Vol. 08, No. 2 (2026) 667-678, doi: 10.24874/PES08.02.006

Proceedings on Engineering Sciences

www.pesjournal.net

AN EFFICIENT AND ACCURATE GHOST-OPTIMIZED LIGHTWEIGHT U-NET FOR SATELLITE IMAGE SEGMENTATION

Deepak Gupta [1]
Om Prakash Singh
Satyasundara Mahapatra

Received 05.01.2026.
Revised 08.02.2026.
Accepted 25.04.2026

Keywords:

Ghost Convolution, U-Net; Satellite Image Segmentation, Lightweight CNN

A B S T R A C T

Identifying roads, vegetation, land, and water resources is crucial for the sustainable growth and improvement of remote areas globally. For earth monitoring, computing semantic segmentation within the limits of constrained computing conditions is both an opportunity and a challenge. This study addresses the challenge of maintaining the quality of semantically segmented images by introducing an enhanced lightweight architecture. The proposed model provides a low computational cost solution for generating automatic area segmentation for the MBRSC dataset. To achieve optimal performance, an extensive comparison of patch sizes is made to choose patches of size 256x256 from the various tiles during image preprocessing. The proposed lightweight U-Net model enhances the mathematical and structural complexity of the encoder as well as the decoder by incorporating ghost convolutions. The model's performance is assessed using a custom loss function that uses focal, dice, and tversky loss. Ghost convolution-based U-Net is proved the best option as it demonstrated the ideal efficiency-accuracy trade-off due to outstanding computational efficiency of 0.38 GFLOPs, 1.88 MB size and maintains high segmentation quality with 87.3% test accuracy, 58.4% MIoU. The proposed architecture has achieved better efficiency ratios of 153.68 MIoU/GFLOP and 31.06 MIoU/MB.

© 2026 Published by Faculty of Engineering

1. INTRODUCTION

The term "remote sensing image" refers to any picture taken from a distance using equipment or sensors attached to vehicles like planes, helicopters, and satellites. Images like this allow us to learn more about things like the earth's surface and atmosphere without having to get our hands dirty (Toth & Jóźków, 2016). Aerial and satellite photographs are the two most common forms of remote sensing data. Aerial and satellite images both provide useful information, but they gather data in different ways and have different qualities. According to Spasev et al. (2023) at regular intervals, satellites circle the earth, gathering data from vast regions. Viewing the whole world in this way is very conceivable. Drones or aircraft flying lower to the ground are used to get aerial photos. Their smaller coverage area and higher level of detail make them perfect for researching localized regions. The amount of detail needed and the availability of data are two factors that determine whether aerial or satellite imagery is better suited for a given application. Semantic segmentation, or

[1] Corresponding author: Deepak Gupta
Email: deepakgupta.cse6@gmail.com

assigning a distinct annotation to each pixel in an image, has emerged as a prominent research topic for remote sensing imaging. Using this method, one may examine the location and connections of items on the ground in great detail. As reported by Neupane et al. (2021), segmentation allows for a more thorough comprehension of the picture's content via fine grained pixel-level identification, as opposed to object detection's emphasis on detecting and generally locating objects. From urban planning, (Boguszewski et al., 2022), to disaster management (Rashkovetsky et al., 2021, environmental monitoring and precision agriculture (Ulku, 2024), semantic segmentation in remote sensing affects a wide range of fields.

The field of semantic segmentation underwent a radical shift with the advent of deep learning, particularly fully convolutional networks FCNs and CNNs. This was achieved by allowing the automated acquisition of hierarchical representations from data, reported by Seth & Sharma (2022). FCN-based methods combined with encoder-decoder architectures have emerged as the prevailing paradigm for semantic segmentation. For precise prediction, those initial methods used spatial pooling after a series of sequential convolutions, (Pastorino et al., 2022). Subsequent approaches, including Unet (Ronneberger et al., 2015) and SegNet (Badrinarayanan et al., 2017), integrated low- and high-level feature maps during decoding by means of the upsampling technique. The goal was to properly delineate object bounds and record the image's overall context. Models like PSPNet (Zhao et al., 2017) and UperNet (Xiao et al., 2018) used spatial pyramid pooling to enhance efficiency by capturing contextual data at several scales in upper layers. Later advancements, shown by PSANet (Zhao et al., 2018) and DRANet (Fu et al., 2020), substituted conventional pooling with attention methods on top of encoder feature maps to more effectively capture long-range relationships.

Modern networks often use deeper network topologies or multilayer decoders to provide more precise results for significant prediction. These methods will unavoidably result in a rise in network complexity, thereby decreasing the network's speed. Lightweight deep learning (DL) refers to the technique of compressing DNN models in order to optimize their execution on devices that have limited computing capabilities. This is accomplished while keeping the compressed models' performance at parity with the originals. To address the aforementioned problems, a lightweight U-net model is suggested in the hopes of reducing computing costs while yet ensuring adequate detection performance. It improves upon the classic U-net design by adding ghost convolution to lessen the total processing burden by cutting down on parameter computations. Furthermore, the decreased network complexity and parameter count of proposed model makes it more practical in real-world scenarios compared to other benchmark networks.

2. LITERATURE REVIEW

Convolutional neural networks (CNNs) have shown their effectiveness in several computer vision applications, including segmentation and classification (Zhao et al., 2018). The CNN approach automatically extracts the hierarchical properties of several objects in an image. The network has the ability to collect intricate details in the shallow levels and complex characteristics in the deeper layers. A fully convolutional network (FCN) (Long et al., 2015), is also presented for image segmentation, which does dense prediction by assigning a label to each pixel of the view. Due to its significant improvement in segmentation accuracy, the FCN has now become the standard framework for semantic labeling tasks. FCN has effectively been used in the analysis of remote sensing data for tasks such as LULC, urban object recognition, and detecting changes. The U-Net architecture is another deep learning method used for semantic segmentation. During feature extraction, U-Net's contraction-expansion design and skip connections retains spatial information and minimize spatial resolution loss. U-Net is utilized for remote sensing applications such as road segmentation, (Zhang & Chi, 2020), environment protection (Avinash et al., 2025), crop monitoring, (Liu & Shi, 2020), Natural resource monitoring (Fan et al., 2022), oil spill detection and forest mapping (Sylvain et al., 2019). Numerous studies have been conducted by researchers with the aim of enhancing the accuracy of segmentation in remote sensing photographs but light weight and efficient semantic segmentation is still an uphill battle for resource constrained environments.

