Dead Tree Segmentation from Aerial Images

YIQIN XIE
z5469338

YIROU LI
z5563505

YINGJI JIN
z5472432

JIAJUN LI
z5502131

LONGLONG LIU
z5514302

***Abstract*—This project investigates automatic dead tree segmentation from aerial imagery to enhance forest monitoring. We compare traditional image processing, machine learning methods (Random Forest, SVM, XGBoost), and deep learning models (U-Net, ECA-UNet, ResUNet). Experimental results show that U-Net achieves the best overall performance, and deep learning methods demonstrate significant advantages in small-object detection and complex backgrounds.**

***Index Terms*—Dead tree segmentation; Aerial imagery; Semantic segmentation; Random Forest; Ensemble voting; Deep learning; U-Net**

I. Introduction

Forests are vital ecosystems, supporting biodiversity and regulating climate and soil stability. Dead trees disrupt ecological balance and heighten wildfire risks, making their monitoring essential for environmental management and disaster prevention.

Developments in aerial and satellite remote sensing now enable large-scale forest observation, but the massive data volume renders manual analysis impractical, necessitating automated computer vision solutions.

This project uses the *Aerial Imagery for Dead Tree Segmentation* dataset from Kaggle, comprising 444 aerial images with segmentation masks. To segment standing dead trees from these images and identify the most effective method, several approaches were implemented. We used a traditional image segmentation method as a baseline, and further developed two machine learning methods and deep learning methods.

II. Literature Review

A. Traditional Computer Vision Methods

As a baseline, we employed a traditional image segmentation method to compare against machine learning and deep learning approaches. Supported by both statistical correlation analysis and prior work, we used the two features as input channels, performing segmentation based on pixel histograms and regional connectivity [1].

B. Machine Learning-based Methods

In recent years, traditional machine learning methods have continued to play an important role in remote sensing image analysis, particularly in scenarios with limited data or a strong demand for model interpretability. In this project, we designed two complementary strategies for pixel-level dead tree classification, tailored to the characteristics of different classifiers:

Full-feature input strategy: This approach leverages models that are tolerant of redundant features, such as Random Forest, allowing us to include a large number of input features and let the model implicitly select the most discriminative ones during training.

Multi-model ensemble strategy: This approach employs multiple classifiers (SVM and XGBoost) and fuses their predictions through a weighted soft voting mechanism. As training multiple models increases computational cost, we manually selected a subset of relevant features and applied resampling to reduce training time and mitigate overfitting.

For the first strategy, we employed the *Random Forest* model, which has an inherent ability to perform feature selection by favoring the most informative features during training(Belgiu and Drăguţ, 2016) [2]. Due to its ensemble structure and random feature subsetting at each tree node, Random Forest is highly robust to high-dimensional and redundant inputs, making it particularly well-suited for remote sensing tasks such as land cover classification and vegetation monitoring. These characteristics make it an ideal candidate for the full-feature strategy.

For the second strategy, we used *Support Vector Machine (SVM)* and *XGBoost* as base classifiers and integrated their outputs using a weighted soft voting scheme. These two models exhibit strong complementary behavior. SVM excels at constructing clear decision boundaries and performs well when class separability is high. In contrast, XGBoost is capable of capturing complex nonlinear relationships through gradient-boosted decision trees, and it handles noisy and redundant features more effectively, thanks to its built-in regularization and feature selection mechanisms (Shao et al., 2024) [3]. By combining their probabilistic outputs, the ensemble benefits from SVM's precision in well-defined regions and XGBoost's robustness in complex or ambiguous regions, resulting in improved segmentation performance, especially for small or weakly defined dead tree targets.

C. Deep Learning-based Methods

Deep learning, as one of the machine learning methods, has dominated in various application areas due to the ability to learn enormous volumes of data and automatically extract features. Compared to traditional machine learning methods, such as Random Forests and Support Vector Machines, deep learning delivers superior performance in identifying the structural characteristics of objects. By constructing multi-layer networks, deep learning enables computers to automatically learn complex relationships within data, extracting higher-dimensional and more abstract information, thus enhancing the representation capability of features [4].

In the field of remote sensing, deep learning is widely used for semantic segmentation in imagery. Recent research shows that U-Net has demonstrated strong performance in pixel-wise semantic segmentation tasks [5] [6].

U-Net is one of the earliest convolutional neural networks (CNNs) to propose an encoder-decoder architecture for semantic segmentation, originally designed for medical image segmentation [7]. It has a symmetrical "U"-shaped structure, consisting of a down-sampling encoder and an up-sampling decoder. The encoder extracts features and produces a prediction map through multiple convolution and pooling layers. The decoder then uses transposed convolutions to restore spatial resolution and generate the final segmentation output. Skip connections are used to transfer features from the encoder to the decoder, which effectively preserves spatial information and improves segmentation accuracy.

