Scalable Satellite Imagery Analysis: A Cascade Framework for Sparse Target Detection

Arvind Manivannan
University of Washington
arvinm3@uw.edu

Tarun Narayanan Venkatachalam
University of Washington
tarunnv@uw.edu

Yanlin Huang
University of Washington
yanlinh@cs.washington.edu

Favyen Bastani
Allen Institute for AI
favyenb@allenai.org

Abstract

Remote sensing is a crucial tool for monitoring events affecting climate change, such as tracking forest loss, identifying pollution sources, and monitoring the deployment of renewable energy infrastructure. However, applying state-of-the-art deep learning models to monitor the entire Earth is expensive. In this paper, we propose a cascade framework to reduce this cost: we apply a small MLP on precomputed embeddings of each image patch to serve as a preliminary filter, identifying key patches that warrant further examination by more resource-intensive deep models. Our approach reduces per-task inference runtime by 5x with a $< 1\%$ impact on accuracy. By reducing inference cost, our method enables nonprofits and other organizations with limited resources to scale monitoring efforts to more environmental and conservation applications.

1 Introduction

Remote sensing methods have been developed to provide invaluable insights for environmental applications, including for tracking deforestation to preserve carbon sinks [1, 2], detecting illegal fishing to protect marine ecosystems [3, 4], and monitoring renewable energy infrastructure to assess clean energy transition progress [5]. However, global satellite imagery analysis presents significant challenges due to the vast volumes of data involved. For example, since 2015, the European Space Agency's Sentinel-2 mission has been capturing 10 m/pixel images across the Earth. Processing monthly global Sentinel-2 mosaics from 2015-2024 with a Swin-Base model [6] would cost $25,000 on AWS. This is exacerbated as more deep models are developed that either improve on existing models or address new applications. For instance, after developing ten deep models, the cumulative cost balloons to $250,000.

Many of these tasks involve objects or events that occur relatively sparsely across global satellite imagery. For instance, while forest loss is progressing at a tremendous pace, it typically affects a small percentage of global forest area in a given year. Similarly, objects like wind turbines and offshore oil platforms appear in a small fraction ($\sim 1\%$) of global image patches. One way to reduce cost is to develop heuristic filters that restrict where the deep model is applied, e.g., wind turbines do not appear in residential areas. However, such heuristic filters offer a limited speedup, are error-prone, and are not easily adaptable to new tasks.

To address these challenges, we propose a cascade framework for globally applying remote sensing models with sparsely occurring targets. Our approach replaces heuristic filters with small learned models that predict whether a patch contains at least one object instance based on precomputed image embeddings. We evaluate our method on two remote sensing tasks, and find that our method provides a substantial 5x reduction in per-task costs with an error rate of $\leq 1.2\%$ across both tasks. Thus,

Tackling Climate Change with Machine Learning: workshop at NeurIPS 2022.