3. MAETRIALS AND METHODS

3.1 Dataset Preparation

An open access dataset, published by Humans in the Loop, annotated for a joint initiative with MBRSC in Dubai, is utilized, (Guo et al., 2021). The dataset is composed of aerial photos of Dubai landscape collected using MBRSC satellite (Humans in the Loop, 2024). The MBRSC satellite imaging is useful for urban planning, environmental surveillance, and infrastructure construction. Buildings, land, roads, vegetation, water, and unlabeled are the six distinct groups into which all of the taken photos are categorized using pixel-wise semantic segmentation. The unlabeled class handles ambiguous areas. Images are divided into eight bigger tiles. Each tile is further into 9 smaller images of varying sizes for easy dataset management. This data sheds light on Dubai's ecosystem, and the marking of structures, roadways, and flora makes in-depth research conceivable. The figure 1 shows both the original picture and its corresponding mask.

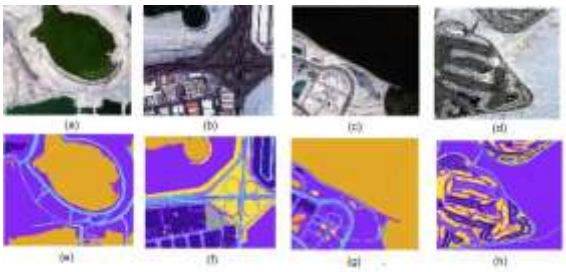


Figure 1. (a)-(d) Original Input Image, (e)-(h) corresponding masks

3.2 Preprocessing

Data preprocessing is used to improve the quality and compatibility of data before using it for analysis. Firstly, the varying size images are resized using cropping such that uniform patches may be generated. Image cropping is the process of removing a portion of an image to draw attention on a particular area or to resize the image to a desired size. While patching, the original high-resolution images are loaded and then divided into smaller, easier to handle components known as patches. Nonoverlapping patch generation is supported to preserve patch uniqueness. When dealing with the segmentation of a satellite image, the choice of an optimal patch size as input contributes to achieving the balance between computational efficiency and performance. Comparing U-net with ResNet18 and ResNet34 at different input patch sizes makes it clear that even though the parameter count and model size are kept fixed, the computational complexity (FLOPs) highly depends on input resolution, as shown in table 1. Smaller patch sizes like 128x128 lead to a minimum computational load with only 4.38G FLOPs needed by ResNet18 and 6.8 GFLOPs by ResNet34, but due to the small spatial contexts, the model cannot utilize notions of large-scale semantic structures and spatial dependencies, which are of pivotal significance to accurately segment high-resolution satellite images. In contrast, applying 256x256 input patches makes the computations expensive about four folds, but would enhance contextual information learning capacity of the model effectively without overloading system in terms of memory and processing demands. In comparison, 512x512 and 1024x1024 versions, albeit better in terms of the context capture, are 16 and 64 times more costly in terms of their computational requirements. Thus, 256x256 can be compelling compromise specially for semantic segmentation tasks in resource constrained environments.

Deep learning models need images with a uniform range of pixel values for processing. During normalization, minmax scaler maintains the spatial connections between pixels by evenly scaling the values from 0 to 1, which is crucial for tasks such as semantic segmentation. The original masks are in RGB format, whereas the object information given in the JSON file used the hexadecimal color codes.

Table 1. Comparison of Computational Complexity and Model Size at Different Input Resolutions

Patch Size	ResNet34		ResNet18		
	Size (MB)	FLOPs (G)	Size (MB)	FLOPs (G)	Rel. Cost
128×128	96.18	6.80	57.63	4.38	1×
256×256	**96.18**	**27.20**	**57.63**	**17.54**	**~4×**
512×512	96.18	108.82	57.63	70.16	~16×
1024×1024	96.18	435.27	57.63	280.65	~64×

The hex color codes are transformed into RGB labels, shown in figure 2, which are then turned into integer values to indicate distinct classes. Semantic segmentation, as it pertains to remote sensing imagery, entails labeling each pixel into relevant category like vegetation, land, or water etc. These categories are commonly represented by categorical values (e.g., 0, 1, 2 etc.) and do not possess any intrinsic order. One-hot encoding is a technique that ensures the model does not make any ordinal assumption about the classes. A binary matrix is created from the category labels; a separate vector represents each class. Preprocessing steps like dividing the pictures into parts, converting colors, and encoding labels, prepares data for deep learning models. This technique also increases computational efficiency and boosts the model's capacity to discover significant patterns. This eventually leads to improved performance in the semantic segmentation task.

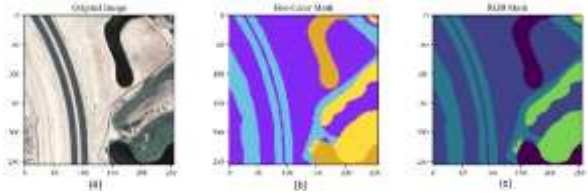


Figure 2. (a)Original Input Image (b) Mask in Hex Color (c) RGB mask

3.3 Methodology

consisting of aerial photography from Dubai. The dataset was divided into eight bigger tiles, resulting in a total of 72 photos. The pixel-wise semantic segmentation of each picture was tagged into six classes: buildings, land, roads, vegetation, water, and unlabeled. During pre-processing patching, min-max normalization and class encoding were applied. A total of 1,305 photos became available after the preprocessing using cropping as rescaling operation. The dataset was split into 1044 training, 130 validation, and 131 testing images. With great accuracy score, the trained model performed well on the training sets. Then, performance was further validated on test set of 131 unseen images. The entire workflow id depicted in figure 3. Various measures of model performance includes accuracy, IoU, tversky loss, dice loss, focal loss and an integrated bespoke loss function. Constant monitoring of these indicators is critical for refining training parameters and tackle challenges like over-fitting or under-fitting, promising optimum performance of the proposed model in semantic segmentation tasks. It

effectively segmented various terrain characteristics with precision, showcasing its ability to precisely identify and separate diverse elements. Evaluating the anticipated segmentation mask on an unexplored collection of images further demonstrated the model's ability to identify intricate spatial patterns in aerial images of Dubai.

