MACHINE LEARNING
Science and Technology

PURPOSE-LED PUBLISHING™ Part of the MACHINE LEARNING Series

PAPER • OPEN ACCESS

BiCrossNet: resource-efficient cross-view geolocalization with binary neural networks

To cite this article: Federico Fontana *et al* 2025 *Mach. Learn.: Sci. Technol.* **6** 035036

IOP Publishing Mach. Learn.: Sci. Technol. 6 (2025) 035036 https://doi.org/10.1088/2632-2153/adfa67

MACHINE LEARNING
Science and Technology

OPEN ACCESS

RECEIVED
27 March 2025

REVISED
10 July 2025

ACCEPTED FOR PUBLICATION
11 August 2025

PUBLISHED
21 August 2025

PAPER

BiCrossNet: resource-efficient cross-view geolocalization with binary neural networks

Federico Fontana[1,*], **Thomas Jantos**[2], **Jan Steinbrener**[2], **Luigi Cinque**[1], **Gian Luca Foresti**[3] and **Bernhard Rinner**[2]

[1] Sapienza University of Rome, Via Salaria 113, Rome, Italy
[2] University of Klagenfurt (AAU), Universitätsstraße 65/67, Klagenfurt am Wörthersee, Austria
[3] University of Udine (UNIUD), Via Palladio, 8, 33100 Udine, UD, Italy
[*] Author to whom any correspondence should be addressed.

E-mail: fontana.f@di.uniroma1.it

Keywords: cross-view geolocalization, binary neural networks, efficient deep learning

Abstract

This paper presents BiCrossNet, a novel approach to cross-view geolocalization utilizing binary neural networks to significantly reduce computational complexity while maintaining competitive performance. Key contributions include the development of a Bi-Gradual Unfreezing method to enhance transfer learning, a Bi-Partitioned Optimization strategy to improve training stability, and the use of logit-based knowledge distillation to supplement standard losses. Experimental results on the University-1652 and SUES-200 datasets demonstrate that BiCrossNet establishes a new benchmark in the efficiency-performance trade-off. It achieves up to a 90.87-fold reduction in operations and uses 4.64 times less disk space compared to similar-performing state-of-the-art models on the SUES-200 dataset, and a 30-fold reduction in operations and 5.13 times less disk space on the University-1652 dataset. The code is available at https://anonymous.4open.science/r/BiCrossNet-FB7A/README.md.

1. Introduction

In recent years, unmanned aerial vehicles (UAVs) have gained popularity due to their ability to capture high-quality multimedia data from the sky. As an agile robotic platform [1], UAVs capture images with a larger field of view, freer angles, and less occlusion than ground-view images. Due to improved target visibility, maneuverability and flexibility, UAVs have been widely deployed in many fields, such as accurate delivery [2, 3], autonomous driving [4, 5], inspection of critical infrastructure [6], and agriculture, plant inventorying and protection [7–9].

Geolocalization, which identifies the real-world geographic position, is fundamental to many UAV applications. UAV localization and navigation often rely on Global Navigation Satellite Systems (GNSS) [10, 11]. However, when GNSS signals are weak or denied, localization becomes inaccurate or impossible. Consequently, image-based cross-view geolocalization, which assists GNSS in achieving accurate localization, has emerged as a significant research area. This task typically involves matching images from a UAV's onboard camera with reference satellite-view images. Accurately establishing the UAV's position necessitates precise georeferencing of these satellite images, enabling effective alignment and comparison despite differing perspectives. This matching process presents substantial challenges due to vast visual appearance variations caused by different viewing angles, scales, and temporal changes.

Most recent learning-based cross-view geolocalization methods aim to learn a mapping function that projects multi-view images into a shared semantic space for feature discrimination. This ensures that images from the same location are close in feature space, and vice versa. For instance, some approaches focus on robust feature extraction using advanced CNN architectures [12, 13], while others explore attention mechanisms or transformer-based models to capture long-range dependencies and contextual information [14–16]. Techniques like specialized partitioning strategies [14] or multi-classifier designs [17] have also been proposed to enhance discriminative power. Further methods involve joint representation learning and

keypoint detection [18], or address practical challenges in UAV-satellite matching [19, 20]. Some works also investigate multi-environment self-adaptation [21] or employ transformer-based feature aggregation [22]. While these methods (e.g. [12, 15, 18, 20–22]) have demonstrated progressively better accuracy, they often prioritize performance over computational efficiency. This oversight is critical, as UAVs are typically resource-constrained platforms where low latency and minimal computational footprint are paramount for real-time operation. The significant computational demands of many state-of-the-art deep learning models hinder their practical deployment on such devices.

This paper addresses this critical gap by introducing BiCrossNet, a novel approach that leverages binary neural networks (BNNs) for cross-view geolocalization. BNNs drastically reduce computational complexity by binarizing weights and activations, replacing expensive floating-point operations with efficient bit-wise operations. However, training BNNs effectively to maintain competitive accuracy presents its own set of challenges, including information loss due to quantization and training instability.

To overcome these challenges and harness the efficiency of BNNs for cross-view geolocalization, BiCrossNet incorporates several specialized techniques. Our work makes the following key contributions:

- We introduce BiCrossNet, which, to our knowledge, is the first application of BNNs for cross-view geolocalization. It achieves an optimal balance between computational cost and performance, attaining average precision comparable to many existing networks, including state-of-the-art transformer-based solutions, while offering substantial reductions in operations and disk space (e.g. a 90.87-fold reduction in operations on SUES-200 and a 30-fold reduction on University-1652).
- We develop a Bi-Gradual Unfreezing approach tailored for BNNs, enhancing their transfer learning capabilities from pre-trained ImageNet models. This method addresses the common tendency of BNNs to overfit or suffer from catastrophic forgetting during fine-tuning.
- We propose a Bi-Partitioned Optimization strategy for BNNs, employing distinct optimizers. This differentiated approach enhances training stability and overall performance by allowing each optimizer to specialize in the distinct learning dynamics of each weight type.
- We implement logit-based knowledge distillation, transferring knowledge from a full-precision teacher model (ConvNext-tiny) to our proposed BNN. This technique supplements standard losses and helps the BNN learn more robust representations, as demonstrated in our ablation studies.
- Extensive testing demonstrates that BiCrossNet sets a new benchmark for performance and efficiency on two public datasets: *University-1652* [12] and *SUES-200* [23], achieving SOTA-competitive results in terms of the efficiency-accuracy trade-off.

2. Related work

2.1. Cross-view geolocalization

Cross-view geolocalization refers to matching ground or drone views with satellite views, encompassing two primary tasks: matching ground views with satellite views and drone views with satellite views. Initially, studies were limited by data acquisition constraints, often relying on hand-crafted features [24–26]. With the advent of deep convolutional neural networks (CNNs), research shifted towards learning deep representations [27–29]. This led to the creation of public datasets like CVUSA [28] and CVACT [29], which consist of ground-to-satellite image pairs. Workman and Jacobs [27] were pioneers in using pre-trained CNNs to extract high-level features for cross-view localization, showcasing their ability to encapsulate semantic geographic information. Influenced by Siamese network principles [28, 30] integrated NetVLAD [31] into a Siamese-like framework to craft robust image descriptors capable of withstanding significant viewpoint variations. Shi *et al* [32] explored domain alignment and employed a polar transform to warp aerial images, aiding in the alignment of multiple views. The increasing prevalence of UAVs spurred the development of UAV-centric datasets, advancing UAV-view geolocalization. Zheng *et al* [12] introduced University-1652, a dataset of UAV-satellite image pairs, evaluating image-retrieval from a classification perspective. They later expanded this with University-160k [15] by adding satellite-view gallery distractors. Zhu *et al* [23] developed the SUES-200 dataset, which assesses the impact of varying UAV altitudes on geolocalization. Stronger backbone networks have markedly improved image matching accuracy. Dai *et al* [16] utilized a Vision Transformer (ViT) [33] as the backbone, achieving results competitive with CNN-based frameworks by leveraging its global feature extraction capabilities. To thoroughly mine contextual information from aerial images, Wang *et al* [14] proposed a rotation-invariant square-ring partition approach, explicitly analyzing contextual data to improve robustness to orientation changes. Shen *et al* [17] designed a potent multiple-classifier architecture (MCCG) based on ConvNeXt to extract rich discriminative features, achieving state-of-the-art results on several benchmarks. While these methods

achieve impressive accuracy, their computational demands often neglect the efficiency crucial for resource-constrained UAVs.

