

Deep neural network for semantic segmentation of satellite images

Sardor Karimov^{1*}, *Dildora Sotvoldiyeva*¹, *Durbek Khalilov*¹ and *Nurillo Mamadaliyev*¹

¹Fergana branch of TUIT named after Muhammad Al-Khwarizmi, Fergana, Uzbekistan

Abstract. Deep neural networks have become a crucial tool for satellite image processing, particularly in semantic segmentation tasks. This paper explores the use of deep neural networks for automated feature extraction and classification in Earth satellite images. It focuses on how deep architectures like U-Net and MobileNet handle multi-channel spectral data to achieve precise segmentation of various land covers and objects of interest. The paper discusses data preprocessing techniques, loss function selection, and optimization, along with examples of successful applications in mapping, agricultural monitoring, and urban planning. The study highlights the effectiveness of deep neural networks in addressing complex satellite image segmentation challenges and showcases their potential for future research and practical use in land management and environmental monitoring.

1 Introduction

Semantic image segmentation involves identifying and isolating specific regions within an image that correspond to various object classes. In the context of Earth remote sensing (ERS), this technique is applied across multiple fields, including geoinformatics, georesource engineering, automated relief map creation, urban planning, land use analysis, and ice monitoring. Despite advances in object classification methods and algorithms, the segmentation problem remains only partially resolved. In many cases, certain steps of the process still require manual intervention by operators, leading to considerable time consumption. Furthermore, there is no universally accepted approach that underlies most algorithms, nor is there a general algorithm that can achieve optimal segmentation for all types of images.

There are currently numerous methods for semantic image segmentation that utilize convolutional neural networks (CNNs). These methods generally achieve good performance and accuracy when segmenting relatively small images, using various CNN architectures for classification and dense labeling. However, neural network-based image processing methods have certain limitations and are not without flaws. Additionally, the high resolution of satellite images leads to significant computational costs during segmentation. Therefore, developing techniques to reduce these costs while improving segmentation quality is a highly relevant challenge.

* Corresponding author: sk363688@gmail.com

At present, there are no standardized guidelines for designing CNN architectures, such as the number of layers, the number and size of feature maps, the size of convolutional filters, or the choice of training algorithm. When designing a CNN, it is important to consider that using too few parameters may decrease classification accuracy, while an excessive number of parameters increases computational complexity without necessarily improving the network's classification performance..

Semantic segmentation in computer vision involves assigning a semantic label to each region of an image. Most modern CNN architectures for image segmentation are based on the principles outlined in [1]. The key idea is to adapt traditional CNN classification so that the output is a class probability map instead of a probability vector. Typically, a standard CNN serves as an encoder to generate feature maps at various levels of image decomposition. The encoder is then followed by a decoder that resizes the feature maps to match the original spatial dimensions of the input image, producing a heat map for each class. Deep semantic segmentation networks are generally built on the principles of fully convolutional networks and encoder-decoder architectures [2].

An encoder typically consists of a series of convolutional layers followed by batch normalization (BN) and a rectified linear activation function (ReLU), with pooling layers following the convolution blocks. Essentially, the encoder operates as a standard convolutional network trained to classify the input image. The decoder mirrors the encoder in terms of layers and serves to interpolate the encoder's output. In the final layer of the decoder, a 1×1 convolution is commonly used, followed by a sigmoidal activation function, to generate the segmented output image.

U-Net [3] and MobileNet [4] are two distinct neural network architectures designed for different purposes. U-Net is often employed for image segmentation, featuring convolutional layers for extracting features and transposed convolutional layers to expand the dimensions and produce detailed segmented maps.

In the U-Net architecture, max-pooling is applied over a 2×2 pixel region during pooling operations. As the input image passes through several layers of convolution and subsampling, it is transformed into abstract feature maps, which serve as the outputs of the corresponding encoding blocks. U-Net can be seen as a modified version of earlier networks, as it combines the output data from the decoder layers with feature maps from the encoder at matching layers. In this architecture, interpolation in the decoder layers is achieved using transposed convolution [5]. U-Net has demonstrated significant improvements in segmentation accuracy across various types of images and can perform well even when trained on small datasets. However, its drawbacks include relatively low performance and high resource demands due to the complexity and resource-intensive nature of its encoders, such as ResNet [6], Inception [7], and EfficientNet [8].