3.4 Model Architecture

The U-Net is a specialized architecture created exclusively for semantic segmentation of biomedical images. Two main channels make up the distinctive U-shaped structure of U-Net architecture. Data about the input images' context is retrieved by the contracting path (encoder). The topology of this neural network adheres to the standard convolution neural network design, which involves the repetitive use of convolution layers, activation functions such as ReLU, and max pooling. Following this path results in lower spatial resolution feature maps but more channels capturing complex features. The expanding path (decoder) is responsible for accurately determining the exact location of the segmented items in question. Transposed convolutions are used to enhance the resolution of feature maps acquired from the contracting route. This upsampled feature map set is then joined, via skip connections, with the related feature map set of same scale. The skip connections aid in preserving the intricate spatial information that is diminished during the downsampling process. The expansion path steadily improves feature map spatial resolution while lowering channels, creating a segmentation map comparable to the original image. The key benefit of U-Net is its ability to efficiently combine semantic information, enabling accurate segmentation results. This is especially advantageous for operations that need precise assessment of the exact position of objects.

Depth-wise separable convolution boosts the efficiency of convolutional layers in U-Net like networks. It does this by decomposing a typical convolution into two distinct stages: It starts by processing each input channel independently through a kernel. Each channel's spatial features are captured in this way. This phase employs a 1x1 convolution operation to merge the results obtained from the depth-wise step and project them onto a fresh set of channels. This combines data from many channels to provide new features. Conventional convolutions concurrently learn filters for all input and output channels. Depthwise separable convolution greatly decreases parameter count by independently learning filters for each input channel (depth-wise) and then combining them using a compact 1x1 point wise convolution. This results in a lighter model that could use less memory and train more quickly. Given an input feature map $X \in R^{H\,x\,W\,x\,C}$ where H denotes height, W width, and C channel count. The depthwise convolution applies a filter K_c of height M and width N to each channel c independently.

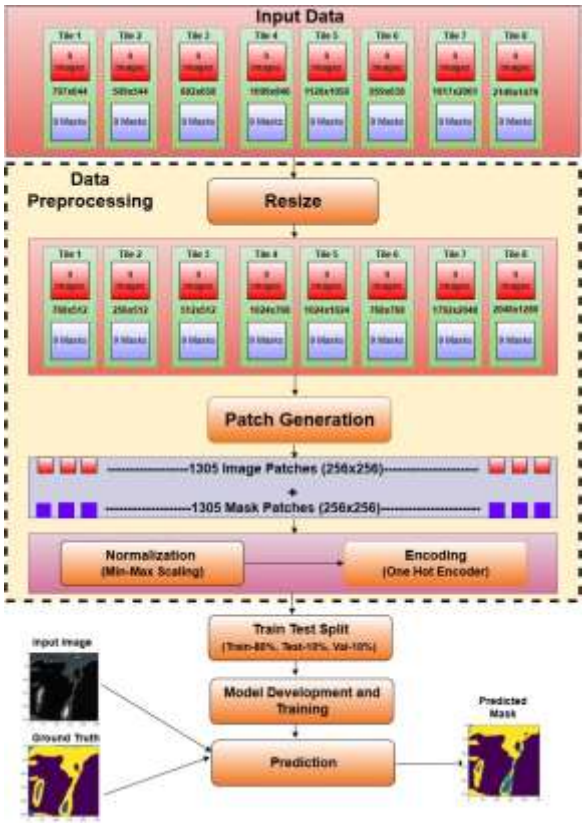


Figure 3. Methodology of Proposed Model

The depthwise convolution output Y_c for channel c is given by:

$$Y_c(g,h) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} K_c(m,n).\ X_c(g+m, h+n) \quad (1)$$

The pointwise convolution merges channels by imposing a 1 × 1 convolution on the depthwise output. Given, depthwise convolution output $Y \in R^{H\,x\,W\,x\,C}$ and pointwise filter $P \in R^{1\,x\,1\,x\,C\,x\,D}$,the pointwise convolution output $Z \in R^{H\,x\,W\,x\,D}$ is:

$$Z(g,h,d) = \sum_{c=0}^{C-1} P(1,1,c,d).Y_c(g,h) \quad (2)$$

The overall operation of depthwise separable convolution can be expressed as the combination of depthwise and pointwise convolutions:

$$Z(g,h,d) = \sum_{c}^{C-1} P(1,1,c,d).\left(\sum_{m=0}^{M-1} \sum_{n=0}^{N-1} K_c(m,n).\ X_c(g+m, h+n)\right) \quad (3)$$

Bi-linear interpolation is a technique used to increase the resolution of a picture. While U-Net does not directly use it in its convolution layers, it may be utilized alongside U-Net for tasks like as image segmentation that need the enlargement of feature maps. Bi-linear interpolation estimates the target region by weighting the four pixels close to it in the upsampled image.

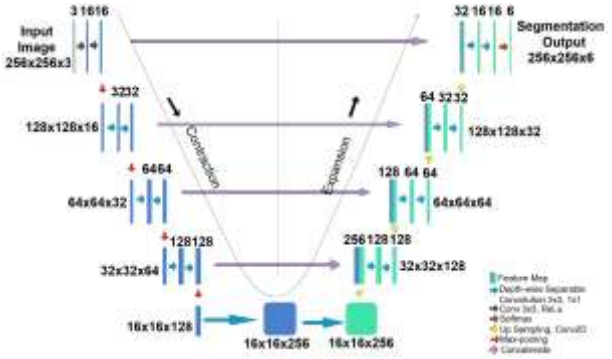


Figure 4. U-Net Architecture with Depth-wise Separable Convolution

This technique facilitates the generation of a more seamless progression between pixels and prevents the appearance of pixelated imperfections, in contrast to less sophisticated approaches such as nearest-neighbor interpolation. In the U-Net model, bi-linear interpolation is used to upsample feature maps during the decoding phase. Given a point (l,m) in the upsampled feature map, the four surrounding points in the lower resolution feature map are:

$$(l_1, m_1), (l_2, m_1), (l_1, m_2), (l_2, m_2)$$

with the corresponding feature values:

$$f(l_1, m_1), f(l_2, m_1), f(l_1, m_2), f(l_2, m_2)$$

The bi-linear interpolation function $f(l, m)$ to compute the value at the point (l, m) is given bm:

$$f(l, m) = \sum_{i=1}^{2} \sum_{j=1}^{2} f(l_i, m_j).w_{ij}(l, m) \quad (4)$$

where the weights $w_{ij}(l, m)$ are given bm:

$$w_{11}(l, m) = (l_2 - l)(m_2 - m) \quad (5)$$
$$w_{12}(l, m) = (l - l_1)(m_2 - m) \quad (6)$$
$$w_{21}(l, m) = (l_2 - l)(m - m_1) \quad (7)$$
$$w_{22}(l, m) = (l - l_1)(m - m_1) \quad (8)$$