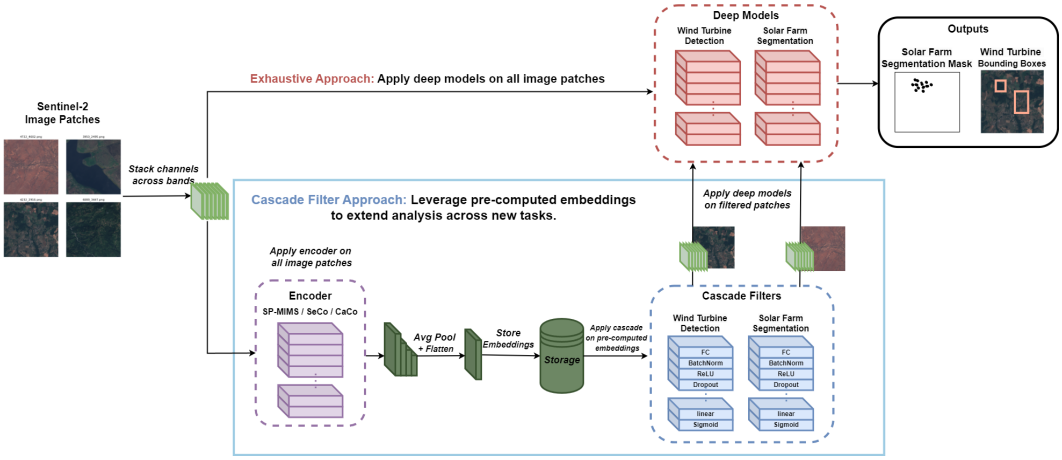

Figure 1: Rather than exhaustively apply a compute-intensive, deep remote sensing model on every image patch, our cascade framework employs a small MLP trained to classify which patches contain at least one instance from precomputed embeddings. During inference, we prune patches where the MLP is confident that there are 0 instances, and only apply the deep model on the remaining patches.

our method significantly accelerates satellite imagery analysis, addressing a critical bottleneck in large-scale environmental monitoring.

2 Related Work

Cascade Filters for Video Analytics. Several approaches have been proposed for optimizing the analysis of large-scale video data. NoScope [7] achieves a 3x speedup in binary classification queries by employing a cascade of specialized models, including fast shallow CNNs trained on individual object categories and difference detectors that identify temporal changes between frames. BlazeIt [8] extents this approach for aggregate and limit queries. In our work, we adapt these methods for the remote sensing domain.

Remote Sensing Foundation Models. Several foundation models have been developed for remote sensing, including SatlasPretrain [5], SeCo [9], and CACo [10]. These models compute strong representations of image patches; while linear probes over the embeddings can provide accurate outputs for many classification tasks, they are insufficient for most localization tasks.

Applications in Renewable Energy Monitoring. Recent advancements in remote sensing for renewable energy applications highlight the broader impact potential of the proposed cascade framework. For example, methods for mapping global rooftop solar potential now employ deep learning models to estimate digital surface models and segment roofs with high accuracy, enabling global scalability [11]. Similarly, enhanced U-Net architectures have been applied to detect photovoltaic facilities from multi-spectral satellite data, achieving fine-grained segmentation even in complex environments [12]. Incorporating the proposed cascade framework into these workflows could filter out regions unlikely to host relevant features, such as buildings or photovoltaic cells, thereby improving efficiency and reducing computational costs for large-scale analysis.

3 Methodology

In this section, we detail our cascade inference approach that reduces the cost of detection or segmentation in remote sensing images. Rather than applying an expensive deep model on every image patch, we employ a fast multi-layer perceptron (MLP) over precomputed image embeddings to first identify patches that most likely contain zero targets. We then skip applying the deep model on those patches. Figure 1 summarizes our approach.

Using a pretrained encoder E, we first compute an embedding $x_{i,j}^t = E(I_{i,j}^t)$ for each image patch $I_{i,j}^t$ over space and time. When a user develops a deep model D for a new task involving sparse targets, such as wind turbine detection, we train an MLP on a random sample of embeddings. Specifically, the MLP f inputs an embedding $x_{i,j}^t$ and estimates the probability $p_{i,j}^t = f(x_{i,j}^t)$ that $I_{i,j}^t$ contains a relevant object. During training of the MLP, the label $y_{i,j}^t$ is 0 if $D(I_{i,j}^t) = \emptyset$, i.e. if the deep model finds no instances in the corresponding image patch, and 1 otherwise.

We tune a confidence threshold h on another random sample of embeddings, X_{val}, to achieve a fixed target recall, such as 99%. We then process the remaining image patches: for each image $I_{i,j}^t$, if $f(x_{i,j}^t) < h$, we predict that there are zero instances in the image; otherwise, we retrieve $I_{i,j}^t$ from storage and compute $D(I_{i,j}^t)$.

