruf Hossain Shuvo, Syed Kamrul Islam, Jianlin Cheng, and Bashir I Morshed. 2022. Efficient acceleration of deep learning inference on resource-constrained edge devices: A review. *Proc. IEEE* 111, 1 (2022), 42–91.

[21] George PH Styan. 1973. Hadamard products and multivariate statistical analysis. *Linear algebra and its applications* 6 (1973), 217–240.

[22] Surat Teerapittayanon, Bradley McDanel, and Hsiang-Tsung Kung. 2016. Branchynet: Fast inference via early exiting from deep neural networks. In *2016 23rd international conference on pattern recognition (ICPR)*. IEEE, 2464–2469.

[23] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. 2021. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*. PMLR, 10347–10357.

[24] Mi Wang, Qianyu Wu, Jing Xiao, Deren Li, and Fang Yang. 2024. Luojia 3-01 Satellite—Real-Time Intelligent Service System for Remote Sensing Science Experiment Satellite. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 17 (2024), 8250–8257.

[25] Qingli Wang, Weiwei Fang, and Neal N Xiong. 2023. TLEE: Temporal-wise and Layer-wise Early Exiting Network for Efficient Video Recognition on Edge Devices. *IEEE Internet of Things Journal* (2023).

[26] Jie Wu, Leyuan Fang, and Jun Yue. 2024. TAKD: Target-aware knowledge distillation for remote sensing scene classification. *IEEE Transactions on Circuits and Systems for Video Technology* (2024).

[27] Kan Wu, Jinnian Zhang, Houwen Peng, Mengchen Liu, Bin Xiao, Jianlong Fu, and Lu Yuan. 2022. Tinyvit: Fast pretraining distillation for small vision transformers. In *European conference on computer vision*. Springer, 68–85.

[28] Gui-Song Xia, Jingwen Hu, Fan Hu, Baoguang Shi, Xiang Bai, Yanfei Zhong, Liangpei Zhang, and Xiaoqiang Lu. 2017. AID: A benchmark data set for performance evaluation of aerial scene classification. *IEEE Transactions on Geoscience and Remote Sensing* 55, 7 (2017), 3965–3981.

[29] Xiaobin Xu, Qi Wang, Yanzhao Hou, and Shangguang Wang. 2023. AI-SPACE: A cloud-edge aggregated artificial intelligent architecture for Tiansuan constellation-assisted space-terrestrial integrated networks. *IEEE Network* 37, 2 (2023), 22–28.

[30] Chuanpeng Yang, Yao Zhu, Wang Lu, Yidong Wang, Qian Chen, Chenlong Gao, Bingjie Yan, and Yiqiang Chen. 2024. Survey on knowledge distillation for large language models: methods, evaluation, and application. *ACM Transactions on Intelligent Systems and Technology* (2024).

[31] Yuting Yang, Licheng Jiao, Fang Liu, Xu Liu, Lingling Li, Puhua Chen, and Shuyuan Yang. 2023. An explainable spatial–frequency multiscale transformer for remote sensing scene classification. *IEEE Transactions on Geoscience and Remote Sensing* 61 (2023), 1–15.

[32] Yi Yang and Shawn Newsam. 2010. 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*. 270–279.

[33] Yuqun Yang, Xu Tang, Yiu-Ming Cheung, Xiangrong Zhang, and Licheng Jiao. 2023. SAGN: Semantic-aware graph network for remote sensing scene classification. *IEEE Transactions on Image Processing* 32 (2023), 1011–1025.

[34] Ziyang Zhang, Yang Zhao, Ming-Ching Chang, Changyao Lin, and Jie Liu. 2025. E4: Energy-Efficient DNN Inference for Edge Video Analytics Via Early-Exit and DVFS. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 39. 1165–1173.

[35] Yichen Zhao, Yaxiong Chen, Shengwu Xiong, Xiaoqiang Lu, Xiao Xiang Zhu, and Lichao Mou. 2024. Co-Enhanced Global-Part Integration for Remote-Sensing Scene Classification. *IEEE Transactions on Geoscience and Remote Sensing* 62 (2024), 1–14.

[36] Weixun Zhou, Shawn Newsam, Congmin Li, and Zhenfeng Shao. 2018. PatternNet: A benchmark dataset for performance evaluation of remote sensing image retrieval. *ISPRS journal of photogrammetry and remote sensing* 145 (2018), 197–209.

[37] Zhuang Zhou, Shengyang Li, Wei Wu, Weilong Guo, Xuan Li, Guisong Xia, and Zifei Zhao. 2021. NaSC-TG2: Natural scene classification with Tiangong-2 remotely sensed imagery. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 14 (2021), 3228–3242.