U-Net uses this interpolation on all of the pixels in the feature map that has been upsampled; this makes the transition from lower resolution feature map's values smooth and guarantees that the output's spatial dimensions are identical to the input images. The architecture is shown in figure 4.

3.5 Proposed U-Net with Ghost Convolution

Han et al. (2020) proposed a ghost module to divide the computational cost of a standard convolution into intrinsic feature maps and ghost feature maps which are computed using cheap linear operations. Mathematically, for an output with C_{out} channels, the ghost module computes only $C' = \frac{C_{out}}{r}$ intrinsic feature maps using standard convolutions, and generates the remaining $C_{out} - C'$ using depth-wise convolutions or other inexpensive operations.

Let F_{std} represent the set of features generated by standard convolution and F_{ghost} be the approximated set generated by cheap transformations. The final feature representation is given by:

$$F_{out} = F_{std} \cup F_{ghost} \quad (9)$$

where $|F_{std}| = C'$ and $|F_{ghost}| = C_{out} - C'$

In the proposed semantic segmentation framework, the baseline U-Net inspired with encoder-decoder structure is modified using ghost convolution module to significantly decrease the computational requirements whilst maintaining a high degree of segmentation accuracy on high-resolution satellite images, shown in figure 5. The network takes input of dimension 256x256x3 which is a compromise between spatial resolution and a small set of available hardware resources. The architecture of an encoder starts with a traditional convolution block, which downsamples an input image and increases the number of channels (3 to 12). Later layers use a twofold hierarchical downsampling pathway: every layer will have two cascaded ghost modules, and that at every stage, the count of channels would be halved evenly. The encoder is a hierarchical convolution structure which successively reduces spatial dimension by half, and increases semantic channels by the same amount. Dropout layers in an intermediate level control the process of learning (an average 0.1-0.3). The output feature maps cover 128x128x24, 64x64x48, 32x32x96 and 16x16x192 bottleneck resolution levels. In order to maintain the quality of representation at a high level, to reduce the computational demands at the last stage, the ghost module with reduction ratio (r=2) is used. The encoder can approximately be associated with the decoder in architecture, restoring the spatial contexts by means of skip connections based on matching encoder layers. Each decoder stage too designs the use of ghost modules to maintain parameter efficiency, and feature concatenation ensures that fine grained spatial detail is maintained. A standard convolution block up-samples the reconstructed output to the original image resolution to create a prediction map with the size 256x256xN, where N is the overall number of semantic classes (N = 6 in the current study). The architecture presented here provide significant parameter reduction and in floating-point operations (FLOPs) with negligible performance impact, and hence may be highly appealing to real-time and edge-computing applications which are under the necessity to operate on remote-sensing data.

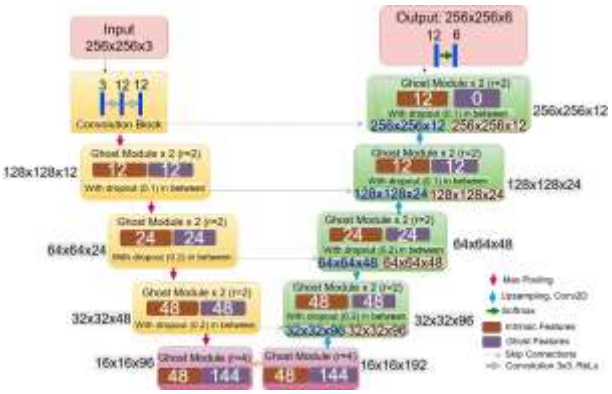


Figure 5. Proposed U-net Architecture with Ghost Convolution

3.6 Parameter Reduction in Ghost Modules

Let Input feature map be $X \in R^{H\times W\times C_{in}}$, Output feature map be $Y \in R^{H\times W\times C_{out}}$, standard convolution layer with Kernel size $k \times k$. The number of parameters in a standard convolution layer is:

$$Params_{standard} = k^2 \cdot C_{in} \cdot C_{out} \quad (10)$$

A ghost module decomposes the output feature generation into two stages: intrinsic feature generation and ghost feature approximation. The number of intrinsic features is defined as $C' = \left\lceil \frac{C_{out}}{r} \right\rceil$, where r denotes the ghost ratio, typically set to 2 or 4. The remaining output channels are synthesized as ghost features, given by $C_{ghost} = C_{out} - C'$.

In the first step, intrinsic feature maps are generated through a conventional convolutional layer. The number of learnable parameters involved in this intrinsic convolution is:

$$Params_{intrinsic} = k^2 \cdot C_{in} \cdot C' \quad (11)$$

In the second step, ghost features are created by applying lightweight operations, often depthwise and pointwise convolutions. Specifically, a depthwise 1×1 convolution is applied to each of the C' intrinsic channels, contributing:

$$Params_{depthwise} = 1^2 \cdot C'$$

Optionally, a channel adjustment convolution may be used to expand these ghost features to the required number when remaining channels differ from intrinsic channel, incurring additional parameters:

$$Params_{channel} = 1^2 \cdot C' \cdot C_{remaining} \quad (13)$$

where $C_{remaining} = C_{out} - C'$.