Our cascade inference approach reduces overall execution time because applying the MLP on precomputed embeddings is orders of magnitude faster than applying the deep model on the image patch. Although the embedding is insufficient for localizing objects or even accurately predicting whether every patch contains an instance, there are many patches that clearly contain no instances based on the embedding alone. For example, even weak embeddings should be sufficient to conclude with confidence that patches covered uniformly by forest or water do not contain wind turbines. With a strong pretrained encoder for remote sensing, many more patches can be confidently excluded.

Below, we detail each component of our approach.

Deep model. We assume that the user has developed a new deep model D that inputs an image patch and outputs bounding boxes or segmentation masks, depending on the task. We do not make assumptions about the architecture of D. We do, however, assume that the task exhibits sparse targets. This is common in remote sensing: wind turbines, solar farms, offshore platforms, and annual forest loss events all occur in fewer than 1% of image patches [5].

Encoder. We compute embeddings $x_{i,j}^t = E(I_{i,j}^t)$ that provide strong representations of image patches by implementing E with foundation models. From our ablation study in Section A.1, we find that the SatlasPretrain multi-image multi-spectral (SP-MIMS) model provides the best performance.

MLP Training. Our classification MLP inputs an embedding $x_{i,j}^t$ computed by E, and outputs a probability $p_{i,j}^t$ through a sigmoid activation. We use a small sample of the embeddings $X_{\text{train}} = \{x_{i_0,j_0}^{t_0}, \ldots, x^{t_N}, {}_{i_N,j_N}\}$ to train the MLP. We apply D on each corresponding image patch to derive labels; specifically, $y_{i,j}^t = 0$ if $D(I_{i,j}^t) = \emptyset$ and $y_{i,j}^t = 1$ otherwise, i.e., the labels reflect whether a patch contains at least one instance, and are supervised by the deep model. We train the MLP on pairs $(x_{i,j}^t, y_{i,j}^t)$ using cross entropy loss. Additional details can be found in Section A.3.

Confidence threshold tuning. The confidence threshold h dictates whether an image patch $I_{i,j}^t$ is processed further by the deep model: if the probability output by the MLP $p_{i,j}^t < h$, then we do not apply D on $I_{i,j}^t$, thereby minimizing unnecessary processing.

Different values of h offer a tradeoff between precision and recall. In the classification task, we define a true positive as a patch $I_{i,j}^t$ such that $D(I_{i,j}^t) \neq \emptyset$ and $p_{i,j}^t \geq h$, i.e., the MLP correctly predicts that the patch contains an instance. A false positive occurs when $p_{i,j}^t \geq h$ but $D(I_{i,j}^t) = \emptyset$, and a false negative occurs when $p_{i,j}^t < h$ but $D(I_{i,j}^t) \neq \emptyset$. While a false positive merely affects runtime, since we will observe that there are no instances in the patch once we apply D, a false negative affects accuracy.

To set h, we first assume that the user configures a recall target R for the MLP. For example, $R = 99\%$ means that 99% of patches where D would have predicted an instance are recovered correctly, while 1% of patches with instances are missed. Then, we establish a validation set X_{val} similar to the training set. We set h so that the MLP f achieves a recall of R in X_{val}. In this way, we provide the maximum speedup achievable for a fixed tolerable accuracy reduction.

4 Evaluation

We train and evaluate deep models on two human-labeled datasets X_D: (1) a wind turbine dataset, consisting of 13,955 Sentinel-2 patches paired with wind turbine bounding box labels; and (2) a solar farm dataset, consisting of 3048 Sentinel-2 patches paired with solar farm segmentation labels.

Table 1: Performance metrics for five backbones across two target tasks are presented, including runtime and F1 scores for both exhaustive and cascade approaches, the latter tuned to fixed recalls of 99% and 95%. On Swin-v2-Base, which provides the highest accuracy, the cascade approach reduces per-task runtime by 5x with a $\leq 1\%$ accuracy reduction and by 6x with a $\leq 5\%$ accuracy reduction for the respective tasks.