2.2. BNNs

Courbariaux *et al* [34] introduced BNNs, where both weights and activations are binarized (typically to -1 or $+1$) using the sign function. This substantially replaces costly floating-point arithmetic with efficient bit-wise operations (XNOR and popcount).

However, BNNs inherently suffer from quantization errors due to the drastic reduction in precision. XNOR-Net [35] mitigated this by introducing channel-wise scaling factors to better reconstruct full-precision values from binarized weights, a technique that became fundamental. ABC-Net [36] attempted to approximate full-precision weights via linear combinations of multiple binary weight bases and used multiple binary activations to reduce information loss.

Inspired by ResNet [37] and DenseNet [38], Bi-RealNet [39] integrated shortcuts to bridge the performance gap between 1-bit and real-valued CNNs. BinaryDenseNet [40] further enhanced BNN accuracy by increasing shortcut density. IR-Net [41] introduced the Libra-PB method to minimize information loss during forward propagation by maximizing information entropy of quantized parameters. ReActNet [42] proposed generalized sign (RSign) and PReLU (RPReLU) functions, enabling learnable reshaping and shifting of distributions with minimal added cost. Other works have explored mitigating angular bias (RBNN [43]), eliminating L_2 regularization to maximize entropy (SiMaN [44]), rejuvenating 'dead weights' (ReCU [45]), and using adaptive binary sets and equalization techniques (AdaBin [46]). BiDet [47] advanced BNNs for object detection. More recently, BNext [48] achieved impressive ImageNet [49] classification accuracy (around 80%) by incorporating knowledge distillation and architectural modifications like Squeeze-and-Excite [50] modules. Despite these advancements, BNNs still face challenges such as reduced representational capacity compared to full-precision networks, training instability (especially for deeper architectures), and significant information loss. These issues can make optimization difficult and hinder performance on complex tasks. Our work aims to address some of these limitations in the context of cross-view geolocalization. To our knowledge, we are the first to apply BNNs to this important task.

3. Proposed method

3.1. Overview of BNN

In a traditional CNN setting, for an input tensor $a \in \mathbb{R}^{c_{\text{in}} \times h \times w}$ and a weight tensor of a 2D convolution $w \in \mathbb{R}^{c_{\text{in}} \times c_{\text{out}} \times k_h \times k_w}$, the output after the convolution operation is expressed as

$$\text{out} = \text{Conv}(\boldsymbol{a}, \boldsymbol{w}). \tag{1}$$

Here, the output tensor out is denoted by $\mathbb{R}^{c_{\text{out}} \times h' \times w'}$, where h' and w' represent the output's height and width, respectively. The stride, padding, filter dimensions, and the number of channels influence these dimensions. The computational operations typically use a 32-bit representation.

In the domain of BNNs, a pivotal transformation involves the binarization of both activation and weight tensors, whereby every tensor element is represented as either -1 or $+1$. The binarization is governed by the Sign function

$$\text{Sign}(x) = \begin{cases} -1 & \text{if } x < 0 \\ +1 & \text{if } x \geqslant 0 \end{cases}. \tag{2}$$

This approach replaces traditional floating-point multiplications with more efficient bit-wise operations, notably using the *XNOR* operation followed by a *BitCount* (or popcount) operation. The *XNOR* operation for elements in $\{-1, 1\}$ is defined as

$$a_i \odot w_i = \begin{cases} +1 & \text{if } a_i = w_i \\ -1 & \text{if } a_i \neq w_i \end{cases}. \tag{3}$$

Consider two binary tensors, $a_b \in \{-1, 1\}^{c_{\text{in}} \times h \times w}$ and $w_b \in \{-1, 1\}^{c_{\text{in}} \times c_{\text{out}} \times k_h \times k_w}$. The convolution output is then calculated as

$$\text{out}_b = \text{BitCount}(a_b \odot w_b) \tag{4}$$

where $\text{out}_b \in \mathbb{N}^{c_{\text{out}} \times h' \times w'}$ and $\odot$ signifies the *XNOR* operation. The *BitCount* function quantifies the number of 1's in the result of the *XNOR* operation (after mapping $-1, +1$ to $0, 1$), enhancing computational efficiency [34].

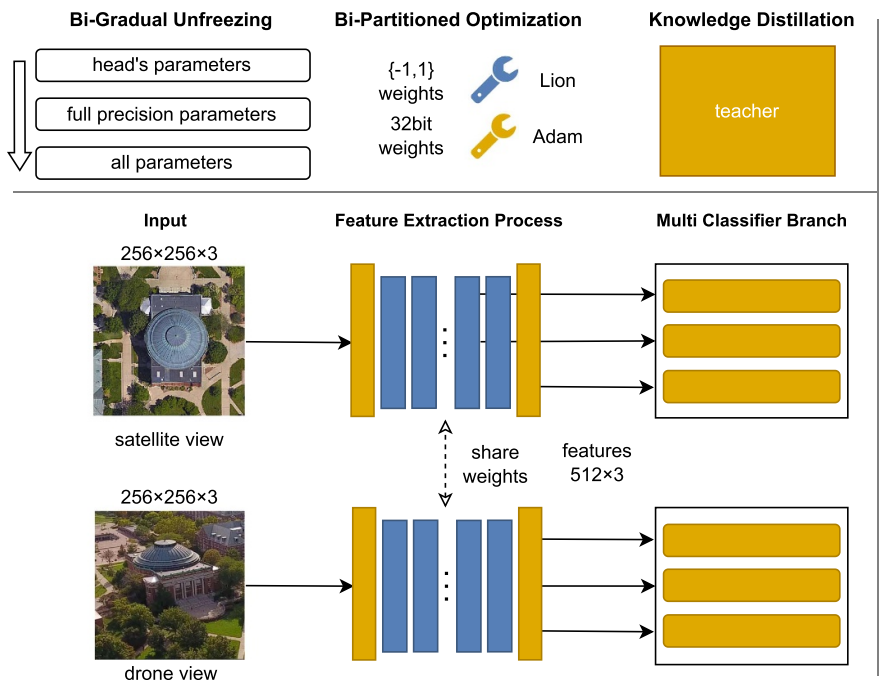

Figure 1. Our BiCrossNet architecture utilizes BNNs for geolocalization. The diagram illustrates the two operational modes. **(a) The training process follows the path 1a → 4a:** (1a) Features are extracted, (2a) then fed into the Multi-Classifier Branch, (3a) where the loss is computed, and (4a) the network is updated via backward pass. The process integrates three advanced learning strategies: Bi-Gradual Unfreezing, Bi-Partitioned Optimization, and logit-based knowledge distillation. **(b) The inference process follows the path 1b → 3b:** (1b) The Feature Extraction Process is used, (2b) features are concatenated, and (3b) ranked based on MSE distance. During inference, the Multi-Classifier Branch is discarded. **Legend:** Orange layers represent full-precision modules, and blue layers represent binarized modules.

During BNN training, weights w_b and activations a_b are binarized in the forward pass per equation (2). Real-valued latent weights w_r and pre-activation values a_r are maintained and updated during backpropagation. Due to the non-differentiable nature of the Sign(x) function (derivative is zero almost everywhere), the straight-through estimator [51] is commonly used to approximate gradients.