Xu et al. employed a single-tree semantic segmentation method based on an improved U-Net, which integrates the Efficient Channel Attention (ECA) module introduced by Wang et al. into the decoder [4] [8]. The module introduces only a few additional parameters, but it significantly enhances segmentation performance. As illustrated in Figure 4, after channel-wise global average pooling without dimensionality reduction, the ECA considers each channel and its k neighbouring channels to achieve efficient local cross-channel interaction. This approach has been proven to maintain a good balance between computational cost and performance.

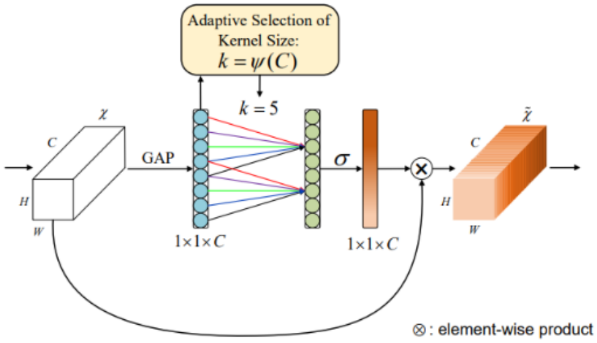

Fig. 1: ECA structure

ResUNet is a variant of the U-Net architecture that integrates residual learning into the encoder-decoder structure to enhance feature extraction and gradient propagation in deep networks. By incorporating residual blocks into the U-Net encoder, ResUNet effectively mitigates the vanishing gradient problem and improves feature reuse, which is crucial for segmenting small and sparse objects in complex environments [9].

In the context of remote sensing, ResUNet has demonstrated excellent performance in pixel-level object detection and land-cover classification. Diakogiannis et al. (2020) proposed ResUNet-a, an advanced variant specifically designed for remote sensing semantic segmentation, which integrates residual connections, atrous convolutions, pyramid scene parsing, and multi-task learning based on Dice loss, achieving state-of-the-art performance on the ISPRS datasets [10]. More recently, Xu et al. (2023) applied an improved ResUNet for UAV-based forest monitoring, demonstrating that the combination of residual connections and attention modules can effectively enhance the segmentation of sparse tree crowns in complex canopy environments [4].

III. Methods

A. Image segmentation

In this study, we utilized the **Normalized Difference Vegetation Index (NDVI)** and **image entropy (entropy_img)** as input channels, combining pixel histograms and regional connectivity to perform image segmentation. The selection of these features was supported by both correlation analysis and prior literature. Li and Chen (2021) proposed an entropy-based adaptive segmentation strategy that significantly improved boundary fitting in remote sensing imagery, validating the effectiveness of `entropy_img` [1].

We evaluated several candidate features—including R, G, B, NDVI, GNDVI, entropy, and NDVI statistics—and found that NDVI and `entropy_img` exhibited the highest correlation with the ground-truth mask based on both *mutual information* and *Pearson correlation*. These two features were thus chosen for threshold-based segmentation.

The segmentation process involved normalization, empirical thresholding, and morphological operations (e.g., opening and closing) to smooth edges and remove noise. While efficient, this method lacks contextual modeling and struggles with small or complex targets.

To identify the optimal threshold combination, we applied a grid search strategy that exhaustively evaluated pairs of entropy and NDVI thresholds, aiming to maximize the Intersection over Union (IoU) between the predicted mask and ground truth. The best-performing threshold pair was:

$$\textbf{Entropy} > \textbf{0.9 and NDVI} < \textbf{0}$$

B. Random Forest

Patch-level Classification In this method, a sliding window is applied to the NRG images to extract fixed-size patches. Each patch is labeled based on the class of its center pixel, which simplifies the labeling process and avoids ambiguity at patch edges.

Since the dataset is highly imbalanced, with far fewer positive samples than negative ones, a sampling strategy is used to retain all positive samples while randomly selecting a limited number of negative samples at a fixed ratio. This helps improve model performance on the minority class.

The pixel values and other features within each patch are flattened into one-dimensional vectors and used to train a Random Forest classifier. During testing, the same sliding window is applied to generate patches, and the model predicts the class of each patch's center pixel. As the prediction remains at the patch level, it does not account for spatial continuity or precise object boundaries.

Patch-based Classification with Reconstruction The trained Random Forest model from the previous patch-level classification approach is also used in this method. However, instead of evaluating only on sampled patch-level test data, this method maps predictions back to image coordinates to produce a sparse pixel-level output that aligns with the original image structure.

After training the Random Forest classifier, a sliding window is applied to the test images to extract patches. Each patch is classified independently, and the predicted label of its center pixel is mapped back to the corresponding location in the original image. This results in a sparse prediction map, where only the center pixels of the patches are labeled.

Although this method brings the output into the spatial domain and allows for better visual inspection, it does not produce a complete dense segmentation map. Pixels that are not covered by patch centers remain unlabeled, which still limits the ability to assess spatial continuity and object boundaries.