	Wind Turbine				Solar Farm
Deep Model Backbone	ResNet50	ResNet152	Swinv2_tiny	Swinv2_base	Swinv2_base
Global target occurrence (%)	1.3	1.36	1.26	1.11	1.24
Deep model throughput (img/sec)	221	160	162	87	75
Exhaustive Approach					
Fixed runtime (hours)	0	0	0	0	0
Per-task runtime (hours)	136.04	190.58	187.59	339.13	381.82
Runtime for 10 tasks (days)	56.68	79.408	78.16	141.3	159.09
Accuracy (F1 score)	77.73	81.16	80.65	81.28	90.65
Cascade Approach (C_{99})					
Fixed runtime (hours)	316.4	316.4	316.4	316.4	316.4
Per-task runtime (hours)	49.58	68.83	55.5	80.51	88.85
Runtime for 10 tasks (days)	33.84	41.86	36.31	46.73	50.2
Accuracy (F1 score)	76.89	80.29	79.69	80.74	89.73
Cascade Approach (C_{95})					
Fixed runtime (hours)	316.4	316.4	316.4	316.4	316.4
Per-task runtime (hours)	34.06	51.99	42.92	58.78	59.79
Runtime for 10 tasks (days)	27.38	34.85	31.07	37.68	38.1
Accuracy (F1 score)	73.8	77.1	76.54	77.13	85.91

Methods. We contrast our cascade approach with the exhaustive method, which involves applying the deep model on every image patch. We assess the detection runtime using 4 GPUs on 100 million image patches, equivalent to 5 global Sentinel-2 mosaics. For each deep model backbone, we train a corresponding MLP for the cascade approach on a sample of 30,000 image patches, with labels derived from the deep model outputs. We train and evaluate two cascades, C_{99} and C_{95}, which we tune to have fixed recalls of 99% and 95% on X_{train}.

Metrics. We evaluate the methods in terms of runtime and F1 score. We breakdown runtime into two categories: fixed runtime (remains constant as the number of tasks increases) and per-task runtime. In the exhaustive approach, fixed runtime is zero since there are no task-agnostic components; per-task runtime consists of training and inference of the deep model. For our cascade approach, the fixed runtime includes computing embeddings for all patches, while the per-task runtime covers training the deep model and MLP, applying the MLP on every embedding, and applying the deep model on the subset of patches filtered by the cascade. Additional details can be found in Section A.5.

Results. Table 1 compares the exhaustive and cascade approaches. To better understand our approach's scalability, we include extrapolated runtimes for 10 downstream tasks based on the fixed and per-task runtimes for wind turbine detection and solar farm segmentation.

Across the four backbone architectures and fixed recall values, our cascade approach consistently provides a speedup. On the 10-task extrapolation, C_{99} achieves a 3.17x speedup with the most accurate deep model (Swin-v2-Base) for solar farm segmentation, and a 1.67x speedup with the fastest model (ResNet50) for wind turbine detection. Similarly, C_{95} provides a 4.18x speedup with the most accurate model and a 2.07x speedup with the fastest model for the same tasks, respectively. Notably, we observe that: (1) even for relatively fast deep models, our approach still offers a speedup; (2) employing our cascade approach with the largest, most accurate models offers the greatest speedup with an extremely marginal drop in accuracy for the downstream task; and (3) larger speedups occur with an increase in the number of image patches or the number of tasks.

5 Conclusion

By utilizing pre-computed embeddings to exclude patches that do not contain relevant objects, our proposed cascade approach significantly accelerates global remote sensing inference, achieving a 5x

reduction in per-task runtime. Ultimately, our approach facilitates the deployment of deep models across a broader range of environmental monitoring applications while minimizing compute costs.

References

[1] Zayd Mahmoud Hamdi, Melanie Brandmeier, and Christoph Straub. Forest damage assessment using deep learning on high resolution remote sensing data. *Remote Sensing*, 11(17):1976, 2019.