MobileNet is specifically designed for image classification while minimizing the number of parameters, making it well-suited for mobile devices and embedded systems due to its lightweight architecture. In U-Net, MobileNet serves as an efficient encoder to extract features from input images. Its streamlined design allows for fast image processing and efficient feature extraction. By replacing the standard convolutional layers in U-Net's encoder, MobileNet reduces both the number of parameters and the computational load, making it ideal for resource-intensive situations. MobileNet is often used as an encoder in U-Net when high performance is needed in environments with limited resources.

2 The solution methods

Earth remote sensing images are known for their high resolution, with an average size often exceeding 2000×2000 pixels. In contrast, most CNNs are designed for input images with a resolution of 256×256 pixels. When the original image is divided into smaller fragments for

processing, the total segmentation time increases in proportion to the number of fragments. To efficiently segment large images, it is essential to boost the performance of the neural network architecture while preserving segmentation accuracy. As previously mentioned, the U-Net architecture provides good accuracy and is therefore used in this context. One method to enhance its performance is by using high-efficiency encoders. MobileNet is a notable example of such an encoder, reducing memory usage during computation while maintaining high prediction accuracy. This pre-trained network can even operate on mobile devices.

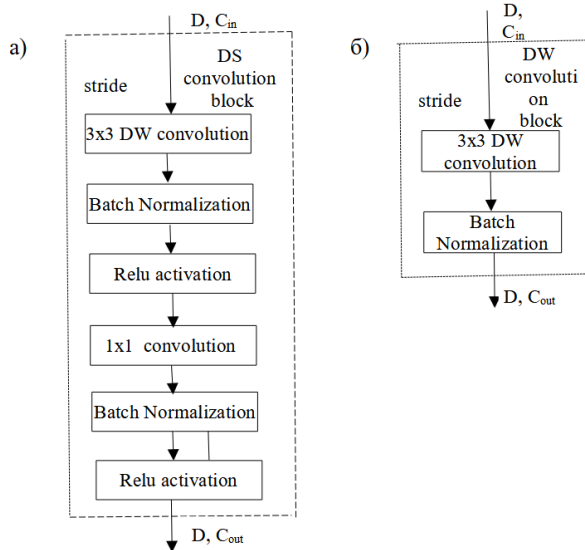


Fig.1. Convolution blocks: a) depth separable convolution block (DS); b) depth convolution block (DW); D - image dimensionality, C_{in} , C_{out} - number of input and output channels, stride - convolution step

The MobileNet architecture utilizes depthwise separable convolution (DSW), which breaks down standard convolution into two steps: depthwise convolution (DW) and pointwise convolution (a 1×1 convolution). In a standard convolution, filtering and combining inputs into new outputs occur simultaneously. However, with depthwise separable convolution, these processes are split. First, depthwise convolution applies a single filter to each input channel, and then pointwise convolution combines the results. This separation of filtering and combining into two distinct layers significantly reduces computational complexity and the overall model size.

The network architecture, as depicted in Fig. 1, consists of two types of blocks. The first type employs depthwise separable convolution (DSW), incorporating batch normalization and ReLU activation functions. The second type utilizes depthwise convolution, also with batch normalization and the same activation function. Most of these blocks use a convolution stride of 1, while a stride of 2 is used to reduce spatial dimensions. These convolution blocks form two types of basic blocks at the convolutional level. The structures of these block types are shown in Fig. 2 and Fig. 3. The base block of the first type features a residual connection, where the result of successive DS and DW convolutions is added to the output of a separate depthwise convolution.

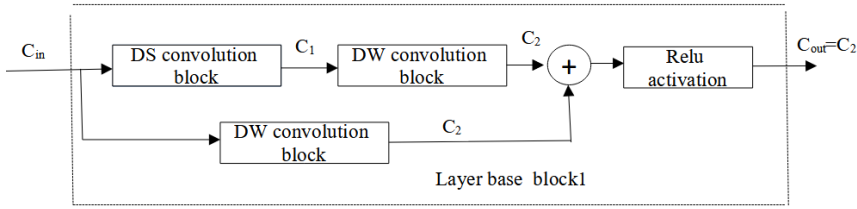


Fig. 2. Structure of the basic convolution level block of the first type. C_{in} , C_{out} - number of input and output channels; C_1 - number of channels of depth separable convolution, C_2 - number of channels of depth convolution