Pixel-level Feature-based Classification In this approach, classification is performed directly at the pixel level without using patches. For each pixel, a 17-dimensional feature vector is extracted, incorporating spectral values from RGB and NRG channels, vegetation index (NDVI), local mean and variance computed within a fixed-size window, and normalized spatial coordinates.

The resulting pixel-level features are used to train a Random Forest classifier with balanced class weighting to handle class imbalance. During inference, the same feature extraction procedure is applied to the test images, and each pixel is classified independently. A fixed probability threshold is then applied to the classifier output to generate binary predictions. Since all pixels are individually evaluated, this method yields a dense, full-resolution segmentation map. It also helps preserve spatial details and improves the recognition of small or fine structures.

To determine the optimal number of trees, the model was trained and evaluated using different tree counts ranging from 10 to 50. Although the accuracy improvements were relatively small, 50 trees consistently achieved slightly better performance across metrics such as IoU, precision, recall, and F1-score. Therefore, 50 was selected as the final tree count.

In addition, a series of experiments were conducted to evaluate the impact of varying the probability threshold from 0.5 to 0.95. It was observed that lower thresholds resulted in higher recall but lower precision, while higher thresholds did the opposite. A threshold of 0.748 was chosen as it provided the best trade-off between false positives and false negatives, reflected by improved F1-score and balanced performance across key metrics.

Method Comparison Although patch-based approaches offer better class balance during training, it is more often to use pixel-wise prediction maps in practical applications. However, when we convert patch-level classification results back to pixel-level outputs, it may lead to the loss of spatial boundary information, which can have an adverse effect on the localization accuracy of small objects [9].

In the pixel-based approach, features are extracted from each pixel and its local neighborhood, which can predict based on each pixel when training Random Forest model. Since there is no need to convert patch-level to pixel-level in this method, the loss of spatial boundary information can be reduced and this method may improve the recognition accuracy of small objects.

To better illustrate the differences between the three approaches, a comparison of their processing figure is shown in Fig. 2.

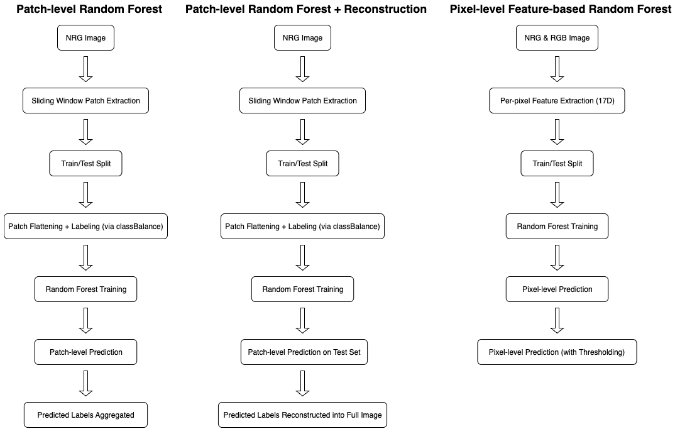
Patch-level Random Forest
NRG Image
Sliding Window Patch Extraction
Train/Test Split
Patch Flattening + Labeling (via classBalance)
Random Forest Training
Patch-level Prediction
Predicted Labels Aggregated
Patch-level Random Forest + Reconstruction
NRG Image
Sliding Window Patch Extraction
Train/Test Split
Patch Flattening + Labeling (via classBalance)
Random Forest Training
Patch-level Prediction on Test Set
Predicted Labels Reconstructed into Full Image
Pixel-level Feature-based Random Forest
NRG & RGB Image
Per-pixel Feature Extraction (17D)
Train/Test Split
Random Forest Training
Pixel-level Prediction
Pixel-level Prediction (with Thresholding)

Fig. 2: Comparison of the processes for (a) patch-level, (b) patch-based with reconstruction, and (c) pixel-level feature-based methods.

C. SVM and XGBoost with Weighted Soft Voting

We also explored a fusion strategy that integrates two complementary models—SVM and XGBoost—using a weighted soft voting mechanism. The objective was to combine their respective strengths to improve overall classification performance.

Data Preprocessing Dead trees typically affect regions rather than isolated pixels, and the images contain noise. To address this, we initially attempted a patch-based feature extraction strategy using a sliding window, aiming to capture contextual and neighborhood information. However, we found that when converting the pixel-wise mask into patches using hard thresholding, the label assignment became ambiguous due to edge smoothing. Even under optimal configurations (threshold, window size, stride), the IoU between patch-generated labels and the original mask was only around 0.35, indicating about 65% information loss. As a result, we abandoned the sliding window approach.

Instead, we adopted pixel-level neighborhood feature extraction using functions such as `generic_filter()`, `sobel()`, and `entropy()`. However, this approach produced a very large number of samples—over 5.8 million—which significantly increased training time and introduced overfitting risks.

To address this, we implemented a custom `resample()` function to reduce the dataset size while maintaining class balance. This function allowed us to control the maximum number of samples per class and ensured a manageable and balanced training set.