[2] Pablo Pozzobon De Bem, Osmar Abílio de Carvalho Junior, Renato Fontes Guimarães, and Roberto Arnaldo Trancoso Gomes. Change detection of deforestation in the brazilian amazon using landsat data and convolutional neural networks. *Remote Sensing*, 12(6):901, 2020.

[3] Fernando Paolo, Tsu-ting Tim Lin, Ritwik Gupta, Bryce Goodman, Nirav Patel, Daniel Kuster, David Kroodsma, and Jared Dunnmon. xview3-sar: Detecting dark fishing activity using synthetic aperture radar imagery. *Advances in Neural Information Processing Systems*, 35:37604–37616, 2022.

[4] Patrick Beukema, Favyen Bastani, Piper Wolters, Henry Herzog, and Joseph George Ferdinando. Satellite imagery and ai: A new era in ocean conservation, from research to deployment and impact. In *NeurIPS 2023 Computational Sustainability: Promises and Pitfalls from Theory to Deployment*, 2023.

[5] Favyen Bastani, Piper Wolters, Ritwik Gupta, Joe Ferdinando, and Aniruddha Kembhavi. Satlaspretrain: A large-scale dataset for remote sensing image understanding, 2023.

[6] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021.

[7] Daniel Kang, John Emmons, Firas Abuzaid, Peter Bailis, and Matei Zaharia. Noscope: Optimizing neural network queries over video at scale, 2017.

[8] Daniel Kang, Peter Bailis, and Matei Zaharia. Blazeit: Optimizing declarative aggregation and limit queries for neural network-based video analytics, 2019.

[9] Oscar Mañas, Alexandre Lacoste, Xavier Giro i Nieto, David Vazquez, and Pau Rodriguez. Seasonal contrast: Unsupervised pre-training from uncurated remote sensing data, 2021.

[10] Utkarsh Mall, Bharath Hariharan, and Kavita Bala. Change-aware sampling and contrastive learning for satellite images. In *CVPR*, 2023.

[11] Vishal Batchu, Alex Wilson, Betty Peng, Carl Elkin, Umangi Jain, Christopher Van Arsdale, Ross Goroshin, and Varun Gulshan. Satellite sunroof: High-res digital surface models and roof segmentation for global solar mapping, 2024.

[12] Zixuan Dui, Yongjian Huang, Jiuping Jin, and Qianrong Gu. Automatic detection of photovoltaic facilities from Sentinel-2 observations by the enhanced U-Net method. *Journal of Applied Remote Sensing*, 17(1):014516, 2023.

A Appendix

A.1 Exploring MLP Performance Across Different Encoders

In this experiment, we evaluated the performance of the MLP architecture using embeddings generated by different backbone encoders. We selected the SeCo (ResNet 50), CACo (ResNet 50), SP-SiRGB (Swin-v2-base), and SP-MIMS (Swin-v2-base) models. The confidence threshold was adjusted to achieve a fixed recall of 99% on X_{train}, and we evaluated precision on X_{test}.

The observed differences in performance can be attributed to the following factors:

- **Contrastive Learning:** Representations obtained through contrastive learning may be more coarse-grained, potentially overlooking smaller features such as solar farms or platforms.
- **Training Data Quality:** The SatlasPretrain models were initially trained on noisy OpenStreetMap labels for various categories, including solar farms and platforms. In contrast, SeCo and CACo were predominantly trained on Sentinel-2 Imagery.
- **Temporal Pooling:** The SP-MIMS model's representation quality can also be attributed to the fact that it takes multiple images across different timestamps as input through the Swin backbone, and merges the representations by performing temporal pooling across the images.

Table 2: Precision metrics of MLPs across various encoders at 99% recall.

