IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 23 February 2023, accepted 7 March 2023, date of publication 10 March 2023, date of current version 17 May 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3255988

RESEARCH ARTICLE

Using Layer-Wise Training for Road Semantic Segmentation in Autonomous Cars

SHAHRZAD SHASHAANI[1], MOHAMMAD TESHNEHLAB[1], AMIRREZA KHODADADIAN[2], MARYAM PARVIZI[2], THOMAS WICK[2], AND NIMA NOII[3]

[1]Intelligent Systems Laboratory, Faculty of Electrical and Computer Engineering, K. N. Toosi University of Technology, Tehran 16317-14191, Iran
[2]Institute of Applied Mathematics, Leibniz Universität Hannover, 30167 Hannover, Germany
[3]Institute of Continuum Mechanics, Leibniz Universität Hannover, 30167 Hannover, Germany

Corresponding author: Mohammad Teshnehlab (teshnehlab@eetd.kntu.ac.ir)

The work of Maryam Parvizi was supported by the Alexander von Humboldt Foundation Project "$\mathcal{H}$-matrix approximability of the inverses for FEM, BEM, and FEM-BEM coupling of the electromagnetic problems."

ABSTRACT A recently developed application of computer vision is pathfinding in self-driving cars. Semantic scene understanding and semantic segmentation, as subfields of computer vision, are widely used in autonomous driving. Semantic segmentation for pathfinding uses deep learning methods and various large sample datasets to train a proper model. Due to the importance of this task, accurate and robust models should be trained to perform properly in different lighting and weather conditions and in the presence of noisy input data. In this paper, we propose a novel learning method for semantic segmentation called layer-wise training and evaluate it on a light efficient structure called an efficient neural network (ENet). The results of the proposed learning method are compared with the classic learning approaches, including mIoU performance, network robustness to noise, and the possibility of reducing the size of the structure on two RGB image datasets on the road (CamVid) and off-road (Freiburg Forest) paths. Using this method partially eliminates the need for Transfer Learning. It also improves network performance when input is noisy.

INDEX TERMS Autonomous cars, layer-wise trains, computer vision, convolution neural networks, semantic segmentation.

I. INTRODUCTION

Image semantic segmentation is based on pixel-level classification and is used widely for image or scene understanding. In this kind of segmentation, there is no difference between various objects of the same class which is common in instance segmentation. Segmentation can extract vital information from a given image pixel by pixel; therefore, it is widely used when the shape of objects is not known exactly and it can vary in different scenes. In this method, a final label will not be generated for the given image, on the contrary, each pixel has its label at the end of the process. Many computer vision tasks like self-driving cars, augmented reality wearables, home-automation devices [1], etc., need scene perception at a very low level, such as the pixel level.

Convolutional Neural Networks (CNNs) can detect and localize each object of a specific class in the given image and label each pixel. In recent years, large labeled datasets alongside powerful computers have helped Deep Convolutional Neural Networks (DCNNs) outperform many common computer vision algorithms [1]. For these reasons, various architectures of CNNs for segmentation approaches, such as Alexnet [2], ResNet [3], VGGnet [4], and GoogleNet [5] have been applied in many types of research [6]. For this, a well-pre-trained architecture is used as a base model to achieve higher accuracy [6], [7]. Then, the existing model is fine-tuned with the destination dataset by using Transfer Learning (TL) and making some changes to the pre-trained model in some cases.

Outdoor perception is more challenging due to dynamic and complex situations such as light, color, and weather conditions in different time slots. Even in structured outdoor environments, such as urban roads, there are still

The associate editor coordinating the review of this manuscript and approving it for publication was Mehul S. Raval.

several challenges for rare object detection, like puddles [8]. Autonomous driving needs rich and robust information about scene understanding for the segmentation process [6]. Image segmentation has been frequently used in autonomous driving on both road and off-road paths. Several labeled datasets for road and off-road path segmentation for supervised learning exist. Most existing labeled datasets for semantic segmentation are mainly based on RGB cameras. Also, there are low sample datasets based on RGB-D, LiDAR, near-infrared sensors, etc. [8].

In this paper, we use an effective well-designed DCNN called ENet [1] with layer-wise training to boost the training phase in time and compact the so-called DCNN without significant loss of accuracy. The main advantage of this method is determined when our input data is noisy. {The main contributions of the proposed training method can be listed as follow:

- r-wise training requires no Transfer Learning and is only trained on the target dataset.
- Adding noise to the input images can produce robust results since the features have been extracted more accurately through layer-wise training.
- Training epochs for the final training can be significantly reduced.
- A layer-wise training method can reduce the model's size by removing some encoder layers without significantly affecting IoU.

We use the DCNN architecture for two well-defined datasets called CamVid and Freiburg Forest that cover both urban and off-road areas. First, we review some of the existing methods and datasets in Section II. In section III, we discuss the ENet structure and our approach for layer-wise training. Then, in Section IV, we evaluate the performance of our proposed learning method using two different datasets. Finally, we summarize the information and results in Section V.