Feature Exploration We extracted 11 candidate features, including RGB bands, NDVI, GNDVI, local NDVI variance, Sobel magnitude and angle, local entropy, saturation, perceived luminance, and mean NDVI. Based on mutual information and Pearson correlation scores (see Table I and Table II), local entropy, NDVI, and mean NDVI were identified as the most relevant to the segmentation labels.

Feature	MI with Label	Pearson Correlation
ENTROPY_img	0.0452	0.0757
NDVI	0.0145	0.0213
NDVI_mean	0.0094	-0.0993
R	0.0063	—
SAT	0.0043	-0.0145
L	0.0042	-0.0034
G	0.0036	0.0091
NDVI_var	0.0009	0.0473
NDVI_sobel	0.0007	-0.0389
GNDVI	0.0001	0.0377
SOBEL_angle	0.0000	-0.1134

TABLE I: Mutual Information between features and labels

Accordingly, we selected different feature subsets for SVM and XGBoost.

SVM Model Support Vector Machine (SVM) is a discriminative classifier that maximizes the margin between classes. It is suitable for complex background segmentation tasks due to its ability to construct non-linear boundaries using kernel functions. However, according to Faska, Moulay and Bouhmadi(2023), SVM sensitive to noise and not ideal for large-scale datasets [11].

Considering these characteristics, we selected the following five features for SVM: NDVI, NDVI variance (3×3 window), local entropy, saturation, and mean NDVI (5×5 window). We resampled the training set to include 5,000 positive and 2,000 negative samples. A radial basis function (RBF) kernel was used, with hyperparameters tuned via cross-validation: $C = 10$ and $\gamma = 0.01$.

XGBoost Model XGBoost is an ensemble learning method based on gradient boosting decision trees. Faska, Moulay and Bouhmadi(2023) proposed, instead of trying to find a single efficient and optimal learner, ensemble-based techniques take the advantage of each basic model; they integrate their outputs to obtain a more consistent and reliable learner [11]. It iteratively improves classification by correcting the residuals of previous trees. Its strength lies in handling high-dimensional and non-linear feature spaces, with built-in mechanisms for automatic feature selection.

For XGBoost, we retained all five features used in SVM and added additional features, including NIR reflectance (healthy trees), Sobel magnitude, and perceived luminance, resulting in a richer feature set. We used 10,000 samples from both the positive and negative classes.

Weighted Soft Voting After training both models, we performed weighted soft voting on their predicted probabilities as follows:

$$\text{avg\_proba} = \alpha \cdot p_{\text{SVM}} + (1 - \alpha) \cdot p_{\text{XGB}} \tag{1}$$

Ensemble-based models like XGBoost are more effective for complex remote sensing classification tasks where feature interaction is important(Chen and Guestrin, 2016) [12]. We observed that XGBoost consistently outperformed SVM in terms of IoU on the validation set. Hence, we assigned more weight to XGBoost by setting $\alpha = 0.05$. We then scanned over threshold values to identify the optimal classification boundary, with the best performance achieved at a threshold of 0.863. The final prediction rule is as follows:

$$\text{avg\_proba} = 0.05 \cdot p_{\text{SVM}} + 0.95 \cdot p_{\text{XGB}} \tag{2}$$

$$\text{best\_preds} = \nVdash (\text{avg\_proba} > 0.863) \tag{3}$$

This weighted ensemble strategy effectively combined the complementary characteristics of the two models, resulting in improved IoU and robustness compared to either individual model.

D. Deep learning

Dataset A dataset class `DeadTreeDataset` is implemented to preprocess and load data for semantic segmentation. The dataset takes RGB images, NIR grayscale images, and corresponding segmentation masks as input. The RGB and NIR images are concatenated to form a 4-channel input tensor, and the mask is binarized to represent dead tree presence (1) or absence (0).

Data augmentation We apply random data augmentation, including: Horizontal flip (p=0.5), Vertical flip (p=0.5), Random rotation within ±15°. All images are first scaled to $[0, 1]$ and then normalized using mean = [0.485, 0.456, 0.406, 0.5] and std = [0.229, 0.224, 0.225, 0.5]. Each returned sample is a tuple `(image, mask)`, where:

`image` is a 4-channel `float32` tensor of shape $[4, 128, 128]$ normalized to approximately $[-2, 2]$,

`mask` is a binary `float32` tensor of shape $[1, 128, 128]$ with values $0, 1$. Samples with empty masks (all zeros) are skipped to ensure the presence of positive samples in training.

U-Net Based on the U-Net architecture shown in the fig 3, we divided it into three main parts: the encoder module, the decoder module, and the output module.

Encoder Module: Consists of four down-sampling blocks, each containing two convolutional layers followed by a max pooling layer.

Decoder Module: Contains four relevant up-sampling blocks. Each block has two convolutions followed by a transposed convolution. After each up-sampling, the result is concatenated with the corresponding encoder feature.