ENCODER	PRECISION
SECO (RESNET 50)	0.017
CACO (RESNET 50)	0.018
SP-SIRGB (SWIN-V2-BASE)	0.021
SP-MIMS (SWIN-V2-BASE)	**0.064**

A.2 Data Preprocessing

This section outlines the steps involved in pre-processing image patches before feature extraction using SP-MIMS.

Normalization. We begin by normalizing the image patches to standardize pixel values, which mitigates the variations caused by different lighting conditions or sensor discrepancies. This normalization scales the pixel values between 0 and 1. In certain cases, we clip pixel values within the range of 0 to 255 for specific spectral channels to standardize the channels and reduce noise.

Stacking bands. We process Sentinel-2 image patches, which have 13 spectral channels, according to the requirements of each encoder. For multi-spectral encoders, we stack all nine spectral channels. However, for models processing only RGB data, we only utilize the true color image (TCI) band. SP-MIMS processes nine of the 13 spectral channels across four images captured at different times using a Swin-v2-base backbone. Each patch yields four feature maps at different scales.

Encoder output. The output dimensionality, $x_{t,j}$, from the encoder E, is determined by the structure of the encoder. When using SP-MIMS, the four feature maps are processed with average pooling and flattening, and then concatenated to form a single 1920-dimensional embedding for each image patch. The MLP processes a 1920-dimensional input vector derived from the concatenated feature maps $x_{t,j}$.

Applying SP-MIMS to our image patches, we generate embeddings that significantly reduce the data size, approximately by 40 times, while preserving the bare minimum information for our cascade to effectively filter out significant portions of image patches for a wide variety of tasks involving sparse targets.

A.3 Extended Methodology: Supplementary Details

Confidence threshold tuning. h must be carefully tuned. If h is too high, then we will skip applying D on patches that do actually contain instances, thus impacting accuracy. If h is too low, then we will spend more time than needed applying D on patches that clearly contain zero instances; at the extreme, $h = 0$ is essentially the exhaustive approach of applying D on every patch.

Precision is the ratio of true positives to positive predictions (true positives plus false positives), and recall is the ratio of true positives to positive labels (true positives plus false negatives). Then, recall is equivalent to the accuracy reduction from false negative errors. Meanwhile, precision corresponds to the speedup conferred by the cascade method. For example, if instances appear in 1% of patches, and the MLP provides 1% precision, then we would need to apply D on every patch; if it instead provides 5% precision, the MLP will mark only 1/5 of all image patches for further processing by the deep model.

Table 3: Additional Performance metrics for different deep model backbones across various tasks. Training and inference processes are standardized on 4 NVIDIA A6000 GPUs.

	Wind Turbine				Solar Farm
Deep Model Backbone	ResNet50	ResNet152	Swinv2_tiny	Swinv2_base	Swinv2_base
Deep Model Metrics					
Global target occurrence (%)	1.33	1.36	1.26	1.11	1.24
Accuracy (F1 score)	77.73	81.16	80.65	81.28	90.65
Precision (%)	77.36	78.76	76.61	77.26	x
Recall (%)	78.1	83.71	85.13	85.74	x
Training time (Hours)	10.5942	16.7641	16.1576	19.53	12.37
Single patch inference time (seconds)	0.0045161	0.00625753333	0.0061716333	0.0113918	0.0133
Cascade Approach (C_{99})					
Precision (%)	4.28	4.54	5.49	5.76	5.99
Recall (%)	98.92	98.93	98.82	99.34	98.99
Filtered patches (%)	31.07	29.96	22.95	19.27	20.7
Cascade Approach (C_{95})					
Precision (%)	7.11	6.71	8.07	8.95	9.66
Recall (%)	94.95	95.00	98.82	94.9	94.77
Filtered patches (%)	18.71	20.27	15.61	12.4	12.84

Training and validation dataset sizes. Since positives are rare (per our sparse target assumption), with a randomly sampled training dataset, we find that the dataset size N must be large (up to 10K) to include sufficient positives to train an accurate MLP. Rather than solely use a random sample of global image patches, we extend X_{train} and X_{val} with examples from the underlying dataset X_D that was used to train the deep model D. For these examples, we set the label y based on the presence of ground truth labels in the X_D instead of based on the outputs of D. We split X_D evenly and randomly between X_{train} and X_{val}. If X_D is unavailable, we can completely rely on a random sampling of global image patches using labels generated by D.