II. PRIOR ART

In this section, we will discuss semantic segmentation and different prior semantic segmentation methods in autonomous driving. In the first part, there is a short review of existing datasets in road segmentation with varying sensors for both road and off-road paths. In the second part, methods based on road segmentation are discussed.

Image semantic segmentation is a fundamental task in the field of computer vision with wide usage from three-dimensional reconstruction [9] to self-driving cars. Image semantic segmentation is a pixel-level classification task, so at the end of the process, each pixel is labeled in a specific class. This algorithm recognizes each category, labels pixels according to the recognized category, and provides location information about given categories [9]. Considering these advantages, it is used in road scene understanding applications that require the ability to recognize different forms and understand the spatial information between classes [10]. In typical road scenes, most pixels belong to large classes, such as roads, and objects in the background, such as sky and buildings, so the network must produce smooth segmentation to achieve appropriate accuracy for smaller-size classes [10].

TABLE 1. Road and off-road track datasets based on RGB and/or LiDAR sensor data.

Name	Data Type		Type
	RGB	LiDAR	
Freiburg Forest [11]	✓		Off-road track
Yamaha-CMU [12]	✓		Off-road track
RELLIS-3D [13]	✓	✓	Off-road track
Off-Road Terrain [14]	✓		Off-road track
RUGD [15]	✓		Off-road track
KITTI [16]	✓	✓	Road track
CamVid [17]	✓		Road track
CityScapes [18]	✓		Road track
Mapillary [19]	✓		Road track

A. DATASETS

Several labeled datasets gathered from different types of sensors for road, and off-road path segmentation for supervised learning exist. For path detection, sensors, such as RGB cameras, RGB-D, light detection and ranging (LiDAR), and near-infrared sensors are mainly used [8]. But most existing labeled datasets for path semantic segmentation are usually based on RGB cameras and LiDAR sensors. A quick review of some of the existing datasets based on RGB and, or LiDAR sensor data is gathered in Table 1.

In the following, we will discuss several CNN architectures proposed for semantic segmentation tasks in autonomous driving. Autonomous driving on different paths relies heavily on computer vision for detecting paths and avoiding moving and still objects, such as pedestrians, bicyclists, other vehicles, obstacles, etc. So far, semantic segmentation with large sample datasets is used to acquire robust models for road segmentation tasks.

B. ROAD SEMANTIC SEGMENTATION METHODS

For DCNNs, a suitable and sufficient amount of data is needed. There are datasets with relatively high volume data and accurate labeling for path detection in urban areas. But there are a few datasets for off-road path detection, which are much smaller than the datasets in urban areas, and their labeling has been done with less accuracy. In supervised learning, for a precision prediction, the amount of data must be large and their labels must be accurate. Therefore, due to the existence of small datasets for off-road areas, many researchers use TL.

Some studies [10], [20], and [1] used different encoder-decoder architectures for semantic segmentation. These architectures extract multi-level feature maps in the encoder part, then recover spatial information step by step in the decoder part. For pixel-wise classification purposes, the decoder network performs non-linear upsampling on the low-resolution encoder feature maps and transfers them to full input-resolution feature maps. Encoder and decoder parts can be the same, or the structure of their layers can be completely different. Using residual blocks or reusing features can facilitate feature exploration in the encoder part. However, applying them in the decoder part causes feature map explosion, so they are not beneficial in decoders.

In [10], a novel and practical deep fully CNN architecture, called SegNet, is presented. SegNet has an encoder part similar to the VGG16 network with 13 convolutional layers, followed by a corresponding decoder network with a pixel-wise classification layer at the end. This model has been evaluated on the CamVid and SUN RGB-D [21] datasets. In [20], the orthogonal concepts for encoder-decoder architectures are combined, called Dual-Path Dense-Block Networks (DPDB-Net). The dense block incorporates feature reuse only for the encoder. The proposed architecture was evaluated on the Freiburg Forest and CamVid datasets. In [1], a novel efficient deep neural network architecture named ENet is proposed for tasks that require low-latency operations. This method is up to 18 times faster, requires 75 times fewer FLOPs, has 79 times fewer parameters, and provides similar or better accuracy to existing models such as SegNet. This model was evaluated on the CamVid, Cityscapes, and SUN datasets and compared the trade-offs between the network's accuracy and processing time between ENet and other state-of-the-art models. So, in this paper ENet is used as a base model.

Other approaches used by the papers are of mixed methods for robust segmentation. Learning from fused representations is one of them. For example, the article [22] proposed a novel semantic segmentation architecture and the Convoluted Mixture of Deep Experts (CMoDE) fusion techniques. CMoDE enables a multi-stream Deep Neural Network (DNN) to learn features from complementary modalities and spectra. The model comparatively evaluates class-specific features of expert networks based on the scene condition to learn fused representations. This model is evaluated on three publicly available datasets: Synthia [23], Cityscapes, and Freiburg Forest. Using a multi-task approach to share a common latent space is another way for robust segmentation. In [24], a multi-task approach is proposed by supplementing the semantic segmentation task with edge detection, semantic contour, and distance transform tasks. The complementary tasks can produce more robust representations that enhance semantic labels by sharing a common latent space. Also, the influence of contour-based tasks on latent space and their impact on the final results of semantic segmentation were explored. The effectiveness of learning in a multi-task setting for hourglass models in the Cityscapes, CamVid, and Freiburg Forest datasets was demonstrated by improving the state-of-the-art without any refinement on post-processing.