Output Module: Applies three convolutional layers to gradually reduce the number of channels, and uses a Sigmoid activation function to produce a single-channel probability map.

Feature	ENTROPY	NDVI	NDVI_mean	R	SAT	L	G	NDVI_var	NDVI_sobel	GNDVI	SOBEL_angle
ENTROPY_img	1.00	0.95	-0.22	–	-0.33	-0.48	-0.011	0.066	-0.68	0.96	-0.23
NDVI		1.00	-0.0079	–	-0.37	-0.53	-0.012	0.055	-0.61	1.00	0.0062
NDVI_mean			1.00	–	-0.17	-0.15	-0.00059	0.027	0.14	-0.055	0.85
R				–	–	–	–	–	–	–	–
SAT					1.00	0.85	-0.0029	-0.074	0.44	-0.37	-0.17
L						1.00	-0.00006	0.022	0.51	-0.54	-0.18
G							1.00	-0.0045	0.0026	-0.012	-0.00065
NDVI_var								1.00	-0.14	0.068	0.0091
NDVI_sobel									1.00	-0.64	0.17
GNDVI										1.00	-0.042
SOBEL_angle											1.00

TABLE II: Pairwise Pearson Correlation Between Features

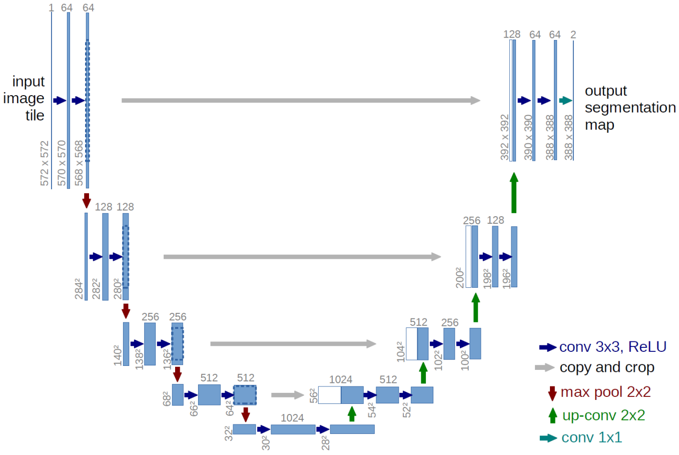

Fig. 3: U-net network model structure

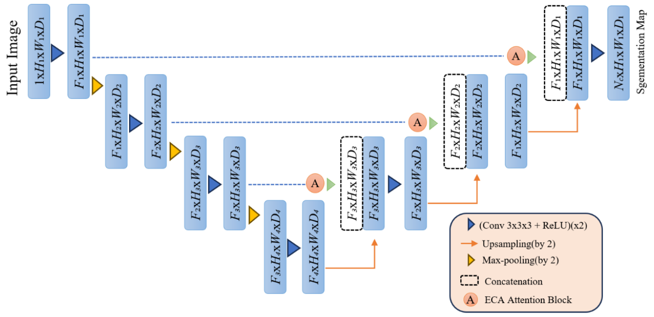
Input Image
$1\text{x}H_1\text{x}W_1\text{x}D_1$
$F_1\text{x}H_1\text{x}W_1\text{x}D_1$
$F_1\text{x}H_2\text{x}W_2\text{x}D_2$
$F_2\text{x}H_2\text{x}W_2\text{x}D_2$
$F_2\text{x}H_3\text{x}W_3\text{x}D_3$
$F_3\text{x}H_3\text{x}W_3\text{x}D_3$
$F_3\text{x}H_4\text{x}W_4\text{x}D_4$
$F_4\text{x}H_4\text{x}W_4\text{x}D_4$
A
$F_3\text{x}H_3\text{x}W_3\text{x}D_3$
$F_3\text{x}H_3\text{x}W_3\text{x}D_3$
$F_2\text{x}H_3\text{x}W_3\text{x}D_3$
A
$F_2\text{x}H_2\text{x}W_2\text{x}D_2$
$F_2\text{x}H_2\text{x}W_2\text{x}D_2$
$F_1\text{x}H_2\text{x}W_2\text{x}D_2$
A
$F_1\text{x}H_1\text{x}W_1\text{x}D_1$
$F_1\text{x}H_1\text{x}W_1\text{x}D_1$
$N_c\text{x}H_1\text{x}W_1\text{x}D_1$
Sgementation Map
(Conv 3x3x3 + ReLU)(x2)
Upsampling(by 2)
Max-pooling(by 2)
Concatenation
ECA Attention Block

Fig. 4: ECA-Unet model structure

ECA-UNet As shown in the fig 4, compared to the U-Net structure, ECA-UNet introduces an ECA attention block, which is inserted after the skip connection and before the subsequent convolution operation, enhancing the interactions between feature channels. This block makes the model focus more on important channel information, thereby improving segmentation performance.