3.2. BiCrossNet architecture

In this section, we introduce our proposed BiCrossNet method. The overall network structure is illustrated in figure 1.

The first component is the **Feature Extraction Process**. We employ the BNext [48] network design, specifically BNext-tiny (with stage ratios 1:1:3:1 and base width 32) as our backbone, to extract feature maps from drone and satellite views. This process incorporates a spatial attention mechanism and concludes with an average pooling operation to produce feature vectors. The BNext architecture is chosen for its demonstrated high performance in BNNs on classification tasks, providing a strong foundation for feature extraction.

The second component is the **Multiple Classifier Branch**. This module, inspired by MCCG [17], integrates a multi-scale feature aggregation technique. It captures a richer set of feature representations from different layers or stages of the backbone. These aggregated features are then processed through separate classifier layers (each with an identical structure). This enhances the robustness and accuracy of the learned representations by forcing the network to learn discriminative features at multiple semantic levels. Although this component is not binarized, it is utilized *only during training* to guide feature learning and is discarded during inference. For the final inference result, the features are extracted from the BNext backbone directly, right before where the Multiple Classifier Branch would connect.

3.3. Loss functions

The training of BiCrossNet is guided by a composite loss function consisting of three main parts, applied to the outputs of the Multiple Classifier Branch.

First, we use the standard **Cross-Entropy Loss** (L_{CE}) for classification, encouraging the network to correctly classify the input images based on their geographical location (treated as classes during training):

$$L_{CE} = -\sum_{i} y_i \log(\hat{y}_i) \tag{5}$$

where y_i is the true label (one-hot encoded class) and $\hat{y}_i$ is the predicted probability for class i.

Second, to ensure that images of the same location from different views (drone and satellite) are mapped closely in the feature space while images from different locations are separated, we adopt the **Triplet Loss** ($L_{triplet}$) [52]. Following the approach in [17], we use a margin of 0.3:

$$L_{triplet} = \sum_{i} \left[d\left(x_i^{(a)}, x_i^{(p)}\right) - d\left(x_i^{(a)}, x_i^{(n)}\right) + 0.3 \right]_+ \tag{6}$$

where $x_i^{(a)}$ is the anchor feature, $x_i^{(p)}$ is the positive sample feature (same location, different view or instance), $x_i^{(n)}$ is the negative sample feature (different location), and d denotes the distance metric (e.g. Euclidean distance). The notation $[\cdot]_+$ represents the hinge function, $\max(0, x)$.

The overall loss S combines these components across the multiple feature representations from the Multi-Classifier Branch and includes the knowledge distillation loss (discussed next):

$$S = \frac{1}{3} \sum_{k=1}^{3} \sum_{j=1}^{2} \left(L_{CE_{jk}} + L_{triplet_{jk}} + L_{distill_{jk}} \right) \tag{7}$$

where k refers to the kth feature representation from the multi-scale aggregation (typically 3 scales are used), j distinguishes between drone and satellite views, and $L_{distill_{jk}}$ is the knowledge distillation loss for that specific feature set.

3.4. Logit-based knowledge distillation

To further enhance the BNN's performance and mitigate information loss from binarization, we employ Logit-based **Knowledge Distillation** (KD). This is a training paradigm where a compact 'student' model learns from the outputs of a larger, pre-trained 'teacher' model, in addition to the ground-truth labels. Our implementation of this technique transfers knowledge from a higher-capacity full-precision teacher model to our BNN student model. The teacher model used is ConvNext-tiny [53], trained on the same target dataset. The distillation loss ($L_{distill}$) is defined using mean squared error (MSE) between the logits (pre-softmax outputs) of the teacher and student networks:

$$L_{distill} = \frac{1}{N} \sum_{i} \left(\hat{y}_i^{teacher} - \hat{y}_i^{student} \right)^2 \tag{8}$$

where $\hat{y}_i^{teacher}$ and $\hat{y}_i^{student}$ are the logits from the teacher and student networks for the ith sample, respectively, and N is the number of samples. Using logits helps the student mimic the teacher's class probability distribution similarities, capturing more nuanced patterns than just matching hard labels. This is particularly beneficial for BNNs, as it provides a richer supervisory signal during training.

3.5. Bi-gradual unfreezing for fine-tuning

The proposed Bi-Gradual Unfreezing method is utilized to fine-tune the BNext backbone, pre-trained on ImageNet [49], adapting it to the cross-view geolocalization datasets. This phased approach is designed to stabilize training and prevent catastrophic forgetting, which BNNs can be particularly prone to. The process is as follows (algorithm 1):

(i) **Phase 1 (head training, n_1 epochs):** All pre-existing network weights (both full-precision and binary latent weights in the backbone) are frozen. Only the newly integrated head (the Multiple Classifier Branch and any final layers specific to the new task) is trained. This allows the head to adapt quickly to the new data domain without disrupting the learned features in the backbone.

(ii) **Phase 2 (full-precision backbone fine-tuning, n_2 epochs):** The full-precision weights in the backbone (e.g. first and last layers of BNext, batch normalization parameters, scaling factors) are unfrozen and trained along with the head. The binary weights (or their latent real-valued counterparts that get binarized) remain frozen. This allows for more extensive adaptation of the network.

(iii) **Phase 3 (full network fine-tuning, n_3 epochs):** All weights in the network, including the binary weights (i.e. their latent real-valued counterparts), are unfrozen and permitted to update. This allows the entire network to fine-tune to the specific nuances of the target dataset.

Algorithm 1. Bi-Gradual unfreezing method.

Input: Pre-trained BNN *NET* (BNext backbone) on ImageNet [49], new dataset D, number of epochs for each phase boundary n_1, n_2, and total epochs n_3, newly integrated head *HEAD*.
Freeze all pre-existing weights in *NET*. Only *HEAD* is trainable.
for epoch $= 1$ to n_1 **do**
    Train *NET* + *HEAD* on D.
end for
Unfreeze full-precision weights in *NET*. Latent binary weights in *NET* remain frozen. *HEAD* remains trainable.
for epoch $= n_1 + 1$ to $n_1 + n_2$ **do**
    Train *NET* + *HEAD* on D.
end for
Unfreeze all weights in *NET* (including latent binary weights). *HEAD* remains trainable.
for epoch $= n_2 + 1$ to $n_1 + n_2 + n_3$ **do**
    Train *NET* + *HEAD* on D.
end for
Output: Fine-tuned network *NET*' (NET + HEAD)

Algorithm 2. Bi-Partitioned optimization method.

Input: BNN *NET* with full-precision weights W_{fp} and latent binary weights W_{bin_latent}, new dataset D, total number of epochs n_3.
Initialize Adam optimizer for W_{fp}.
Initialize Lion optimizer for W_{bin_latent}.
for epoch $= 1$ to $n_1 + n_2 + n_3$ **do**
    for each batch in D **do**
        Perform forward pass.
        Compute gradients ∇L for all weights.
        Update W_{fp} using Adam with $\nabla L_{W_{fp}}$.
        Update W_{bin_latent} using Lion with $\nabla L_{W_{bin\_latent}}$.
    end for
end for
Output: Optimized *NET*'

This structured and phased approach allows for incremental network adaptation, mitigating the risk of catastrophic forgetting and enhancing the retention of useful features acquired during pre-training.

3.6. Bi-partitioned optimization

We propose a Bi-Partitioned Optimization approach for BNNs, where distinct optimization algorithms are strategically applied to different categories of network weights. Full-precision weights (e.g. in the first and last layers of BNext, batch normalization parameters, scaling factors, and the Multi-Classifier Branch) are managed by the Adam optimizer [54]. Adam is well-suited for these weights due to its adaptive learning rates and momentum-based updates, which generally lead to stable convergence. Concurrently, the latent real-valued weights that are subsequently binarized are managed by the Lion optimizer [55]. Lion is employed for these binarized weights as it is designed with sign-based updates, which can be more effective for parameters undergoing hard quantization. It often shows good performance in terms of generalization and can be more memory-efficient than Adam. This dual-optimizer setup (algorithm 2) leverages the strengths of each optimizer type. It allows for specialized handling of the different learning dynamics: the smoother optimization landscape of full-precision weights benefits from Adam's adaptivity, while the highly non-convex and discrete nature of binary weights (via their latent counterparts) can be better navigated by an optimizer like Lion. This contributes to a more robust and effective training process, improving overall stability and performance compared to using a single optimizer for all weight types, as shown in our ablation studies (section 4.5).