Another advantageous method is using the TL approach. In [6], a TL-based semantic segmentation of off-road driving environments is presented. First, a pre-trained segmentation network called DeconvNet is trained on Pascal VOC datasets. Because of the large size of DeconvNet, a smaller network, called the lightweight network, was proposed and then fine-tuned on the Freiburg Forest dataset. Also, to provide more accurate results, synthetic datasets that simulate the off-road driving environment (considering real-world variations) were used as the intermediate domain before training with real-world off-road driving data. The Freiburg Forest dataset was considered a real-world off-road driving dataset.

Regardless of the architecture, in some research [8], multimodal input or data fusion is used. The network can achieve faster convergence and accommodate more textual information while using multimodal images in segmentation. In [8], an attempt has been made to find the appropriate exploitation of different imaging methods for road scene segmentation versus using an RGB modality. A novel multi-level feature fusion network was proposed by exploring deep learning-based early and later fusion patterns for semantic segmentation. Using polarized cameras is a sensory enhancement that can significantly increase image perception abilities to detect highly reflective areas such as glass and water. The proposed multimodal fusion network outperforms unimodal networks and two typical fusion architectures. The model was evaluated on the Freiburg Multispectral Forest dataset.

III. PROPOSED METHOD

A. INTRODUCTION TO ENet

According to [1], a novel, fast, and efficient DNN architecture named ENet is proposed. A quick review of this network is shown in Figure 1.

Different internal components of the block are shown in Figure 2 with their input and output dimensions. As seen, in the defined blocks, there are three sequential convolution layers. Therefore, we are dealing with a deep structure where no matter how much we go towards using activation functions, such as ReLU and its derivatives to prevent gradient vanishing, the effect of the error on the final layers (decoder) is more than the initial layers (encoder). The encoder section is responsible for feature extraction and is the first layer that the input passes through. Therefore, it is essential to extract the correct and appropriate features and transfer them to the next layer. The weaker the features extracted in the initial layers of the network are, the less reliable the output of the network is in the decoder layers. These layers decode the extracted features and provide them to the full convolution part so that it can make decisions about semantic segmentation and appropriately label each pixel. As a result, the lack of proper training of the initial layers, as they are the input passage to the next layers, causes a decrease in the final performance of the network. The very proper training of

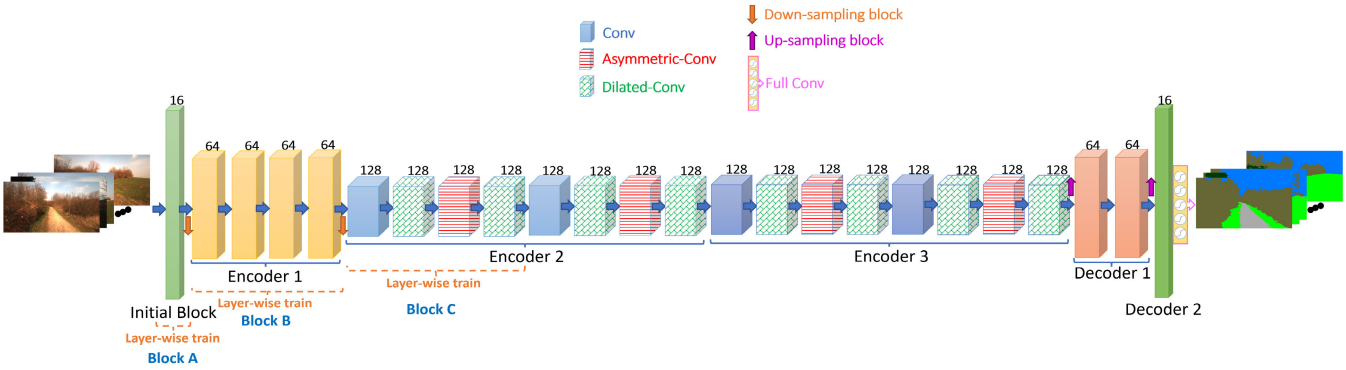

FIGURE 1. ENet structure [1]. Block structures with different convolution (Conv) types are shown. Conv: plane blocks, Asymmetric Conv: horizontal strips, Dilated Conv: diagonal brick blocks, Down Arrow: down-sampling block, Up Arrow: up-sampling block, and Full convolution block.