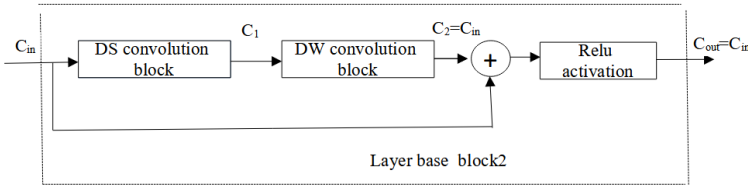


Fig. 3. Structure of the basic block of the convolution level of the second type. C_{in} , C_{out} - number of input and output channels; C_1 - number of channels of depth separable convolution, C_2 - number of channels of depth convolution

Each convolutional layer of the network is formed by a combination of these basic blocks. Fig. 4 shows the structure of a convolutional network layer consisting of a base block of the first type and N base blocks of the second type.

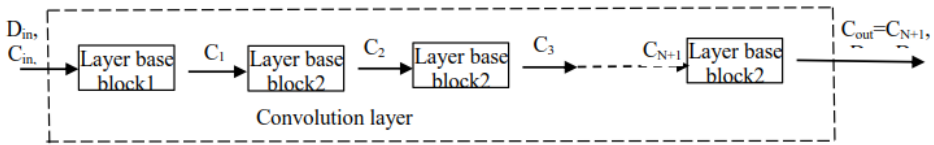


Fig. 4. The convolutional layer of the network. D_{in} , D_{out} - dimensions of input and output feature maps; C_{in} , C_{out} - input and output number of channels; C_1 - number of channels of the basic block of the first type, C_2, \dots, C_{N+1} - number of channels in the corresponding blocks of the second type

The encoding section of the network begins with a standard convolutional layer featuring a 3x3 kernel and a stride of 2, followed by batch normalization and max pooling. This is succeeded by a series of convolutional layers, each consisting of a sequence of basic convolutional blocks (as shown in Figure 4). Each convolutional layer typically increases the number of filters while reducing the spatial dimensions of the feature maps. In this study, the encoder is composed of four such convolutional layers.

The decoding section of the network comprises multiple layers. Each layer first upsamples the input feature map and then merges it with the feature map from the corresponding encoder layer. This combined result is subsequently processed using depthwise convolutions, along with batch normalization and ReLU activation functions. The structure of the decoder layer employed in this study is illustrated in Figure 5.

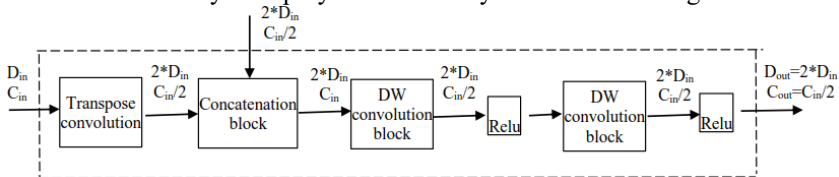


Fig.5. Structure of the decoding layer. D_{in} , D_{out} - dimensions of input and output feature maps; C_{in} , C_{out} - input and output number of channels

To enhance the dimensionality of the input feature map, transpose convolution is applied to double its size, while reducing the number of channels by half. This adjustment aligns the feature map's dimensions and channel count with those of the previous encoder layer. Upon merging these maps, a new feature map is created that matches the dimensionality of the corresponding encoder layer and has the same number of channels as the input. This feature map is then processed using two depthwise convolution blocks. The resulting feature map is passed to the next decoding layer. The final decoding layer takes both the feature map from the previous layer and the input image. It includes an additional deep convolution with a channel count corresponding to the number of image classes and a sigmoidal activation function. The output of this layer is a "heat" map, which serves as the segmented image.

3 Pre-processing

High-resolution satellite images are too large to be processed directly by neural networks. Simply downscaling these images leads to a loss of segmentation accuracy. To address this, the initial preprocessing step involves dividing both the dataset images and their corresponding masks into smaller slices. For the network design, an input image size of 512x512 pixels was chosen. This choice helps to balance computational efficiency during training and usage while maintaining adequate segmentation accuracy. The resulting image and mask fragments are resized to this specified dimension. In this study, each original image is divided into 16 segments.