Combining all components, the total number of parameters in a Ghost module becomes:

$$Params_{ghost} = k^2 \cdot C_{in} \cdot C' + C' + C' \cdot (C_{out} - C')$$

Assuming that Cout is divisible by the ghost ratio r, the number of intrinsic feature maps becomes $C' = \frac{C_{out}}{r}$. Substituting this into the Ghost parameter expression:

$$Params_{ghost} = k^2 \cdot C_{in} \cdot \frac{C_{out}}{r} + \frac{C_{out}}{r} + \frac{C_{out}}{r}\left(C_{out} - \frac{C_{out}}{r}\right)$$

Now simplifying the expression:

$$= k^2 \cdot C_{in} \cdot \frac{C_{out}}{r} + \frac{C_{out}}{r} + \frac{C_{out}}{r} \cdot \left(\frac{(r-1) \cdot C_{out}}{r}\right)$$

$$Params_{ghost} = \frac{k^2 C_{in} C_{out}}{r} + \frac{C_{out}}{r} + \left(\frac{(r-1) \cdot C_{out}^2}{r^2}\right) \quad (15)$$

To quantify the efficiency of ghost module, parameter reduction ratio R, defined as the ratio between number of parameters in the ghost module and those in a standard convolution:

$$R = \frac{Params_{ghost}}{Params_{standard}} \quad (16)$$

Taking the ratio of Ghost Module parameters to standard convolution parameter gives the reduction ratio:

$$R = \frac{\frac{k^2 C_{in} C_{out}}{r} + \frac{C_{out}}{r} + \left(\frac{(r-1) \cdot C_{out}^2}{r^2}\right)}{k^2 \cdot C_{in} \cdot C_{out}} \quad (17)$$

$$R = \frac{1}{r} + \frac{1}{r k^2 C_{in}} + \frac{(r-1) C_{out}}{r^2 k^2 C_{in}} \quad (18)$$

The parameter ratio R between a ghost module and a standard convolution can be asymptotically analyzed by decomposing its expression into three additive terms. The leading term is $\frac{1}{r}$, which reflects the proportion of parameters used in the intrinsic convolution. This term is independent of the number input and output channels, and thus remains constant as these values grow. It represents the core efficiency gain of Ghost modules — a reduction by a factor of r in the number of parameters compared to standard convolution layers.

The second term in the expression is $\frac{1}{r k^2 C_{in}}$, which decays inversely with the number of input channels. As $C_{in} \to \infty$, this term tends to zero rapidly. The decay is further accelerated with larger kernel sizes k and higher reduction ratios r. This term accounts for the parameter cost of lightweight operations applied during the generation of ghost features, which are independent of the output dimension. Consequently, its contribution becomes negligible in deep CNN layers with high channel counts.

The third and most complex term is given by $\frac{(r-1)C_{out}}{r^2k^2C_{in}}$. It depends on both the output and input channels and can be approximately expressed as $\frac{C_{out}}{C_{in}}$. Its asymptotic behavior varies based on the relative growth of these dimensions. If C_{in} and C_{out} grow proportionally — as is common in deeper layers — the term remains bounded and decays like $O\left(\frac{1}{C_{in}}\right)$. If $C_{out} \ll C_{in}$, it decays even faster. However, in the rare case where $C_{out} \gg C_{in}$, the decay is significantly slower, and the term may become nonnegligible. This reflects the computational cost of generating the remaining $\frac{r-1}{r}$ ghost features.

Overall, as $C_{in} \to \infty$ (and Cout scales similarly), both the second and third terms diminish, and the parameter ratio converges to the leading term. Therefore, the asymptotic behavior of the ghost module can be succinctly approximated as:

$$R \approx \frac{1}{r} + O\left(\frac{1}{C_{in}}\right)$$

confirming that for sufficiently large channel dimensions, ghost modules provide a near-ideal reduction in parameter cost proportional to $\frac{1}{r}$, with minimal overhead. This final expression makes it evident that as the ghost ratio r increases, the number of learnable parameters in the ghost module becomes significantly smaller than that of the standard convolution, particularly due to the quadratic dependence of the original on C_{out}. Thus, ghost modules provide a computationally efficient alternative while preserving representational capacity.

3.7 Performance Metrics

To assess the model performance Accuracy, Intersection over Union (IoU), Dice Loss, Focal Loss, Tversky Loss along with the combined loss are analyzed.

$$Accuracy = \frac{\sum_{i=1}^{N} 1\{\widehat{y_i} = y_i\}}{N} \quad (20)$$

$$IoU = \frac{|A \cap B|}{|A \cup B|} \quad (21)$$

$$Dice_c = \frac{2\sum_{i=1}^{N} p_{i,c} g_{i,c}}{\sum_{i=1}^{N} p_{i,c} + \sum_{i=1}^{N} g_{i,c}} \quad (22)$$

$$Dice\ Loss = \frac{1}{C}\sum_{c=1}^{C}\left(1 - \frac{2\sum_{i=1}^{N} p_{i,c} g_{i,c}}{\sum_{i=1}^{N} p_{i,c} + \sum_{i=1}^{N} g_{i,c}}\right) \quad (23)$$

$$Focal\ loss = \frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_{i,c}(-\alpha_c(1 - p_{i,c})^{y} log(p_{i,c})) \quad (24)$$

$$Tversky\ Loss = \frac{1}{C}\sum_{c=1}^{C}\left(1 - \frac{\sum_{i=1}^{N} p_i^c g_i^c}{\sum_{i=1}^{N} p_i^c g_i^c + \alpha\sum_{i=1}^{N} p_i^c(1-g_i^c) + \beta\sum_{i=1}^{N}(1-p_i^c) g_i^c}\right) \quad (25)$$

Integrating dice loss, focal loss, and tversky loss into a combined loss function may greatly improve the performance of semantic segmentation tasks for remote sensing photos. Combining tversky, focal, and dice loss functions allows the model to use the best aspects of each loss function. A fair distribution of weights guarantees that overall segmentation quality (dice loss), resilience against challenging cases (focal loss), and effective management of class imbalance (tversky loss) are all adequately addressed. By assigning weights of 0.50 to Dice Loss, 0.25 to Focal Loss, and 0.25 to tversky Loss, we may adjust the model's prioritization of the loss function's various components. Improved flexibility to certain features of remote sensing images may result from this.

$$\mathcal{L}_{custom} = 0.25 * \mathcal{L}_{focal-loss} + 0.5 * \mathcal{L}_{dice-loss} + 0.25 * \mathcal{L}_{tversky-loss} \quad (26)$$

3.8 Hyper Parameters

The proposed lightweight U-Net model is optimized using the Adam optimizer, with its default parameters ($\beta_1 = 0.9, \beta_2 = 0.999, and\ \epsilon = 1e^{-7}$). To balance convergence speed and model stability, a learning rate of 0.001 is employed. The model is trained for 100 epochs with a batch size of 64, ensuring efficient gradient estimation and computational feasibility. To prevent over fitting and improve generalization, dropout regularization was applied at varying rates across the network layers. Specifically, a dropout rate of 0.1 is applied to the first two convolution layers, 0.2 to the next two, and 0.3 to the middle bottleneck layer. These hyper parameters are selected through heuristic experimentation and guided by established practices in the literature, resulting in a model configuration that effectively balances performance and computational efficiency.