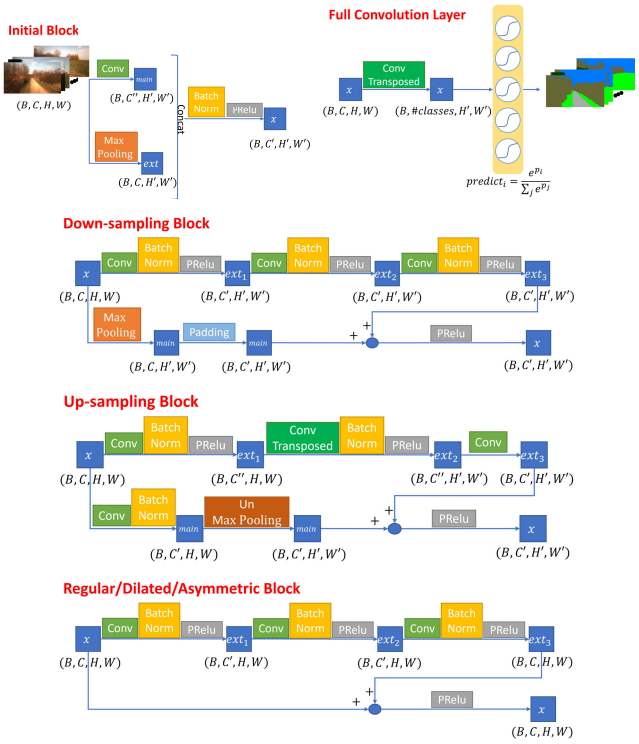

FIGURE 2. Different block structures used in ENet. B, C, H, and W represent the batch size, the number of channels/feature maps, height, and width respectively.

the final layers, which includes the decoder section, cannot correct the deficiencies and mistakes that occurred in the initial layers. The output of the network deteriorates when there is noise in the input data.

Based on the problems raised, we proposed the following training method, which can ensure that the primary layers are trained and perform the function of feature extraction.

B. LAYER-WISE TRAINING

As explained in Section III-A, the primary layers of the semantic segmentation network are responsible for feature extraction from the input images. In order to make sure that this part of the encoder is trained, we can train the initial part of the network individually. For this, it is enough to separate the layers of the encoder that we want to train separately from the overall structure and return them to the structure after training. In this way, the weights of these layers are changed during the separately explained training process to reach their desired value for feature extraction purposes. We call this method the layer-wise training approach.

This training process can be done independently of the available labels. Therefore, we used the unsupervised learning method. Our training was based on the Autoencoder (AE) training approach. Thus, we separate the desired part of the encoder section and add a completely distinct decoder part at the end of the available encoder part. We used sigmoid, linear, ReLU, and Tanh as decoder activation functions and the sig activation function provides more accurate results for overall network performance. Therefore, all results obtained in the following are based on the sig activation function. Now, this shallowly constructed network is an AE. We used the training set to feed the constructed AE, and it must predict the given inputs. Based on the unsupervised learning method, the network error was calculated by comparing the network outputs to the network inputs. Because the constructed network is not deep, it is possible to train all the initial layers of the network. Therefore, gradient vanishing is no longer a concern.

First, we separate a portion of the initial encoder layers from the original model, i.e., ENet, and put it as an encoder part, in the new AE. This portion is shown as Block A in Figure 1 as an example of the selected portion for the encoder part. To create the AE structure, we add the decoder part, which includes transposed conv, batch normalization, and an activation function for the structure after the encoder part. We added only one layer to the decoder part of the proposed AE network in order to keep the proposed network from becoming too deep and to train the encoder layers well. Choosing how many layers to separate from the original structure for layer-wise training depends on us. An example of the process of layer-wise training of Block A is shown in Figure 3.

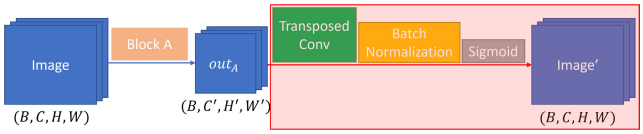

FIGURE 3. ENet layer-wise training for Block A. The red square is shown as the added distinct decoder part.

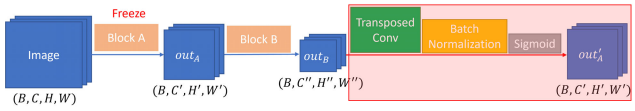

FIGURE 4. ENet layer-wise training for Block B. The red square is shown as the added distinct decoder part.

The constructed AE is trained using unsupervised learning. Then, the trained encoder part (Block A) is separated from the constructed AE and returned to the original structure. In this way, the separated encoder layer-wise training is completed. This process can be repeated for the following encoder blocks several times. For example, to use layer-wise training for the next layers of the encoder, i.e., Block B, we keep the part of the trained encoder, i.e., Block A, and remove the corresponding decoder for Block A. Then we separate the following block sequence of the encoder section from the main structure (Block B), and add it to the constructed structure after Block A. Then add a distinct decoder section matching the newly added encoder. This time the output of the decoder must be compared with the input of the newly added encoder, i.e., Block B. For only the weights of Block B to be trained, we can fix the weights of the previous block(s), i.e., Block A, so that their weights are not updated in the training process, and only the weights of the newly added sections in the constructed AE are trained and updated. In this way, we train the recently added encoder locally. The process of layer-wise training of Block B is shown in Figure 4.

Choosing blocks for layer-wise training can be done in different ways. An example of selecting the blocks for layer training is shown in Figure 1 and called Block A, Block B, and Block C. In this paper, this process is repeated three times for different block sequences of the encoder section totally. For training the first encoder, the Binary Cross Entropy loss was used, and for the rest of the two encoders' blocks, the MSE loss was applied. At the end of the layer-wise training process, the added decoder sections are removed completely, and the encoder part returns to its original structure. During the final classic training of the network, it is possible to enable or disable the training of the encoder's trained weights.