Each color channel of the image fragment is normalized to a uniform range. For mask fragments that originally have three channels, these are transformed into multi-channel images where each channel represents a different type. Pixels in each channel are encoded with a specific value if they match the channel type, while pixels that do not match are encoded as zero. This process results in a dataset composed of image fragments along with their corresponding masks.

4 Experimental results

To train the network, a pre-trained LandCover dataset [10] is utilized, which has been augmented with labeled images from Central Asian regions. This dataset includes 1146 high-resolution satellite images of various land surfaces. Along with the images, it features masks that indicate the land surface type for each pixel. The dataset covers several types: agricultural land, pastureland, barren land, buildings and structures, forested areas, and water areas. Any surface types not listed are categorized as unknown. Each mask is an image where pixel values are encoded with specific colors corresponding to different surface types. All images from the original dataset undergo the preprocessing steps described earlier. Consequently, a working dataset is created, consisting of 18,336 image fragments and their corresponding masks. This dataset is then randomly split into training, validation, and test sets, comprising 60%, 20%, and 20% of the total dataset, respectively.

The network is trained using the transfer learning approach. The encoding portion of the network utilizes a modified version of the MobileNet architecture, which has been pre-trained on the ImageNet dataset [10]. Consequently, the initial weights of the encoder are directly set using those from the pre-trained model. In contrast, the weights for the decoding section of the network are initialized randomly. This method significantly reduces the training time required for satellite image segmentation.

For training, the Adam optimization algorithm is employed to iteratively update the network weights. This algorithm is an enhancement of stochastic gradient descent. A learning rate of 10^{-5} was used for training, along with mechanisms to automatically adjust the

learning rate and to halt training early if the losses did not improve over several epochs. The network was trained with batch normalization and used a batch size of 4, determined through experimentation.

To evaluate the segmentation results, a network with a U-Net architecture, configured to process the same images, was also trained. This network was trained using the same dataset and training parameters. During its training, the U-Net network adjusts approximately 31.4 million parameters. In contrast, the proposed MbUnet network requires tuning only 20.6 million parameters. This demonstrates a significant reduction in computational cost with the proposed network. Figure 6 illustrates the training curves for both the U-Net and MbUnet networks, showing that the MbUnet architecture achieved a 3% improvement in segmentation accuracy.

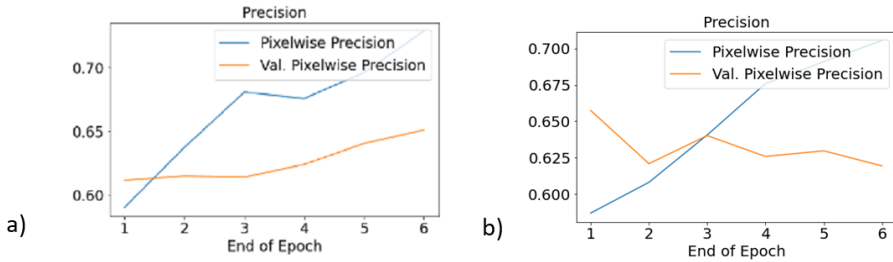


Fig.6. Graphs of Unet and MbUnet networks training as a function of training epochs: a) change of accuracy for Unet; b) change of accuracy for MbUnet

Testing was carried out using a trained neural network on a set of test images. Each processed test image results in 16 segmented sections, with each section being a multi-channel binary image. The number of channels corresponds to the different types of land surfaces segmented. For visualization purposes, these multi-channel images are converted into standard RGB images using a color table, where each land surface type is assigned a specific color. The final result is a full-size image created by assembling these 16 segmented images, showcasing the segmentation results. An example of this satellite image segmentation is illustrated in Figure 7.

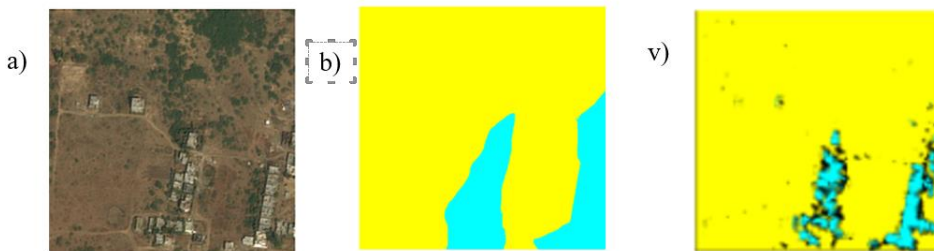


Fig.7. Example of satellite image segmentation: a) original image; b) marked mask; c) segmentation result

During testing, using the corresponding labeled masks from the test set, the pixel-by-pixel segmentation accuracy is also determined. The average accuracy on the test set was about 64%, which corresponds to the accuracy on the validation set used to train the network.