3.9 Experimental Setup

For experimentation purposes, a windows 11 machine is used in conjunction with the open-source PyTorch 2.2.1 framework with CUDA 12.1 equipped with an Intel Core i7-10870H CPU, an NVIDIA GeForce GTX 1650 Ti graphics card with 4 GB of VRAM, and 32 GB of RAM. All models were trained and tested under identical hardware and software conditions to ensure fair comparison.

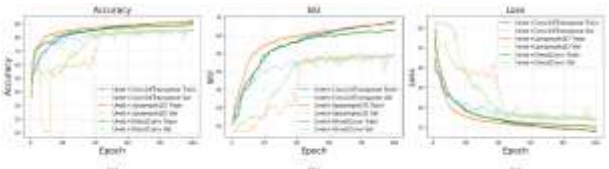


Figure 6. Performance Analysis (a) Accuracy, (b) IoU (c) Loss

4. RESULT AND DISCUSSION

A performance comparison of six variants of U-Net indicates diverse segmentation results. UNet+ResNet34 gave the best training accuracy (97.15%), IoU (88.01%), but clearly indicated overfitting with a validation IoU of 52.8% and a loss of 12.15%. In a similar manner UNet+ResNet18 achieved 95.66% training accuracy and 74.42% IoU but validation IoU did not exceed 60.71%. Extreme Overfitting was experienced with UNet+ MobileNetV2- even though the training accuracy was quite high at 96.22%, the validation accuracy was very poor (72.43%) with a loss of 26.56%. The classical approaches Conv2DTranspose and UpSample2D demonstrated intermediate values of validation IoU 49.83% and 49.05%, and validation loss 13.46% and 13.71%, correspondingly. UNet+ GhostConv, on the contrary, has had 89.17% and 63.06% training accuracy and IoU, respectively, validation accuracy is 85.56% and IoU is 48.9%, which is fairly close, implying that there is no significant overfitting. The UNet+GhostConv gives a fair compromise, attaining a validation loss of 13.51% and is a solid, well-performing, and generalizable model to consider once it comes to segmentation. A comparison of training and validation metrics for two classical approaches with proposed ghost method are shown in Figure 6.

In the reviewed segmentation models, UNet+GhostConv showed the best efficiency with a comparable MIoU of 58.40%, the lowest computer power expenditure (0.38 GFLOPs) and model size (1.88 MB). The recorded MIoU-per-GFLOP (153.68) and MIoU-per-MB (31.06) were the best figures, and thus it is highly appropriate to implement it in resource limited setups. Compare and contrast, classic architectures such as U-Net, consisting of ResNet34 and ResNet18 backbone, performed moderately well in regards to MIoU (58% and 60% respectively), but a much-increased FLOPs (27.20 and 17.54 GFLOPs correspondingly) and memory (96.18 and 57.63 MB). The U-Net with depth separable convolution has also shown high results in terms of accuracy (87.11%) and MIoU (58.56%), but its advantage is a balance in terms of performance and low complexity of the model (1.11 GFLOPs, 1.49 MB) as shown in Table 2. These findings exemplify that high-performing, yet with significantly reduced computational requirements, lightweight models, especially those based on ghost, can enable high segmentation performance.

Table 2. Comparison of Models Based on FLOPs, Model Size, and Performance Metrics

Model	Flop (G)	Model Size (MB)	MIoU	MIoU/ GFLOP	MIoU/MB
UNet+Resnet 34	27.20	96.18	58.63	2.16	0.61
UNet+Resnet 18	17.54	57.63	59.98	3.42	1.04
UNet+Cov2dTranspose	3.44	7.41	58.31	16.95	7.87
UNet+Mobilenetv2	1.71	11.13	33.20	19.42	2.98
UNet+Upsample2D	1.11	1.49	58.56	52.76	39.30
UNet+GhostConv	0.38	1.88	58.40	153.68	31.06

Multi Objective Assessment Framework

To holistically evaluate the models, a multi-objective assessment framework, which couples the accuracy of segmentation part with the resources used. The framework determines three main evaluation dimensions, raw segmentation performance, which is the result of metrics Mean Intersection over Union (MIoU), computational efficiency which is calculated as MIoU per GFLOP, and memory efficiency that is calculated as MIoU per MB of the model size. To overcome the heterogeneity of these measures we use MinMax normalization that leads to the transformation of the measurement to unit interval between 0 and 1.

$$N_i = \frac{x_i - x_{min}}{x_{max} - x_{min}} \quad (27)$$

These normalized components are combined in the multi-objective score in a weighted linear form:

$$S = \alpha . N_{MIoU} + \beta . N_{CompEff} + \gamma . N_{MemEff}, \quad subject\ to\ \alpha + \beta + \gamma = 1 \quad (28)$$

Five different weighting schemes are analysed, to fit various deployment requirements: Balanced ($\alpha = 0.4, \beta = 0.3, \gamma = 0.3$) is used for general purpose applications. Prioritized Segmentation accuracy ($\alpha = 0.6, \beta = 0.2, \gamma = 0.2$), definitely prioritizing segmentation performance in the application, Computeoptimized ($\alpha = 0.2, \beta = 0.6, \gamma = 0.2$)for fast inference focused scenarios, Memory optimized ($\alpha = 0.2, \beta = 0.2, \gamma = 0.6$) for restricted resource environments, Pure Efficiency focused ($\alpha = 0.0, \beta = 0.5, \gamma = 0.5$) for high resource optimization. This approach allows the ranking of different models objectively across a wide variety of constraints in operation whilst ensuring interpret-ability by using a linear model and attaching explicit weights, allowing one to select the model that is most suitable in typical conditions of deployment. The ghost network is Pareto-optimal with multi-objective scores of 0.87 to 0.95 at all priority weightings. This computational efficiency of the architecture (153.68 MIoU/GFLOP) and the small parametric footprint of 1.88MB reflect efficient reduction of feature redundancies without sacrificing representational capacity.