If we add the trained encoders in layer-wise training to their correspondence decoders, we can reconstruct the given images. There are some results of the reconstruction images after complete layer-wise training in Figure 5. The original image and the generated image share a great deal of similarity. Image details are well restored, and the only difference is that the reconstructed images are slightly redder than the original

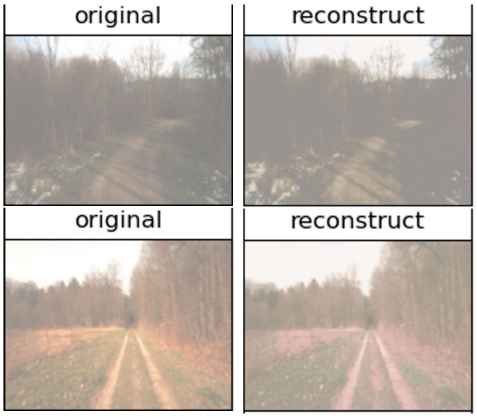

FIGURE 5. Results of ENet layer-wise training.

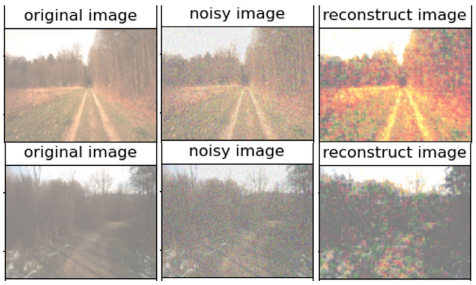

FIGURE 6. Results of ENet layer-wise training for noisy input.

images. We also provide an example for noisy input and the result for corresponding reconstructed output shown in Figure 6.

IV. SIMULATIONS

In this section, we evaluate our proposed model and compare its results with state-of-the-art models on Freiburg Forest and Camvid datasets.

A. DATASET DESCRIPTION

We performed experiments with two different datasets for the road semantic segmentation task. First, we use a real-world off-road autonomous vehicle dataset called the Freiburg Forest dataset. The second dataset is a real-world road scene understanding dataset for semantic segmentation tasks for urban areas called Camvid. In the following section, we describe each of them briefly.

1) FREIBURG FOREST

It is a dataset on forests scene with six classes: sky, road, tree, grass, vegetation, and obstacle. Off-road environments are unstructured (e.g., trails), unlike urban scenes that are highly structured (rigid and geometric objects, e.g., buildings) [24]. The dataset has 230 and 136 samples for training and test sets, respectively [20]. Images were collected at 20 Hz with a resolution of 1024 × 768 pixels on three different days to obtain the variability of data due to lighting conditions. All used samples are RGB images and fully labeled. Like [6],

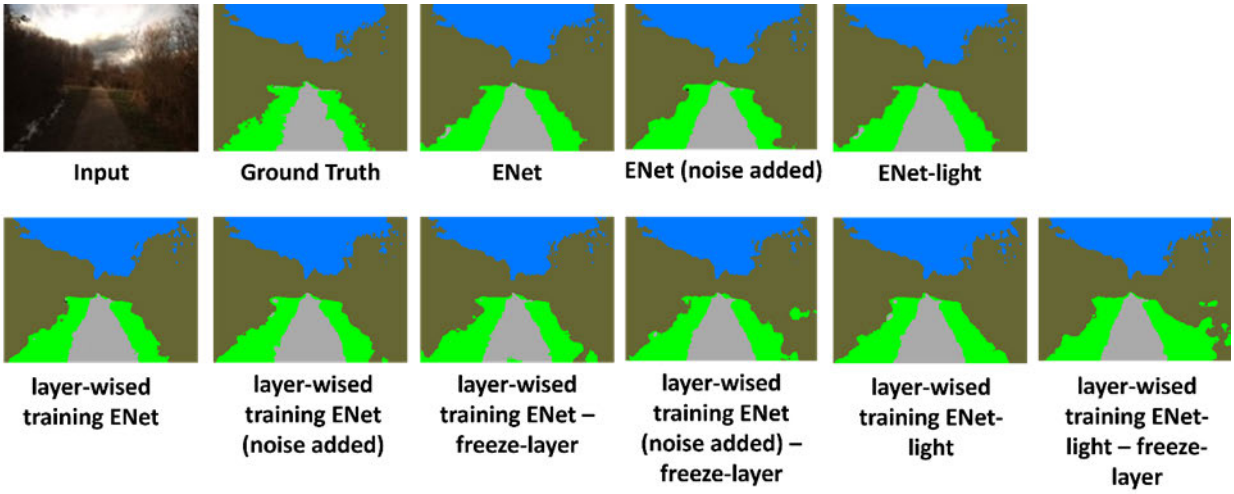

FIGURE 7. Result of the proposed learning method compared to the classic learning method for the Freiburg Forest dataset.

TABLE 2. Comparing layer-wise training vs classic training using ENet on the Freiburg Forest test set (IoU).