Table 1. Number of images in the query and gallery sets of used datasets.

	Training phase		Test phase			
	Drone ↔ Satellite		Drone→Satellite		Satellite→Drone	
Dataset			Query	Gallery	Query	Gallery
University-1652 [12]	37 854	701	37 854	951	701	51 354
SUES-200 [23]	24 000	120	16 000	200	80	40 000

4. Experiments

4.1. Evaluation metrics

We adopt Recall@K (**R@K**) and average precision (**AP**) to evaluate our model, common metrics in cross-view geolocalization. R@K is the ratio of queries for which a correct match is found within the top-K ranked gallery images. Higher R@K indicates better retrieval performance. AP summarizes the precision-recall curve, reflecting the overall quality of the retrieval system. We'll report R@1, R@5, R@10, and AP for a thorough evaluation. While we cannot directly compare R@5 and R@10 with all existing work, these metrics will provide a more complete picture of our results.

To assess the computational complexity of our models, we employ three key metrics. We quantify floating-point operations using **FLOPs** (floating point operations per second [56]), which measures traditional arithmetic computations. For BNNs, we calculate **BinOps** (Binary Operations [48, 57]), specifically counting operations performed on binary values. To provide a unified measure of total computational effort that combines both floating-point and binary operations, we define **Ops** as the sum of FLOPs and BinOps divided by 64 (FLOPs + (BinOps / 64)) [39, 58]. This equivalence, where 64 binary operations are considered equivalent to one 32-bit floating-point operation, aligns with established practices in the field for comparing heterogeneous computations. The FLOPs and BinOps calculations are performed using a modified version of the 'ptflops' library, adapted to accurately support binary operations in accordance with the standards set by previous works [39, 58].

4.2. Implementation details

The proposed method is based on the BNext-tiny and BNext-middle [48] architectures. Data Preprocessing and Augmentation: Input images are resized to 256 × 256 for University-1652 and SUES-200. Images are normalized by subtracting the ImageNet mean and dividing by the ImageNet standard deviation. In training, we perform image augmentation: random padding, random cropping and random flipping.

The training process employs Bi-Gradual Unfreezing (section 3.5) for fine-tuning, with $n_1 = 20$, $n_2 = 20$, and $n_3 = 210$ epochs. Bi-Partitioned Optimization (section 3.6) is used, with Adam for full-precision weights and Lion [55] for (latent) binary weights. For Adam, the learning rate is 0.006, betas (0.9, 0.999), $\epsilon = 1 \cdot 10^{-8}$, and no weight decay. For Lion, the learning rate is 0.001, betas (0.9, 0.99), and no weight decay. All experiments use a batch size of 8 and run for a maximum of 250 epochs. The teacher model for knowledge distillation (section 3.4) is a ConvNext-tiny [53] model trained on the respective dataset. A multistep learning rate scheduler is applied (gamma of 0.1 at epochs [80, 120, 200]). For the first $n_1 + n_2$ epochs (during Bi-Gradual Unfreezing phases 1 and 2), only Adam's scheduler advances. After $n_1 + n_2$ epochs, the Lion optimizer's scheduler starts counting its epochs from zero. We utilized a triplet loss with batch-hard negative mining to train the feature embedding, in alignment with the approach proposed by [17]. **Hardware:** training was performed on NVIDIA RTX 4090 with an Intel Core i9-10 900 K CPU.

4.3. Datasets

We verify robustness on two benchmarks: University-1652 [12] and SUES-200 [23]. Table 1 summarizes their details. They are independent.

The University-1652 [12] dataset contains 1652 locations from 72 universities. The training set has 701 buildings (33 universities), each with 1 satellite view and 54 UAV views. Street-view images are not used. The test set has 951 buildings (remaining 39 universities), with no university overlap between train/test. Sub-tasks: UAV→Satellite (37 854 query drone images, 701 true-matched gallery satellite images, 250 distractors) and Satellite→UAV.

The SUES-200 [23] dataset focuses on scene range impact, with drone and satellite views. It includes drone images at heights: 150 m, 200 m, 250 m, 300 m. It has 200 locations, each with 50 drone images per height and 1 corresponding satellite image (total drone images: 200 × 50 × 4; satellite images: 200). 120 locations are for training, 80 for testing. The gallery set for each sub-task includes test data and training data.

4.4. Results

We compared BiCrossNet's performance on SUES-200 and University-1652 datasets against state-of-the-art methods: SUES-200 [23], LCM [19], LPN [14], FSRA [16], SGM [59], and MCCG [17]. Results are in tables 2 and 3.

SUES-200 dataset: Our method fine-tuned models for each altitude, aligning with [17]. On Drone→Satellite matching (table 2), BiCrossNet exhibits competitive performance despite lower computational cost. At 150 m, R@1 is 69.41%, slightly better than FSRA [16] but with 90× fewer Ops and 4.64× less disk space. Performance improves with altitude, reaching R@1 of 91.72% at 300 m, close to MCCG [17] but with drastically reduced cost. This trend of improving performance with increasing altitude might be attributed to several factors: (1) a wider field of view at higher altitudes may capture more unique contextual information, aiding discrimination despite lower object-level resolution; (2) reduced parallax and a more nadir-like drone view at higher altitudes can decrease the visual domain gap with satellite imagery; (3) the model might be more adept at leveraging global scene characteristics that become more apparent from higher viewpoints. On Satellite→Drone, at 150 m, R@1 is 91.25%, just behind MCCG, but with significantly fewer resources. At 300 m, R@1 is 95.00%, marginally lower than MCCG, but using 67x fewer Ops.

University-1652 dataset: As shown in figure 2 and table 3, BiCrossNet significantly improves the performance-efficiency trade-off. Compared to the full-precision transformer SGM [59], our tiny BiCrossNet achieves comparable/superior performance with 30x fewer Ops and 5.13x less disk space. The BiCrossNet-middle model surpasses SGM in all metrics while using 10x fewer Ops. While not fully matching MCCG's [17] peak accuracy, BiCrossNet-middle operates with approx. 30x fewer Ops.

To complement our quantitative analysis and provide a deeper understanding of BiCrossNet's performance in real-world scenarios, we present qualitative matching examples in figure 3. The figure showcases a challenging false positive case in the Satellite Query to Drone Gallery setting. For gallery ID 118, the model produced an incorrect match, likely influenced by a superficial color similarity between the query and the false positive rather than true structural correspondence. Conversely, in the Drone Query to Satellite Gallery task, the model failed to retrieve the correct match at rank 1 for gallery ID 1247. This misidentification appears to stem from the high visual similarity among parking lots, suggesting that while the model achieves high recall, it could benefit from improved discrimination of stable and semi-stable environmental elements versus more transient or repetitive features. These qualitative examples highlight both the strengths and current limitations of our approach, offering valuable insights into the model's behavior beyond numerical metrics.

4.5. Ablation studies

In this section, we conduct extensive ablation studies to evaluate the impact of various components and modifications of our BiCrossNet model on the University-1652 dataset. We focus on backbone choice, knowledge distillation, the Bi-Gradual Unfreezing strategy, and the Bi-Partitioned Optimization approach.

Backbone choice: We compare BiCrossNet's BNext backbone against other BNN baselines like Binarized ResNet34 (XNOR-Net [35]) and ReCU [45]. Table 8 shows that BNext [48] (both tiny and middle) offers the best trade-off in terms of R@1/AP versus Ops, establishing it as a more suitable BNN architecture for this task.