The ghost model got an almost perfect score of 0.91 compared to 0.78 of the second-best performing Unet+DepthseparableConv under the balanced priority conditions. Such performance boost was even higher in scenarios where the focus of competition was on the computational efficiency, with the Ghost model scoring 0.95 as opposed to 0.59 of the depthseparable counterpart. Conventional UNet+Con2dTranspose model yielded median results with a balanced and accuracy-guided condition with a score of 0.46 and 0.62 respectively, and low computational efficiency with a score of 0.28 with compute-first priority. The implementations of ResNet-based models which perform both ResNet18 and ResNet34 obtained the low scores in each of the evaluation criteria and the ResNet34 model scored zero in case of the pure efficiency priority as shown in Figure. 7. The UNet based on the MobileNetV2 performed especially poorly on all scores between 0.04 and 0.09, proving to be unfit on this type of segmentation task, even though it is known to be efficient on mobile devices. Raw performance scores indicated that with individual MIoU scores set at 59.98%, UNet+ResNet18 gave the best individual MIoU score, but ghostconv offered the best combination of accuracy and efficiency, with an MIoU score of 58.40%, computational efficiency of 153.68 MIoU/GFLOP, and memory efficiency of 31.06 MIoU/MB. The 1.88MB model size and 0.38 GFLOPs computational complexity of the Ghost architecture enable its deployment in resource limited applications, which is especially desirable due to the competitive segmentation performance. These results indicate that the ghost convolution method is an interesting trade-off when high segmentation performance and operation efficiency are the required properties. The qualitative results of model as shown in Figure. 8. Extensive experiments validate the effectiveness of proposed architecture and chosen image processing technique. This constancy across several criteria implies that the proposed lightweight U-Net is learning the aerial pictures underlying properties, making it a promising state-of-the-art choice for remote sensing segmentation solutions.

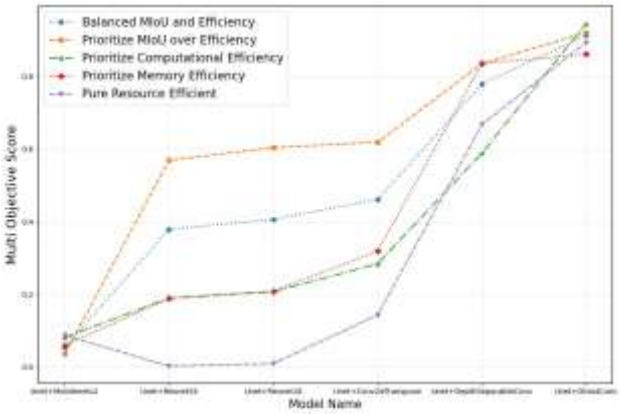


Figure 7. Comparative Model Performance across different Priorities

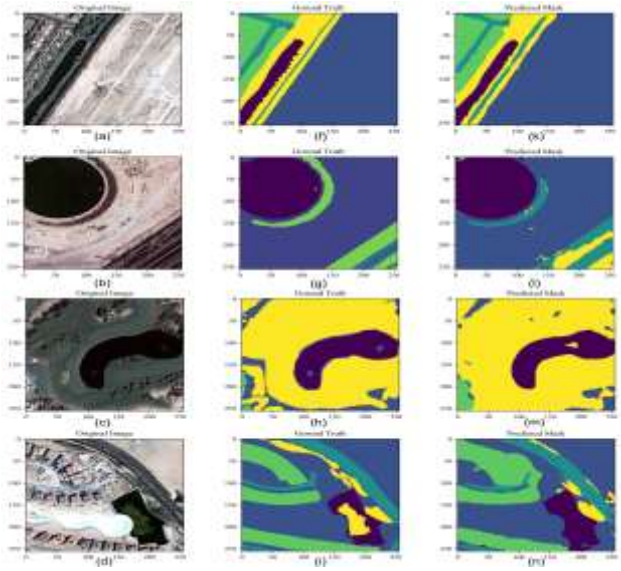


Figure 8. (a)-(d): Original Image, (f)-(i): Ground Truth, (k)-(n): Predicted Mask

5. CONCLUSION

In the high-resolution earth observation world, supporting semantic segmentation accurately and efficiently under resource-constrained settings is a need and a challenge. A U-Net variant, named as UNet+GhostConv, is proposed in this work that fulfills this need by drastically decreasing the computational intensity without compromising accuracy widely. Out of six diversely tested model configurations, UNet+GhostConv performs best with its compact size model with only 1.88 MB and 0.38 GFLOPs of computational cost for an input image of shape 256x256x3 yet has a competitive Mean IoU of 58.4% and an 87.3% test accuracy. More importantly, it has the lowest MIoU per GFLOP (153.68), which makes it the performance of GhostConv can be affected by using the datasets characterized by various levels of resolutions, noise levels, or object scales, thus affecting its out-of-the-box generalizability. Besides, the segmentation accuracy can also be enhanced regarding boundary refinement and identifying rare classes and in the future effort, hybrid designs or multi-scale context aggregation might be needed. Overall, the presented work introduces a low-cost, scalable, and modular niche in the segmentation framework due to balancing the computation affordability and the performance of algorithms. It establishes a basis of further research regarding hybrid lightweight architectures to increase segmentation accuracy even further with maintained efficiency in context of real-world, resource-limited deployments.

Conflict of Interest

The authors declare that they have no conflict of interest.

Author Contributions

Deepak Gupta developed the methodology, conducted experiments and wrote the main manuscript. All authors

contributed in conceptual design, result analysis, reviewed and approved the final manuscript.