Models/Labels	Road	Grass	Vegetation	Sky	Obstacle	mIoU
ENet	87.43	85.35	**91.05**	**92.45**	1.57	71.57
ENet + noise	52.16	53.68	85.55	89.59	0.0	56.20
ENet-light	87.54	85.22	90.23	91.76	0.0	60.95
Layer-wise ENet	88.29	**85.71**	90.97	92.26	**17.79**	**75.0**
Layer-wise ENet + noise	88.62	85.61	90.96	92.05	0.16	71.48
Layer-wise ENet-light	88.49	84.29	90.49	92.39	0.0	71.13
Layer-wise ENet (freeze-layer)	**89.23**	85.60	90.59	92.19	5.14	72.55
Layer-wise ENet (freeze-layer) + noise	87.19	82.84	89.47	92.11	8.64	72.05
Layer-wise ENet-light (freeze-layer)	86.63	81.86	89.24	92.34	0.0	70.01

we merged tree and vegetation classes into a single class and use five classes instead of six classes in training.

2) CamVid

It is a real-world road scene understanding dataset for semantic segmentation that contains 12 classes: building, tree, sky, car, sign, road, pedestrian, fence, pole, sidewalk, cyclist, and unlabeled [24]. The dataset has 367, 101, and 233 samples for training, validation, and test sets, respectively [24]. All used samples are RGB images and fully labeled. We use the original image size, which is 360 × 480. For another experiment on this dataset, we use data augmentation and apply some transformations such as scaling (0.5, 1, 1.5), horizontal flipping, and rotating (0 to 30 degrees with a step of 5 degrees). Data augmentation has been applied only to the training set, and it has led to 15414 train samples. The final results are reported on the original test set.

TABLE 3. Comparing layer-wise training vs classic training using ENet on the Freiburg Forest test set (IoU).

Models/Labels	Road	Grass	Vegetation	Sky	Obstacle	mIoU	Total Acc
SegNet [24]	88.04	**88.04**	90.61	92.68	46.22	81.12	
CGBNet [24]	87.59	87.62	90.63	**92.78**	**46.58**	81.04	
SegNet+MTL [24]	88.04	**88.04**	90.61	92.68	46.22	74.58	
CGBNet+MTL [24]	87.59	87.62	90.63	**92.78**	**46.58**	77.89	
AdapNet [22]						**88.25**	
FCN8-LBP$_{8,7}$ [25]	72.0	78.2	84.0	87.3	16.6	56.4	
FCN8-RGB [25]	84.6	85.7	88.1	91.0	20.0	61.6	
FCN8-RGB-LBP$_{8,7}$ [25]	85.0	86.2	87.4	90.7	27.4	62.3	
Layer-wise ENet (OUR)	**88.29**	85.71	**90.97**	92.26	17.79	75.0	94.63
DPDB-Net - Full [20]	87.28	**87.8**	90.14	92.3		**89.4**	
ParseNet [20]	81.82	85.2	85.2	87.78		85	
M-Net [20]	82.41	84.93	88.7	89.26		86.3	
Fast-Net [20]	84.51	86.72	**90.66**	90.46		88	
GCN [20]	86.29	86.44	88.73	91.94		88.3	
CMnet(RGB) [8]	77.18	73.47	89.78	80.66		79.87	73.65
CMnet(RGB+LiDAR) [8]	81.01	76.55	90.64	83.25		81.64	6.62
Layer-wise ENet (OUR)	**87.45**	85.01	**90.66**	**92.36**		88.87	**97.15**

B. RESULTS

To compare the newly developed learning method with the conventional method, we measured the performance of these two methods in several ways. First, we evaluated the performance of both learning methods on the selected model (ENet). Then by adding random gaussian noise to the input data, only the training dataset, we assessed the performance of both learning methods. In this way, we compared the two training methods by changing the input training set without

TABLE 4. Comparing layer-wise training vs classic training using ENet on the CamVid test set (IoU).

Models/Labels	Sky	Building	Pole	Road	Pavement	Tree	Sign	Fence	Car	Pedestrian	Bicyclist	Unlabeled	Class Avg.	Class IOU
ENet	88.97	**66.30**	**19.48**	88.75	**71.16**	58.89	**17.23**	15.76	65.5	**23.04**	**28.56**	**24.2**	**49.42**	**47.32**
ENet + noise	87.31	61.14	0.07	78.0	32.56	32.73	2.13	2.53	51.35	3.10	0.01	23.24	31.90	31.18
ENet-light	**89.27**	65.06	0.47	**90.63**	69.80	**59.38**	15.38	14.55	**68.11**	19.55	8.91	22.86	45.6	43.67
Layer-wise ENet	89.07	65.66	0.62	**90.4**	**69.63**	53.62	7.84	6.17	**67.66**	19.3	9.88	23.99	41.99	43.62
Layer-wise ENet + noise	88.59	63.77	0.09	85.51	56.08	52.82	**12.04**	6.26	62.85	13.81	2.82	23.88	39.04	40.42
Layer-wise ENet-light	**89.82**	**66.0**	0.05	86.39	58.38	55.78	11.3	7.7	62.03	11.82	6.6	**24.56**	40.04	41.44
Layer-wise ENet (freeze-layer)	89.41	62.91	2.24	89.4	66.68	**56.0**	10.85	**12.35**	66.03	**20.7**	**17.33**	23.22	**43.09**	**44.9**
Layer-wise ENet (freeze-layer) + noise	88.42	62.41	0.26	82.06	44.56	49.99	9.85	8.29	65.69	19.36	2.82	24.1	38.15	39.43
Layer-wise ENet-light (freeze-layer)	87.11	64.19	**2.36**	80.65	53.47	53.94	10.55	7.23	58.93	13.57	7.3	19.67	38.25	39.94
ENet (Augmentation)	**90.69**	73.68	19.60	91.48	73.96	**65.07**	28.45	14.97	72.77	35.03	**44.41**	**28.90**	55.46	53.25
Layer-wise ENet (Augmentation)	90.49	**74.23**	**20.49**	**92.34**	**74.59**	64.22	**32.12**	**17.27**	**76.99**	**37.76**	44.38	28.04	**56.81**	**54.41**