Inference. During inference, the MLP evaluates each of the pre-computed embeddings, $x^t_{i,j}$, to estimate object presence for each image patch, $I^t_{i,j}$. We apply D on the subset of patches where $p^t_{i,j} \geq h$, and concatenate the predictions from D to produce the final output. By skipping applying the deep model on patches that the MLP is confident contain zero instances, our approach achieves a significant speedup compared to the exhaustive inference approach.

A.4 Facilitating Innovation in Remote Sensing Through Cost-Effective Analysis

Recent advancements in remote sensing have led to the development of specialized models tailored to address specific challenges. For instance, Beukema et al. (2023) developed models for near real-time vessel detection, which are deployed in the Skylight maritime monitoring platform. Similarly, Hamdi et al. (2019) utilized convolutional neural networks (CNNs) to achieve high-accuracy classification of forest damage from high-resolution imagery. Paolo et al. (2022) focused on detecting dark fishing activities using synthetic aperture radar (SAR) imagery.

While these specialized models provide high accuracy for their respective tasks, they often require significant computational resources and bespoke training, which can be costly and time-consuming. Our cascade framework complements these models by offering a cost-effective preliminary filtering step. This encourages the development and deployment of more specialized models, making it feasible to apply them on a large scale economically.

A.5 Evaluation Details

Table 3 contains metrics that underpin the accuracy and runtime evaluations presented in Table 1.

Training time. We trained four deep models on wind turbine detection with ResNet50, ResNet152, Swin-v2-Tiny, and Swin-v2-base backbones, plus one model on solar farm segmentation using a Swin-v2-base backbone. Training started with the heads—Faster R-CNN for detection and Mask R-CNN for segmentation—with backbone weights frozen for 25 epochs, then continued for another 25 epochs with unfrozen backbones. The deep model training times are documented in Table 3.

The average MLP training time of 71 seconds is incorporated into our cascade approach's runtime calculations.

Inference time. We calculate single patch inference time by averaging each model's runtime on a dataset X_R, which consists of 30,000 image patches randomly sampled from a global mosaic. The labels from X_R serve as ground truth for evaluating each corresponding MLP. Additionally, the averaged inference times of our MLPs are factored into our cascade approach's runtime calculations.

Global target occurrence. The fine-tuned models are assessed on a subset of X_D, with the confidence threshold optimized to maximize accuracy, precision, and recall. Global target occurrence percentages reflect the likelihood of a target (e.g., wind turbine or solar farm) appearing in a random global image patch. We estimate this occurrence by applying the fine-tuned models to 30,000 random image patches and calculating the percentage of positives—more than one bounding box indicates wind turbine detection, while non-empty segmentation masks denote solar farm identification.

Accuracy comparison. We trained two distinct cascades with confidence thresholds tuned to fixed recalls of 99% and 95% on X_{train}, evaluating them by the MLP's precision and recall on X_R. Unlike deep models that provide detailed target information, the MLP's role is solely to identify image patches containing targets like wind turbines. As such, precision and recall metrics for cascade filters differ from those of deep models. For simplicity, we estimated the accuracy of the cascade approach by multiplying the filter's recall by the exhaustive approach's accuracy (F1 score).

Filtered patch percentage. Larger precision values in the cascade filter leads to faster runtime due to fewer false positives requiring deep model analysis. In Table 3, the percentages of filtered image patches—comprising both true and false positives selected by the cascade filter—are reported, with lower values signifying effective filtering of empty patches by the cascade filter.