ResUNet The ResUNet model is a variant of the U-Net architecture that integrates a ResNet50 backbone as the encoder. In this work, we adapted the first convolutional layer of ResNet50 to accept 4-channel input (RGB + NIR), allowing the network to exploit both visible and near-infrared information for improved tree crown segmentation [9].

The ResUNet consists of three main components:

Encoder Module (ResNet50 Backbone): Utilizes the convolutional layers of ResNet50 pretrained on ImageNet for feature extraction. Residual connections in the encoder help alleviate the vanishing gradient problem and enhance feature reuse.

Decoder Module: Similar to standard U-Net, the decoder progressively upsamples the feature maps using transposed convolutions. Skip connections are used to concatenate encoder features with decoder features at each stage, preserving spatial details and improving small-object segmentation performance.

Output Module: A final convolutional layer maps the decoder output to a single-channel feature map, followed by a Sigmoid activation to generate a probability mask representing dead tree presence.

The incorporation of the ResNet50 backbone significantly enhances the model's ability to capture high-level semantic features from complex forest backgrounds while maintaining the spatial precision necessary for small dead tree crown detection.

IV. Experimental Results

We conducted our experiments on the "Aerial Imagery for Dead Tree Segmentation" dataset, which contains 444 samples. Each data sample includes a RGB image, NIR grayscale images and a corresponding manually annotated segmentation mask images.

To better compare the performance of the models in dead tree segmentation, all images of different sizes were resized to a uniform resolution of 128×128.

The dataset was randomly split into 80% for training and 20% for validation.

The results of the traditional segmentation method are used as the baseline, against which the performance gains of other models are compared.

For the Random Forest classifier, we selected 17 input features and trained the model with the following hyperparameters: 50 estimators, a maximum tree depth of 15, and `class_weight='balanced'` to address class imbalance. The prediction threshold was set to 0.748.

For the ensemble model, we combined a Support Vector Machine (SVM) and an XGBoost classifier using a weighted soft voting approach:

- **SVM**: Trained with an RBF kernel using penalty parameter $C = 10$, kernel width $\gamma = 0.01$, and `class_weight='balanced'`.
- **XGBoost**: Trained with 1000 trees, learning rate of 0.05, maximum depth of 3, and regularization parameters $\alpha = 5$, $\lambda = 10$, and $\gamma = 1$.

The final prediction probability was computed as a weighted average: $0.12 \cdot \text{SVM} + 0.88 \cdot \text{XGBoost}$, and binarized using a threshold of 0.863.

For the Deep Learning Model, we used the Adam optimizer with an initial learning rate of 1e-4. The model was trained for 80 epochs and a batch size of 8 with an early stopping mechanism to prevent overfitting. During training, we experiment with three different loss functions(BCELoss, BCEWithLogitsLoss and a combination of BCELoss and IoULoss), and in the comparative analysis presented below, we report the results based on the best-performing loss function. That is: Unet and ECA-Unet with Combo Loss, and ResUnet with BCEWithLogitsLoss.

To evaluate model performance, we used the following metrics:

IoU (Intersection over Union): Measures the overlap between the predicted segmentation region and the ground truth region.

Accuracy: Measures the overall proportion of correct predictions.

Precision: Measures how many of the pixels predicted as dead trees are actually dead tree pixels.

Recall: Measures how many of the actual dead tree pixels are correctly identified by the model.

F1-score: The harmonic mean of precision and recall, reflecting a balance between accuracy and coverage.

The results of all methods are presented in Table III, which summarizes the performance of the baseline traditional segmentation method, machine learning models, and deep learning architectures.

Model	Accuracy	Precision	Recall	F1 Score	IoU
Image Segmentation	0.8085	0.0547	0.5610	0.0996	0.0524
Voting Model	0.9668	0.2325	0.3181	0.2686	0.1552
Random Forest	0.9635	0.2494	0.4496	0.3208	0.1911
UNet	0.9842	0.5903	0.6012	0.5957	0.4242
ECA-UNet	0.9856	0.7022	0.4368	0.5386	0.3685
Res-UNet	0.9799	0.4829	0.5789	0.5265	0.3573

TABLE III: Performance comparison across all models

V. Discussion

As shown in table III, the baseline traditional segmentation method (Image Segmentation) yields relatively low performance, with an IoU of only 0.0524 and an F1-score of 0.0996. This indicates that for the challenging task of dead tree segmentation, simple thresholding or traditional image processing methods are insufficient for producing accurate results.

Compared to the baseline, the introduction of machine learning methods leads to substantial performance improvement. Random Forest achieves an IoU of 0.1911 and an F1-score of 0.3208; the Voting Model using SVM and XGBoost achieves an IoU of 0.1522 and an F1-score of 0.2686. This improvement demonstrates that machine learning models can better utilize extracted features for classification, thereby enhancing segmentation performance.