Knowledge distillation (KD): We evaluate the impact of logit-based KD using a ConvNext-tiny [53] teacher. Table 4 demonstrates that KD significantly improves accuracy (R@1 and AP) for all BNN backbones, particularly enhancing the AP on the challenging Satellite-to-Drone task. This highlights KD's effectiveness in transferring richer supervisory signals to the BNN.

Bi-Gradual Unfreezing Strategy: This technique gradually unfreezes network layers, transitioning from training only the head to fine-tuning all weights. Table 5 compares standard gradual unfreezing [60] with our Bi-Gradual Unfreezing. The results indicate that Bi-Gradual Unfreezing, which distinctly handles full-precision and binary weight unfreezing stages, leads to better final performance (e.g. BNext-tiny + KD achieves 80.12 R@1 with Bi-Gradual vs. 77.22 R@1 with standard gradual unfreezing). This suggests BNNs benefit more from a carefully staged unfreezing process to mitigate overfitting and catastrophic forgetting.

Bi-Partitioned Optimization: We compare our Bi-Partitioned Optimization (Adam for full-precision, Lion for binary weights) against using a single optimizer (Adam, SGD, AdamW, Lion) for all weights. Table 6 shows that Bi-Partitioned Optimization (76.55 R@1) outperforms using only Adam (75.62 R@1) or Lion (75.57 R@1), and significantly outperforms SGD. This supports the hypothesis that different weight types in

Table 2. Comparison on SUES-200 [23]. Ops, FLOPs, BinOps for a single $2 \times 3 \times 256 \times 256$ inference. R@1/5/10 (%), AP (%).

Drone→Satellite																				
							150 m				200 m				250 m				300 m			
Method	Params#	Params(MB)	W/A	BinOps	FLOPs	Ops	R@1	R@5	R@10	AP	R@1	R@5	R@10	AP	R@1	R@5	R@10	AP	R@1	R@5	R@10	AP
SUES-200 [23]	51.02 M	204.07	32/32	—	24.28G	24.28G	55.65	—	—	61.92	66.78	—	—	71.55	72.00	—	—	76.43	74.05	—	—	78.26
LCM [19]	25.56 M	102.24	32/32	—	24.28G	24.28G	43.42	—	—	49.65	49.42	—	—	55.91	54.47	—	—	60.31	60.43	—	—	65.78
LPN [14]	27.78 M	111.12	32/32	—	36.77G	36.77G	61.58	—	—	67.23	70.85	—	—	75.96	80.38	—	—	83.80	81.47	—	—	84.53
FSRA [16]	24.42 M	97.67	32/32	—	35.44G	35.44G	68.25	—	—	73.45	83.00	—	—	85.99	90.68	—	—	92.27	91.95	—	—	93.46
MCCG [17]	29.96 M	119.83	32/32	—	26.37G	26.37G	82.22	—	—	85.47	89.38	—	—	91.41	93.82	—	—	95.04	95.07	—	—	96.20
Our (tiny)	32.62 M	21.01	1/1	14.13G	0.17G	0.39G	69.41	77.41	81.41	75.01	80.10	86.10	89.10	83.97	88.85	93.35	95.35	90.95	91.72	95.22	96.72	93.52
Satellite→Drone																				
							150 m				200 m				250 m				300 m			
Method	Params#	Params(MB)	W/A	BinOps	FLOPs	Ops	R@1	R@5	R@10	AP	R@1	R@5	R@10	AP	R@1	R@5	R@10	AP	R@1	R@5	R@10	AP
SUES-200 [23]	25.65 M	204.07	32/32	—	24.28G	24.28G	75.00	—	—	55.46	85.00	—	—	66.05	86.25	—	—	69.94	88.75	—	—	74.46
LCM [19]	25.65 M	102.24	32/32	—	24.28G	24.28G	57.50	—	—	38.11	68.75	—	—	49.19	72.50	—	—	47.94	75.00	—	—	59.36
LPN [14]	27.78 M	111.12	32/32	—	36.77G	36.77G	83.75	—	—	66.78	88.75	—	—	75.01	92.50	—	—	81.34	92.50	—	—	85.72
FSRA [16]	24.42 M	97.67	32/32	—	35.44G	35.44G	83.75	—	—	76.67	90.00	—	—	85.34	93.75	—	—	90.17	95.00	—	—	92.03
MCCG [17]	29.96 M	119.83	32/32	—	26.37G	26.37G	93.75	—	—	89.72	93.75	—	—	92.21	96.25	—	—	96.14	98.75	—	—	96.64
Our (tiny)	32.62 M	21.01	1/1	14.13G	0.17G	0.39G	91.25	94.75	96.25	77.01	96.25	98.25	99.25	88.52	95.00	97.50	98.70	91.41	95.00	97.50	98.70	93.08

Table 3. Comparison on University-1652. Ops, FLOPs, BinOps for a single $2 \times 3 \times 256 \times 256$ inference. R@1/5/10 (%), AP (%).

Drone → Satellite										
Method	Params#	Params(MB)	W/A	BinOps	FLOPs	Ops	R@1	R@5	R@10	AP
LCM [19]	25.56 M	102.24	32/32	—	7G	7G	66.65	—	—	70.82
LPN [14]	25.56 M	102.24	32/32	—	7G	7G	75.93	—	—	79.14
LPN [14] with Resnet-101	45 M	180	32/32	—	14.3G	14.3G	76.13	—	—	79.29
SGM [59]	28 M	112	32/32	—	10.8G	10.8G	82.14	—	—	84.72
MCCG [17]	30.85 M	123.4	32/32	—	29.37G	29.37G	89.64	—	—	91.32
Our (tiny)	33.53 M	21.81	1/1	12.59G	0.16G	0.36G	80.55	87.85	90.15	83.48
Our (middle)	138.05 M	92.23	1/1	58.88G	0.38G	1.3G	86.23	91.53	93.33	87.29
Satellite → Drone										
Method	Params#	Params(MB)	W/A	BinOps	FLOPs	Ops	R@1	R@5	R@10	AP
LCM [19]	25.56 M	102.24	32/32	—	7G	7G	79.89	—	—	65.38
LPN [14]	25.56 M	102.24	32/32	—	7G	7G	85.16	—	—	74.79
LPN [14] with Resnet-101	45 M	180	32/32	—	14.3G	14.3G	85.45	—	—	75.45
SGM [59]	28 M	112	32/32	—	10.8G	10.8G	88.16	—	—	81.81
MCCG [17]	30.85 M	123.4	32/32	—	29.37G	29.37G	94.30	—	—	89.39
Our (tiny)	33.53 M	21.81	1/1	12.59G	0.16G	0.36G	90.01	95.22	97.13	80.16
Our (middle)	138.05 M	92.23	1/1	58.88G	0.38G	1.3G	92.82	96.12	97.42	84.41

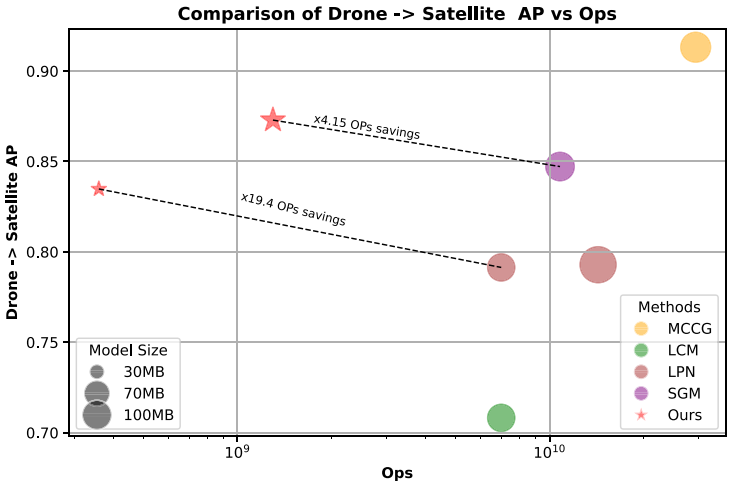

Figure 2. Comparison of state-of-the-art cross-view geolocalization methods with BiCrossNet (stars) on University-1652 (Drone-to-Satellite task). The chart displays average precision (AP), inference operations (Ops), and model size. **Ops (operational volume) is a unified metric for computational cost, defined as FLOPs + (BinOps / 64), where 64 binary operations are considered equivalent to one 32-bit floating-point operation.** Ops are measured for a single inference of a 2 × 3 × 256 × 256 image pair. BiCrossNet demonstrates a significantly better trade-off between AP and Ops.