Funding:
This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

References:

Avinash, K., Chavan, P., N. K. B., & H. K. H. (2025). Semantic segmentation of high-resolution satellite images using deep learning. In *Proceedings of the 9th International Conference on Computational System and Information Technology for Sustainable Solutions (CSITSS)* (pp. 1–4). https://doi.org/10.1109/CSITSS67709.2025.11295249

Badrinarayanan, V., Kendall, A., & Cipolla, R. (2017). SegNet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 39*(12), 2481–2495. https://doi.org/10.1109/TPAMI.2016.2644615

Boguszewski, A., Batorski, D., Ziemba-Jankowska, N., Dziedzic, T., & Zambrzycka, A. (2022). LandCover.ai: Dataset for automatic mapping of buildings, woodlands, water and roads from aerial imagery [arXiv preprint]. https://arxiv.org/abs/2005.02264

Fan, X., Yan, C., Fan, J., & Wang, N. (2022). Improved U-Net remote sensing classification algorithm fusing attention and multiscale features. *Remote Sensing, 14*(15), Article 3591. https://doi.org/10.3390/rs14153591

Fu, J., Liu, J., Jiang, J., Li, Y., Bao, Y., & Lu, H. (2021). Scene segmentation with dual relation-aware attention network. *IEEE Transactions on Neural Networks and Learning Systems, 32*(6), 2547–2560. https://doi.org/10.1109/TNNLS.2020.3006524

Guo, Y., Li, Z., Chen, E., Zhang, X., Zhao, L., Xu, E., ... Liu, L. (2021). A deep fusion U-Net for mapping forests at tree species levels with multi-temporal high spatial resolution satellite imagery. *Remote Sensing, 13*(18), Article 3613. https://doi.org/10.3390/rs13183613

Han, K., Wang, Y., Tian, Q., Guo, J., Xu, C., & Xu, C. (2020). GhostNet: More features from cheap operations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 1577–1586). https://doi.org/10.1109/CVPR42600.2020.00165

Humans in the Loop. (2024). *Semantic segmentation dataset*. Retrieved July 8, 2026, from https://humansintheloop.org/resources/datasets/semantic-segmentation-dataset-2/

Liu, S., & Shi, Q. (2020). Local climate zone mapping as remote sensing scene classification using deep learning: A case study of metropolitan China. *ISPRS Journal of Photogrammetry and Remote Sensing, 164*, 229–242. https://doi.org/10.1016/j.isprsjprs.2020.04.008

Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 3431–3440). https://doi.org/10.1109/CVPR.2015.7298965

Neupane, B., Horanont, T., & Aryal, J. (2021). Deep learning-based semantic segmentation of urban features in satellite images: A review and meta-analysis. *Remote Sensing, 13*(4), Article 808. https://doi.org/10.3390/rs13040808

Pastorino, M., Moser, G., Serpico, S. B., & Zerubia, J. (2022). Fully convolutional and feedforward networks for the semantic segmentation of remotely sensed images. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)* (pp. 1876–1880). https://doi.org/10.1109/ICIP46576.2022.9897336

Rashkovetsky, D., Mauracher, F., Langer, M., & Schmitt, M. (2021). Wildfire detection from multisensor satellite imagery using deep semantic segmentation. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 14*, 1–1. https://doi.org/10.1109/JSTARS.2021.3093625

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention* (pp. 234–241). Springer. https://doi.org/10.1007/978-3-319-24574-4_28

Seth, A., & Sharma, S. (2022). Semantic segmentation: A systematic analysis from state-of-the-art techniques to advance deep networks. *Journal of Information Technology Research, 15*(1), 1–28. https://doi.org/10.4018/JITR.299388

Spasev, V., Dimitrovski, I., Kitanovski, I., & Chorbev, I. (2024). Semantic segmentation of remote sensing images: Definition, methods, datasets and applications. In M. Mihova & M. Jovanov (Eds.), *ICT innovations 2023: Learning: Humans, theory, machines, and data* (pp. 127–140). Springer. https://doi.org/10.1007/978-3-031-54321-0_9

Sylvain, J.-D., Drolet, G., & Brown, N. (2019). Mapping dead forest cover using a deep convolutional neural network and digital aerial photography. *ISPRS Journal of Photogrammetry and Remote Sensing, 156*, 14–26. https://doi.org/10.1016/j.isprsjprs.2019.07.010

Toth, C., & Jóźków, G. (2016). Remote sensing platforms and sensors: A survey. *ISPRS Journal of Photogrammetry and Remote Sensing, 115*, 22–36. https://doi.org/10.1016/j.isprsjprs.2015.10.004

Ulku, I. (2024). ResLMFFNet: A real-time semantic segmentation network for precision agriculture. *Journal of Real-Time Image Processing, 21*(4), Article 101. https://doi.org/10.1007/s11554-024-01474-0

Xiao, T., Liu, Y., Zhou, B., Jiang, Y., & Sun, J. (2018). Unified perceptual parsing for scene understanding. In V. Ferrari, M. Hebert, C. Sminchisescu, & Y. Weiss (Eds.), *Computer Vision – ECCV 2018* (pp. 432–448). Springer. https://doi.org/10.1007/978-3-030-01228-1_26

Zhang, Y., & Chi, M. (2020). Mask-R-FCN: A deep fusion network for semantic segmentation. *IEEE Access, 8*, 155753–155765. https://doi.org/10.1109/ACCESS.2020.3012701

Zhao, H., Shi, J., Qi, X., Wang, X., & Jia, J. (2017). Pyramid scene parsing network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 2881–2890). https://doi.org/10.1109/CVPR.2017.660

Zhao, H., Zhang, Y., Liu, S., Shi, J., Loy, C. C., Lin, D., & Jia, J. (2018). PSANet: Point-wise spatial attention network for scene parsing. In *Computer Vision – ECCV 2018: 15th European Conference, Munich, Germany, September 8–14, 2018, Proceedings, Part IX* (pp. 270–286). Springer. https://doi.org/10.1007/978-3-030-01240-3_17

Deepak Gupta
Amity School of Engineering and Technology, Lucknow, Amity University Uttar Pradesh, India
deepakgupta.cse6@gmail.com
ORCID: 0009-0005-9481-4551

Om Prakash Singh
Amity School of Engineering and Technology, Lucknow, Amity University Uttar Pradesh, India
opsingh@amity.edu
ORCID: 0000-0002-9946-4141

Satyasundara Mahapatra
Pranveer Singh Institute of Technology, Kanpur, Uttar Pradesh India
satyasundara123@gmail.com
ORCID: 0000-0002-9587-6659