However, despite these gains, there remains a significant gap when compared to deep learning models. Deep Learning Models clearly outperform machine learning models across all metrics. UNet achieves the highest IoU 0.4242 and F1-score 0.5957, representing the best overall performance. Res-UNet records slightly lower IoU 0.3573 and F1-score 0.5265 compared to UNet, while ECA-UNet achieves the highest precision 0.7022 but does not surpass UNet in IoU.

The performance of the traditional UNet model is better than that of ECA-UNet and Res-Unet. One possible explanation for this result is that the efficient channel attention mechanism places greater emphasis on weighting between channels, thereby increasing the model's focus on important feature channels. This mechanism effectively suppresses the influence of irrelevant or noisy channels, resulting in a significant improvement in the precision 0.7022, which indicates that the model becomes more cautious and confident when predicting the "dead tree" class, reducing the number of false positives. However, this improvement of precision often comes at the expense of recall, meaning that more true "dead tree" pixels are misclassified as background. Consequently, the overall IoU does not improve and may even decrease.

As for Res-UNet, it introduces residual blocks to alleviate gradient vanishing issues and enhance feature representation ability. However, given the limited dataset size in this project and the highly imbalanced and visually similar pixel distribution of dead tree targets, the additional network depth and parameters introduced by Res-UNet may not have been fully trained to realize their advantages. As a result, its performance is slightly lower than UNet.

In contrast, the basic UNet architecture achieves an effective balance between multi-scale feature extraction and spatial detail preservation through its symmetric encoder–decoder structure with skip connections. In this dataset, such a structure retains spatial location information while capturing the overall shape of target regions, which explains its superior recall 0.6012 and IoU 0.4242 performance.

As shown in the fig 5, we present a representative example to visually compare the segmentation results of the three deep learning models. Each row corresponds to one model (UNet, ECAUnet, and ResUNet), and the three columns respectively show the input RGB image, the ground truth mask, and the predicted mask generated by the model.

From the figure, it can be observed that UNet produces a relatively complete coverage of most dead tree clusters, with strong spatial consistency compared to the ground truth. The segmented regions are coherent, and the boundaries are relatively clean, with only minor omissions or boundary gaps

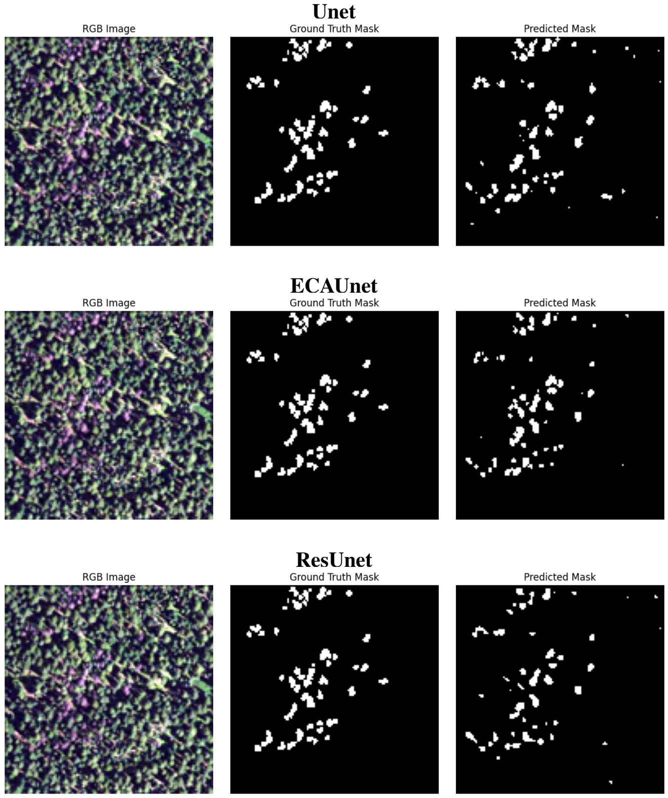

Fig. 5: Segmentation example

in low-contrast or background-like areas. This aligns with its highest IoU and relatively high F1-score in the quantitative results, indicating that it achieves a good balance between preserving spatial details and capturing global context.

ECA-UNet produces more conservative predictions. Its predction had fewer noise points and fewer false positives. However, it also shows noticeable omissions for small targets or low-contrast areas, with several clusters in the ground truth being only partially covered or entirely missing. This matches its high precision and low recall, suggesting that the channel attention mechanism focuses more on high-confidence feature channels, reducing false positives but also suppressing true positives with weak boundaries or less salient textures.

Res-UNet exhibits performance between the two models. Compared to ECA-Unet, it captures more targets, but its predicted regions sometimes contain holes or appear fragmented. Occasional false positives appear in background regions with bright spots or abrupt texture changes. These observations are consistent with its reasonable recall but lower IoU and F1-score than UNet, which indicating that under the current dataset scale and class imbalance, the representational gains from residual blocks have not fully translated into improved shape consistency.