Figure 3. Qualitative matching examples of BiCrossNet on the University-1652 dataset.

BNNs benefit from specialized optimizers. Figure 4 illustrates weight flip behavior. Our Bi-Partitioned method (Lion_ADAM) maintains a healthy flipping rate, allowing better exploration of the loss landscape. Using Lion alone for all weights led to gradient explosion at one point (not shown in final epochs of this plot

Table 4. Knowledge distillation (KD) impact on University-1652. Ops for a $2 \times 3 \times 256 \times 256$ inference. R@1 (%), AP (%).

Ablation	Variant	Params	Ops	Drone → Satellite		Satellite → Drone	
				R@1	AP	R@1	AP
Baseline (No KD)	ReCU [45]	23.72 M	0.53G	55.29	60.16	75.46	54.15
	BNext-tiny [48]	33.53 M	0.36G	72.56	76.39	85.16	41.33
	BNext-middle [48]	138.05 M	1.3G	78.31	82.87	88.52	46.11
With KD	ReCU [45] + KD	23.72 M	0.53G	60.36	65.09	78.89	60.30
	BNext-tiny [48] + KD	33.53 M	0.36G	75.62	79.10	87.59	74.77
	BNext-middle [48] + KD	138.05 M	1.3G	83.04	85.51	91.58	82.93

Table 5. Ablation study for unfreezing strategy on University-1652 (BNext-tiny + KD). R@1 (%), AP (%).

Ablation	Variant	Drone → Satellite		Satellite → Drone	
		R@1	AP	R@1	AP
Baseline	BNext-tiny [48] + KD (No Gradual Unfreeze)	75.62	79.10	87.59	74.77
	+ Gradual Unfreezing [60]	77.22	81.51	86.58	78.93
Our method	+ Bi-Gradual Unfreezing	80.12	82.78	89.69	79.81

Table 6. Ablation study for optimizer choice on University-1652 (BNext-tiny + KD + Bi-Gradual Unfreezing). R@1 (%), AP (%).

Ablation	Optimizer(s)	Drone → Satellite		Satellite → Drone	
		R@1	AP	R@1	AP
Single optimizer	Adam [54] (for all weights)	75.62	79.10	87.59	74.77
	SGD [61] (for all weights)	71.31	73.68	84.87	71.58
	AdamW [62] (for all weights)	73.28	75.66	86.16	72.49
	Lion [55] (for all weights)	75.57	80.71	88.85	74.69
Our method	Bi-Partitioned (Adam + Lion)	76.55	81.48	89.01	76.16

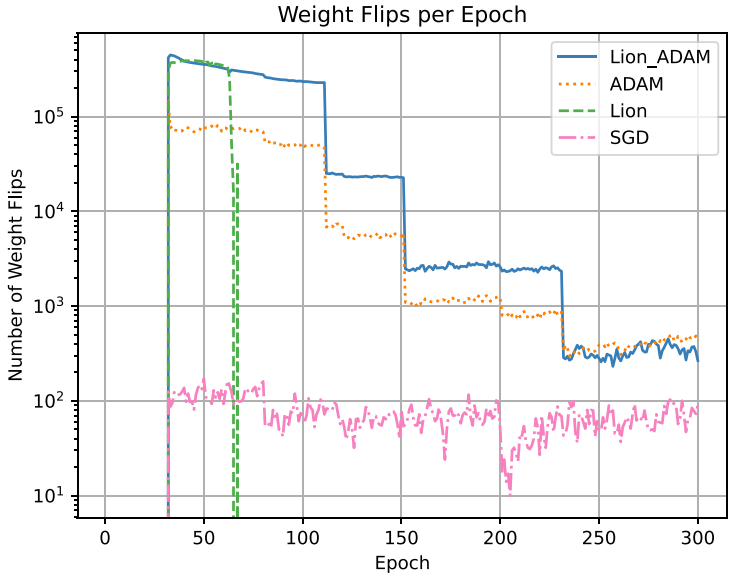

Figure 4. The number of weight flips per epoch during training on SUES-200 [23] for different optimizer strategies. A flip occurs when a weight changes its sign. Bi-Partitioned Optimization (Lion_ADAM) shows a sustained, healthy flipping rate. Using Lion alone for all weights led to a gradient explosion at a certain point during training (indicated by the abrupt termination of its visible line before other methods complete more epochs stably).

for clarity after explosion), due to its sign-based updates which can be aggressive with high learning rates. The dual approach avoids this while harnessing Lion's strengths for binary weights.

Bi-Gradual Unfreezing Epochs To further validate the effectiveness of our Bi-Gradual Unfreezing method and to justify our choice of epoch distribution for each fine-tuning stage, we conducted a detailed ablation

Table 7. Configuration used in 5. N_1 represents epochs for Head Fine-tuning, N_2 for Partial Backbone Unfreezing, and the remaining epochs from Total are for Full Backbone Unfreezing.

ID	N_1 (Epochs)	N_2 (Epochs)	Stage 3 (Epochs)	Total (Epochs)
A	20	20	210	250
B	20	10	220	250
C	20	30	200	250
D	20	20	160	200
E	20	20	260	300
F	10	20	220	250
G	0	0	250	250

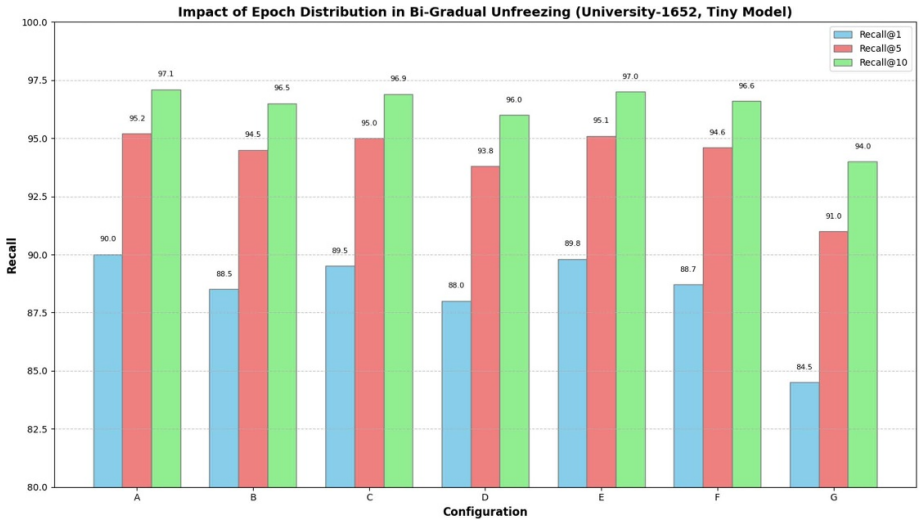

Figure 5. Impact of Epoch Distribution in Bi-Gradual Unfreezing on University-1652 Dataset (Tiny Model) in the setting of Satellite → Drone. Performance (R@1, R5, R@10) is shown for various configurations of epoch allocation across the three Bi-Gradual Unfreezing stages.

Table 8. Backbone choice comparison on University-1652. Ops for a $2 \times 3 \times 256 \times 256$ inference. R@1 (%), AP (%).