applying any changes to the test set. The models trained by adding noise to their input are shown as (model name + noise) in Table 2 and Table 4. In the next step, we made some changes to the structure and made a comparison. This way, we removed the part marked as encoder 3 from the ENet structure and called it ENet-light in Table 2 and Table 4. So, we reduced the number of parameters and confined the feature extraction part to measure the performance of both training methods and determine whether or not the training of the initial layers influences the output. For classic and layer-wise training, the number of epochs is set to 300 and 200, respectively.

Results of the ENet with layer-wise training compared with different conditions compared to classic training are shown in Table 2 for the Freiburg Forest dataset. As we can see, for detecting the road and grass classes, layer-wise training has been able to perform better than classic training. For detecting the vegetation and sky classes, layer-wise training has a relatively similar performance with a slight difference. The obstacle class has a much smaller number of pixels than the other classes and is also very similar to the vegetation class. The performance of both methods in this class is very poor, and with the help of the layer-wise training method, we were able to improve the result a little. In general, the use of layer-wise training has improved the results for this dataset.

We also examined the performance of this method from two other perspectives. First, we added noise to the input data. With this, the result mIoU (mean intersection of unit) of the classic method drops by about 15%, while with the revised method, we have had a 3% and 0.5% drop in accuracy, for the all-layer-train and freeze-layer options, respectively. This test demonstrates an increase in the model's resistance to noise. Second, by using ENet-light, the accuracy of the model has been measured in both methods. In both methods, the detection rate of the obstacle class is significantly reduced, and the mIoU result of both methods is very close to each other.

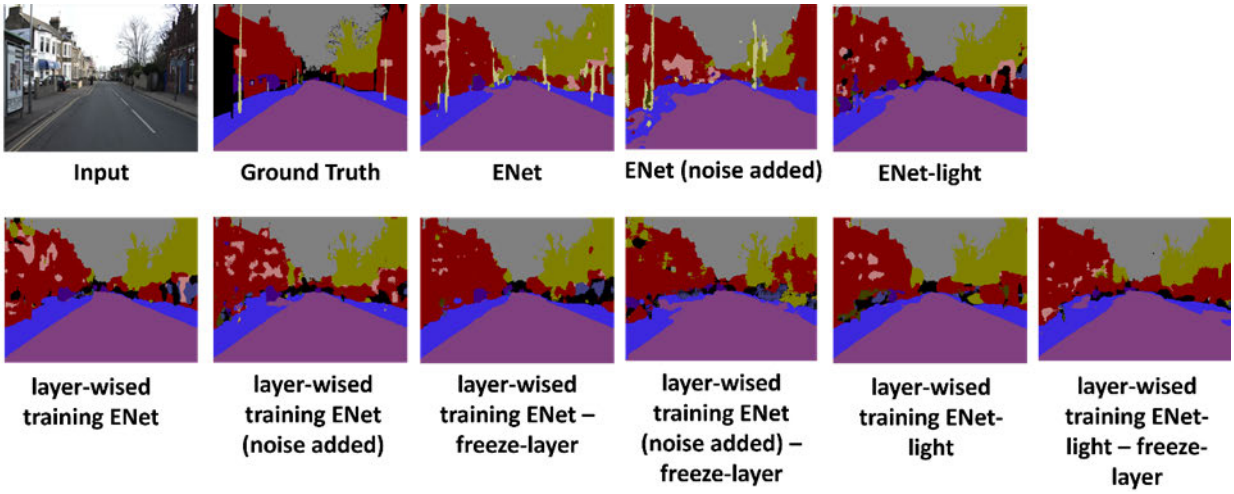

FIGURE 8. Result of the proposed learning method compared to the classic learning method for the CamVid dataset.

TABLE 5. CamVid test set results with sigmoid activation function for the layer-wise training in decoders.