Overall, the common challenges for all three models include missed detections of small and sparse dead tree clusters, unstable separation when tree crowns are close to each other, difficulty identifying targets in low-contrast or shadowed regions, and boundary discontinuities. These issues are closely related to the class imbalance and varying target scales in the dataset, and they suggest that pixel-level supervision alone may not be sufficient to fully constrain object shape and connectivity.

VI. Conclusion

In this project, we implemented and compared multiple dead tree segmentation methods, including traditional image segmentation, machine learning approaches (Random Forest, and an ensemble of SVM and XGBoost), and deep learning architectures (U-Net, ECA-UNet, and ResUNet). Experimental results show that deep learning methods significantly outperform machine learning methods for this task, with the basic U-Net architecture achieving the best overall performance in terms of IoU and F1-score.

Although its two variants, ECA-UNet and ResUNet, did not surpass U-Net on this dataset, each still has strengths: ECA-UNet demonstrated notable improvements in reducing false positives and achieving higher precision, while ResUNet showed potential in feature extraction and gradient propagation. Moreover, the U-Net architecture is highly extensible and can be enhanced with various mechanisms to further improve segmentation performance, which we aim to explore in future work.

However, this work still has certain limitations. The dataset size is relatively small and exhibits severe class imbalance, leading to unstable segmentation results for small targets, low-contrast regions, or occluded areas. Additionally, only single-scale pixel-level supervision was applied, which provides insufficient constraints for object shape and boundary consistency.

Future work will focus on expanding the dataset with higher-resolution and multi-spectral imagery, integrating multi-scale feature fusion and imbalance mitigation strategies, and optimizing U-Net variants with lightweight designs and transfer learning for scalable forest monitoring.

References

[1] Li, X. L., & Chen, J. S. (2021). Region adaptive adjustment strategy based on information entropy for remote sensing image segmentation. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, V-4-2021, 69–74. https://doi.org/10.5194/isprs-annals-V-4-2021-69-2021

[2] Belgiu, M., & Drăguţ, L. (2016). Random forest in remote sensing: A review of applications and future directions. *ISPRS Journal of Photogrammetry and Remote Sensing*, 114, 24–31. https://doi.org/10.1016/j.isprsjprs.2016.01.011

[3] Shao, Y., Li, Y., Jin, Y., Li, J., & Liu, L. (2024). Comparison of Random Forest and XGBoost Classifiers Using Integrated Optical and SAR Features for Mapping Urban Impervious Surface. *Remote Sensing*, 16(4), 665. https://www.mdpi.com/2072-4292/16/4/665

[4] Xu, S., Yang, B., Wang, R., Yang, D., Li, J., & Wei, J. (2025). Single Tree Semantic Segmentation from UAV Images Based on Improved U-Net Network. Drones, 9(4), 237. https://doi.org/10.3390/drones904023

[5] Osco, L.P.; Nogueira, K.; Marques Ramos, A.P.; Silva, F.S.; Costa, A.L.; Souza, G.T.; Rodrigues, T.A.; Alves, M.C.; Pereira, R.F.; Oliveira, R.S. Semantic segmentation of citrus-orchard using deep neural networks and multispectral UAV-based imagery. *Precision Agriculture*, **2021**, 22, 1171–1188.

[6] Ferro, M.V.; Sørensen, C.G.; Catania, P. Comparison of different computer vision methods for vineyard canopy detection using UAV multispectral images. *Computers and Electronics in Agriculture*, **2024**, 225, 109277.

[7] Ronneberger, O., Fischer, P., & Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In Navab, N., Hornegger, J., Wells, W., & Frangi, A. (Eds.), *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, Springer, Cham, 2015, pp. 234–241.

[8] Wang, Q., Wu, B., Zhu, P., Zhang, X., Li, J., Liu, Y., Li, Z., Gao, L., Yu, Z., & Yang, Q. ECA-Net: Efficient Channel Attention for Deep Convolutional Neural Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA, USA, 13–19 June 2020; pp. 11534–11542.

[9] Kampffmeyer, M., Salberg, A.-B., & Jenssen, R. Semantic Segmentation of Small Objects and Modeling of Uncertainty in Urban Remote Sensing Images Using Deep Convolutional Neural Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 1–9.

[10] F. I. Diakogiannis, F. Waldner, P. Caccetta, and C. Wu, “ResUNet-a: A deep learning framework for semantic segmentation of remotely sensed data,” *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 162, pp. 94–114, 2020.

[11] I. Faska, L. Moulay, and K. El Bouhmadi, “A robust and consistent stack generalized ensemble-learning framework for image segmentation,” *Journal of Electrical Systems and Information Technology*, vol. 10, no. 1, pp. 1–14, 2023.

[12] Y. Chen et al., “Comparison of Random Forest and XGBoost Classifiers Using Integrated Optical and SAR Features for Mapping Urban Impervious Surface,” *Remote Sensing*, vol. 16, no. 4, p. 665, 2024.