				Drone → Satellite		Satellite → Drone	
Ablation	Variant	Params	Ops	R@1	AP	R@1	AP
Baseline	Binarized ResNet34 (XNOR-Net [35])	23.68 M	0.50G	31.71	36.33	41.81	45.23
	ReCU [45]	23.72 M	0.53G	55.29	60.16	75.46	54.15
	BNext-tiny [48]	33.53 M	0.36G	72.56	76.39	85.16	41.33
	BNext-middle [48]	138.05 M	1.3G	78.31	82.87	88.52	46.11

study. This study, performed on the University-1652 dataset with our BiCrossNet-Tiny model, investigates how varying the number of epochs for Head Fine-tuning (N_1), Partial Backbone Unfreezing (N_2), and the Total Epochs (which impacts Full Backbone Unfreezing) affects the overall geolocalization performance.

Table 7 presents the configurations tested.

The results, summarized in figure 5, demonstrate the impact of epoch allocation on the final cross-view geolocalization performance. Config A achieved the optimal balance between performance and computational efficiency. Configurations with fewer epochs in earlier stages (e.g. Config F for Stage 1, or Config B for Stage 2) often resulted in suboptimal performance, indicating insufficient adaptation of the unfrozen layers. Conversely, extending the training for more epochs beyond our selected values (e.g. Config C for Stage 2, or Config E for Stage 3) did not yield significant improvements in Recall metrics but substantially increased training time and computational cost. Furthermore, training the model without any gradual unfreezing (Config G) resulted in noticeably lower performance, underscoring the effectiveness of our proposed Bi-Gradual strategy.

This empirical analysis confirms that our selected epoch distribution enables the tiny model to converge effectively, extracting robust features and achieving stable performance on the University-1652 dataset, representing an optimized trade-off informed by extensive preliminary experiments.

On-hardware evaluation: We evaluated inference efficiency on a Banana Pi M5 using the Larq Library, following the methodology of [48]. With a BNext-tiny model and a two-image batch, our optimized ARM implementation achieved a 2.3× speedup (1.15 s) over the PyTorch baseline (2.54 s). A standalone 1-bit convolution layer demonstrated a 22.5× speedup, highlighting the potential of binarized operations. However, the overall model's acceleration was limited by the reliance on PyTorch's C++ API, which introduces floating-point overhead and lacks dedicated BNN kernels, preventing the full realization of BNN efficiency.

5. Conclusion

This paper has introduced BiCrossNet, an innovative BNN-based approach for cross-view geolocalization that significantly reduces computational complexity and memory requirements while maintaining competitive performance. Key contributions include the Bi-Gradual Unfreezing method for enhanced transfer learning, the Bi-Partitioned Optimization strategy for improved training stability, and logit-based knowledge distillation to augment standard losses. Experiments on University-1652 and SUES-200 demonstrate BiCrossNet's superior efficiency-performance trade-off, establishing a new benchmark. Despite its significant improvements, BiCrossNet has limitations. While this work aims for hardware-agnostic efficiency gains in terms of operations and memory, deploying BiCrossNet on specific low-power hardware platforms would require further engineering to fully realize theoretical benefits in practical latency and energy consumption.

Future work will focus on further optimizing the BNN training process, exploring techniques to reduce quantization errors even more, and investigating alternative BNN architectures tailored for geolocalization. Exploring the fusion of multi-band satellite data (e.g. infrared) could also offer performance enhancements. Developing BNNs that are even more robust to the challenges of cross-view matching will be essential for widespread UAV deployment.

Data availability statement

All data that support the findings of this study are included within the article (and any supplementary files).

ORCID iDs

Federico Fontana 0009-0007-0437-7832
Thomas Jantos 0009-0007-6066-0931
Jan Steinbrener 0000-0002-2465-2527
Bernhard Rinner 0000-0002-8793-3828

References

[1] Rinner B, Bettstetter C, Hellwagner H and Weiss S 2021 Multidrone systems: more than the sum of the parts *Computer* **54** 34–43
[2] Dissanayaka D, Wanasinghe T R, De Silva O, Jayasiri A and Mann G K I 2023 Review of navigation methods for UAV-based parcel delivery *IEEE Trans. Autom. Sci. Eng.* **21** 1068–82
[3] Sorbelli F B, Corò F, Palazzetti L, Pinotti C M and Rigoni G 2023 How the wind can be leveraged for saving energy in a truck-drone delivery system *IEEE Trans. Intell. Transp. Syst.* **24** 4038–49
[4] Khan M A, Ectors W, Bellemans T, Janssens D and Wets G 2017 UAV-based traffic analysis: a universal guiding framework based on literature survey *Transp. Res. Proc.* **22** 541–50
[5] Wang S, Jiang F, Zhang B, Rui M and Hao Q 2019 Development of UAV-based target tracking and recognition systems *IEEE Trans. Intell. Transp. Syst.* **21** 3409–22
[6] Greenwood W W, Lynch J P and Zekkos D 2019 Applications of UAVS in civil infrastructure *J. Infrastruct. Syst.* **25** 04019002
[7] Herwitz S R *et al* 2004 Imaging from an unmanned aerial vehicle: agricultural surveillance and decision support *Comput. Electron. Agric.* **44** 49–61
[8] Deng L, Mao Z, Li X, Hu Z, Duan F and Yan Y 2018 UAV-based multispectral remote sensing for precision agriculture: a comparison between different cameras *ISPRS J. Photogramm. Remote Sens.* **146** 124–36
[9] Rokhmana C A 2015 The potential of UAV-based remote sensing for supporting precision agriculture in indonesia *Proc. Environ. Sci.* **24** 245–53
[10] Rieke M, Foerster T, Geipel J and Prinz T 2012 High-precision positioning and real-time data processing of UAV-systems *Int. Arch. Photogram. Remote Sens. Spatial Inf. Sci.* **38** 119–24
[11] Zimmermann F, Eling C, Klingbeil L and Kuhlmann H 2017 Precise positioning of UAVS–dealing with challenging RTK-GPS measurement conditions during automated UAV flights *ISPRS Ann. Photogram. Remote Sens. Spatial Inf. Sci.* **4** 95–102