5 Conclusion

The proposed method achieves high accuracy in satellite image segmentation compared to existing solutions. This approach utilizes an encoder-decoder architecture akin to the U-Net

network, with a modified MobileNet network serving as the encoder. In the decoder, each layer's input feature maps are upsampled using transposed convolution, then combined with the corresponding encoder layer's feature maps and processed through depthwise convolutions. By employing depthwise convolutions in both the encoder and decoder, the approach significantly reduces computational costs and enhances the performance of the deep neural network.

References

1. Long J., Shelhamer E., Darrell T. *Fully convolutional networks for semantic segmentation*. Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3431-3440 (2015)
2. Bai, Kunlun. A comprehensive introduction to different types of convolutions in deep learning. Towards data science. 2019; <https://towardsdatascience.com/a-comprehensive-introduction-to-different-types-of-convolutions-in-deep-learning-669281e58215>.
3. Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. Medical Image Computing and Computer-Assisted Intervention–MICCAI. 2015; 18th International Conference, Munich, Germany, October 5-9.
4. Howard, Andrew G., et al. Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:**1704**, 04861 (2017)
5. Zeiler, Matthew D., Graham W. Taylor, and Rob Fergus. Adaptive deconvolutional networks for mid and high level feature learning. International conference on computer vision. IEEE, (2011)
6. He, Kaiming, et al. Identity mappings in deep residual networks. Computer Vision–ECCV: 14th European Conference, Amsterdam, The Netherlands, October 11–14. (2016)
7. Szegedy, Christian, et al. Rethinking the inception architecture for computer vision. Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.
8. Tan, Mingxing, and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. International conference on machine learning. 2019.
9. Howard, Andrew G., et al. Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:**1704**, 04861 (2017)
10. Boguszewski, Adrian, et al. LandCover. ai: Dataset for automatic mapping of buildings, woodlands, water and roads from aerial imagery. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. (2021)
11. Grishkin V. et al. *DETECTION OF FERTILE SOILS BASED ON SATELLITE IMAGERY PROCESSING*. CEUR Workshop Proceedings, pp. 251-255 (2021)
12. Grishkin V. M., Karimov S. I. AIP Conference Proceedings **2432(1)**, (2022)
13. Grishkin V., Karimov S., Fedorova A. Physics of Particles and Nuclei **55(3)**, 545-547 (2024)
14. Porubay O. et al. E3S Web of Conferences. **538**, 01028 (2024)
15. Khalilov D. et al. E3S Web of Conferences. **508**, 04011 (2024)
16. D.M.Umurzakova, "Neuro-fuzzy control algorithm of dynamic objects with uncertainty of a priori information," 2020 International Conference on Information Science and Communications Technologies, ICISCT 2020, (2020), 9351462. doi: 10.1109/ICISCT50599.2020.9351462.
17. D.M.Umurzakova, "Mathematical Modeling of Transient Processes of a Three-pulse System of Automatic Control of Water Supply to the Steam Generator When

- the Load Changes,” 2020 Dynamics of Systems, Mechanisms and Machines (Dynamics), (2020), 1-4. doi: 10.1109/Dynamics50954.2020.9306117.
18. D.M.Umurzakova, “System of automatic control of the level of steam power generators on the basis of the regulation circuit with smoothing of the signal,” *IUM Engineering Journal*, pp. 287-297, vol. **22(1)**. (2021), doi: 10.31436/iiumej.v22i1.1415.
 19. I.X.Siddikov, D.M.Umurzakova, “Synthesis Algorithm for Fuzzy-logic Controllers,” 2020 Dynamics of Systems, Mechanisms and Machines (Dynamics), 1-5(2020). doi: 10.1109/Dynamics50954.2020.9306165.
 20. I.X.Siddikov, D.M.Umurzakova, “Synthesis of adaptive control systems of a multidimensional discrete dynamic object with a forecasting models,” 2019 International Conference on Information Science and Communications Technologies (ICISCT), 1-5(2019). doi: 10.1109/ICISCT47635.2019.9012033.