[12] Zheng Z, Wei Y and Yang Y 2020 University-1652: a multi-view multi-source benchmark for drone-based geo-localization *Proc. 28th ACM Int. Conf. on Multimedia* pp 1395–403
[13] Zhu R, Yang M, Yin L, Wu F and Yang Y 2023 UAV's status is worth considering: a fusion representations matching method for geo-localization *Sensors* **23** 720
[14] Wang T, Zheng Z, Yan C, Zhang J, Sun Y, Zheng B and Yang Y 2022 Each part matters: local patterns facilitate cross-view geo-localization *IEEE Trans. Circuits Syst. Video Technol.* **32** 867–79
[15] Zheng Z, Shi Y, Wang T, Liu J, Fang J, Wei Y and Chua T-seng 2023 Uavm'23: 2023 workshop on uavs in multimedia: capturing the world from a new perspective *Proc. 31st ACM Int. Conf. on Multimedia* pp 9715–7
[16] Dai M, Jianhong H, Zhuang J and Zheng E 2021 A transformer-based feature segmentation and region alignment method for UAV-view geo-localization *IEEE Trans. Circuits Syst. Video Technol.* **32** 4376–89
[17] Shen T, Wei Y, Kang L, Wan S and Yang Y-H 2023 Mccg: a convnext-based multiple-classifier method for cross-view geo-localization *IEEE Trans. Circuits Syst. Video Technol.* **34** 1456–68
[18] Lin J, Zheng Z, Zhong Z, Luo Z, Li S, Yang Y and Sebe N 2022 Joint representation learning and keypoint detection for cross-view geo-localization *IEEE Trans. Image Process.* **31** 3780–92
[19] Ding L, Zhou J, Meng L and Long Z 2020 A practical cross-view image matching method between UAV and satellite for UAV-based geo-localization *Remote Sens.* **13** 47
[20] Sun B, Liu G and Yuan Y 2023 F3-net: multi-view scene matching for drone-based geo-localization *IEEE Trans. Geosci. Remote Sens.* **61** 1–11
[21] Wang T, Zheng Z, Sun Y, Chua T-S, Yang Y and Yan C 2022 Multiple-environment self-adaptive network for aerial-view geo-localization (arXiv:2204.08381)
[22] Zhao H, Ren K, Yue T, Zhang C and Yuan S 2024 Transfg: a cross-view geo-localization of satellite and uavs imagery pipeline using transformer-based feature aggregation and gradient guidance *IEEE Trans. Geosci. Remote Sens.* **62** 1–12
[23] Zhu R, Yin L, Yang M, Wu F, Yang Y and Hu W 2023 Sues-200: a multi-height multi-scene cross-view image benchmark across drone and satellite *IEEE Trans. Circuits Syst. Video Technol.* **33** 4825–39
[24] Castaldo F, Zamir A, Angst R, Palmieri F and Savarese S 2015 Semantic cross-view matching *Proc. IEEE Int. Conf. on Computer Vision Workshops* pp 9–17
[25] Lin T-Y, Belongie S and Hays J 2013 Cross-view image geolocalization *Proc. IEEE Conf. on Computer Vision and Pattern Recognition* pp 891–8
[26] Senlet T and Elgammal A 2011 A framework for global vehicle localization using stereo images and satellite and road maps *Proc. IEEE Int. Conf. on Computer Vision Workshops* (IEEE) pp 2034–41
[27] Workman S and Jacobs N 2015 On the location dependence of convolutional neural network features *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition Workshops* pp 70–78
[28] Zhai M, Bessinger Z, Workman S and Jacobs N 2017 Predicting ground-level scene layout from aerial imagery *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition* pp 867–75
[29] Liu L and Li H 2019 Lending orientation to neural networks for cross-view geo-localization *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition* pp 5624–33
[30] Chopra S, Hadsell R and LeCun Y 2005 Learning a similarity metric discriminatively, with application to face verification *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition* vol 1 (IEEE) pp 539–46
[31] Arandjelovic R, Gronat P, Torii A, Pajdla T and Sivic J 2016 Netvlad: Cnn architecture for weakly supervised place recognition *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition* pp 5297–307
[32] Shi Y, Liu L, Yu X and Li H 2019 Spatial-aware feature aggregation for image based cross-view geo-localization *Advances in Neural Information Processing Systems* vol 32
[33] Dosovitskiy A *et al* 2020 An image is worth 16x16 words: Transformers for image recognition at scale *Proc. Int. Conf. on Learning Representations*
[34] Courbariaux M, Hubara I, Soudry D, El-Yaniv R and Bengio Y 2016 Binarized neural networks: training deep neural networks with weights and activations constrained to +1 or −1 (arXiv:1602.02830)
[35] Rastegari M, Ordonez V, Redmon J and Farhadi A 2016 Xnor-net: imagenet classification using binary convolutional neural networks *Proc. European Conf. on Computer Vision (ECCV)* pp 525–42
[36] Lin X, Zhao C and Pan W 2017 Towards accurate binary convolutional neural network (arXiv:1711.11294)
[37] He K, Zhang X, Ren S and Sun J 2016 Deep residual learning for image recognition *Proc. IEEE Conf. on Computer Vision and Pattern Recognition* pp 770–8
[38] Huang G, Liu Z, Maaten L V D and Weinberger K Q 2017 Densely connected convolutional networks *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition* pp 4700–8
[39] Liu Z, Wu B, Luo W, Yang X, Liu W and Cheng K-T 2018 Bi-real net: enhancing the performance of 1-bit cnns with improved representational capability and advanced training algorithm *Proc. European Conf. on Computer Vision* pp 722–37
[40] Bethge J, Yang H, Bornstein M and Meinel C 2019 Binarydensenet: developing an architecture for binary neural networks *Proc. IEEE/CVF Int. Conf. on Computer Vision Workshops* pp 0–0
[41] Qin H, Gong R, Liu X, Shen M, Wei Z, Yu F and Song J 2020 Forward and backward information retention for accurate binary neural networks *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition* pp 2250–9
[42] Liu Z, Shen Z, Savvides M and Cheng K-T 2020 Reactnet: towards precise binary neural network with generalized activation functions *Proc. European Conf. on Computer Vision* (Springer) pp 143–59
[43] Lin M, Ji R, Xu Z, Zhang B, Wang Y, Wu Y, Huang F and Lin C-W 2020 Rotated binary neural network *Advances in Neural Information Processing Systems* vol 33
[44] Lin M, Ji R, Xu Z, Zhang B, Chao F, Xu M, Lin C-W and Shao L 2021 Siman: sign-to-magnitude network binarization (arXiv:2102.07981)
[45] Xu Z, Lin M, Liu J, Chen J, Shao L, Gao Y, Tian Y and Ji R 2021 Recu: reviving the dead weights in binary neural networks (arXiv:2103.12369)
[46] Tu Z, Chen X, Ren P and Wang Y 2022 Adabin: improving binary neural networks with adaptive binary sets (arXiv:2208.08084)
[47] Wang Z, Wu Z, Lu J and Zhou J 2020 Bidet: an efficient binarized object detector *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition* pp 2049–58
[48] Guo N, Bethge J, Meinel C and Yang H 2022 Join the high accuracy club on imagenet with a binary neural network ticket (arXiv:2211.12933)

[49] Krizhevsky A, Sutskever I and Hinton G E 2012 Imagenet classification with deep convolutional neural networks *Advances in Neural Information Processing Systems* vol 25
[50] Hu J, Shen Li and Sun G 2018 Squeeze-and-excitation networks *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition* pp 7132–41
[51] Bengio Y, Léonard N and Courville A 2013 Estimating or propagating gradients through stochastic neurons for conditional computation (arXiv:1308.3432)
[52] Hoffer E and Ailon N 2015 Deep metric learning using triplet network *Similarity-Based Pattern Recognition: 3rd Int. Workshop, SIMBAD 2015, (Copenhagen, Denmark, 12 October–14 October 2015) Proc.* vol 3 (Springer) pp 84–92
[53] Liu Z, Mao H, Wu C-Y, Feichtenhofer C, Darrell T and Xie S 2022 A convnet for the 2020s *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition* pp 11976–86
[54] Kingma D P and Ba J 2014 Adam: a method for stochastic optimization (arXiv:1412.6980)
[55] Chen X *et al* 2024 Symbolic discovery of optimization algorithms *Advances in Neural Information Processing Systems* vol 36
[56] Desislavov R, Martínez-Plumed F and Hernández-Orallo J 2021 Compute and energy consumption trends in deep learning inference (arXiv:2109.05472)
[57] Zhu S, Dong X and Su H 2019 Binary ensemble neural network: More bits per network or more networks per bit? *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition* pp 4923–32
[58] Shen J, Fu Y, Wang Y, Xu P, Wang Z and Lin Y 2020 Fractional skipping: towards finer-grained dynamic cnn inference (arXiv:2001.00705)
[59] Zhuang J, Chen X, Dai M, Lan W, Cai Y and Zheng E 2022 A semantic guidance and transformer-based matching method for UAVS and satellite images for uav geo-localization *IEEE Access* **10** 34277–87
[60] Howard J and Ruder S 2018 universal language model fine-tuning for text classification *Proc. 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* ed I Gurevych and Y Miyao (Association for Computational Linguistics) pp 328–39
[61] Robbins H E 1951 A stochastic approximation method *Ann. Math. Stat.* **22** 400–7
[62] Loshchilov I and Hutter F 2017 Decoupled weight decay regularization (arXiv:1711.05101)