Models/Labels	Sky	Building	Pole	Road	Pavement	Tree	Sign	Fence	Car	Pedestrian	Bicyclist	Unlabeled	Class Avg.	Class IOU
ENet [1]	95.1	74.7	35.4	95.1	86.7	77.8	51.0	51.7	82.4	67.2	34.1	-	68.3	51.3
FC-DenseNet56 [26]	92.4	77.6	32.6	92.8	79.9	72.0	31.8	26.2	73.2	37.9	31.1	-	-	58.9
SegNet(basic) [10]	91.2	75.0	44.8	93.3	74.1	84.6	36.9	47.5	82.7	55.0	16.0	-	62.9	-
SegNet [10]	92.4	88.8	27.5	97.2	84.4	87.3	20.5	49.3	82.1	57.1	30.7	-	65.2	55.6
Layer-wise ENet (freeze-layer)	89.41	62.91	2.24	89.4	66.68	56.0	10.85	12.35	66.03	20.7	17.33	23.22	43.09	44.9
Layer-wise ENet (Augmentation)	90.49	74.23	20.49	92.34	74.59	64.22	32.12	17.27	76.99	37.76	44.38	28.04	56.81	54.41

Results of the proposed model compared with other methods are shown in Table 3 for the Freiburg Forest. As we can see in the first dataset, for detecting the road class, which is very important, our model has been able to perform better than other methods. Identifying other classes, our model has nearly the same precision as other methods. The main difference is that our model performs poorly in identifying the obstacle class. One of the reasons for this low accuracy for the obstacle class is the high similarity of the obstacles with the tree (vegetation) class in the test data set.

Results of the ENet with layer-wise training compared with different conditions compared to classic training are shown in Table 4 for the CamVid dataset. Unlike the previous dataset, the use of the new training method could not improve the result. Only identifying the road class, which is of high importance, and the sky class improved. For other classes, the output of both methods was almost the same. The proposed method obtained poor results for classes with a small number of pixels in the dataset, such as the pole, fence, and bicyclist classes.

Similar to the previous dataset, we measured the new method with the two described approaches. By adding noise to the data, the final IoU of layer-wise training was higher than the classical method, and the result mIoU of the classic method dropped by about 16%, while with the revised method, we have had a 3% and 5% drop in mIoU, for the all-layer-train and freeze-layer options respectively. Then, by using ENet-light, in both methods, the mIoU result was reduced. In this approach, the final result mIoU of the classic method dropped by about 4%, while with the layer-wise training method, we have had a 2% and 5% drop in mIoU, for the all-layer-train and freeze-layer options,

respectively. By using an augmented train set, the final mIoU of both methods increased significantly, and in this way, the layer-wise training method outperform the classic training method.

Results of the proposed model compared with other methods are shown in Table 5 for the Camvid dataset. For this dataset, the use of an efficient model such as ENet and the use of the usual training method cannot surpass other introduced methods. Also, the newly introduced training method has not improved the final performance of the network compared to other methods. But if we use data augmentation, we can improve the IoU of the network for the classes that had poor performance. This will bring our results closer to the results of the presented papers.

The results of the proposed learning method compared to the classic learning method for both datasets are shown in Figure 7 and Figure 8.

V. CONCLUSION

In general, layer-wise training makes feature learning happen in the first and middle layers of encoders more effectively, and this makes the trained model more robust. We have shown this robustness by adding noise to the input, which has increased the final accuracy in comparison to classic training because the features have been extracted better in starter layers. Also, by adding layer-wise training to the training process, the training time may increase in general, but the number of final model training epochs can be reduced significantly. In addition, by adding layer-wise training, the model can become smaller by removing some encoder layers without much change in IoU.

Also, the idea of Transfer Learning has been used in most datasets with a small amount of data, such as Freiburg Forest. The network should be trained on a larger dataset in this method first. Then, the network should be trained again on the target dataset with changes to the network layers and sometimes without changing them. This task has a longer training time and requires a larger dataset. This means that layer-wise training requires no Transfer Learning, and only the target data set is used to train the network. Due to limited training data without Transfer Learning, the proposed learning method has not reduced the network detection ability, and as shown, the layer-wise trained networks are resistant to noisy data.

REFERENCES

[1] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, "ENet: A deep neural network architecture for real-time semantic segmentation," 2016, *arXiv:1606.02147*.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 2, pp. 84–90, Jun. 2012.

[3] D. Qiao and F. Zulkernine, "Drivable area detection using deep learning models for autonomous driving," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2021, pp. 5233–5238.

[4] D. K. Dewangan and S. P. Sahu, "Optimized convolutional neural network for road detection with structured contour and spatial information for intelligent vehicle system," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 36, no. 6, May 2022, Art. no. 2252002.

[5] J. Cheng, H. Li, D. Li, S. Hua, and V. S. Sheng, "A survey on image semantic segmentation using deep learning techniques," *Comput., Mater. Continua*, vol. 74, no. 1, pp. 1941–1957, 2023.

[6] S. Sharma, J. E. Ball, B. Tang, D. W. Carruth, M. Doude, and M. A. Islam, "Semantic segmentation with transfer learning for off-road autonomous driving," *Sensors*, vol. 19, no. 11, p. 2577, Jun. 2019.

[7] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3431–3440.

[8] Y. Zhang, O. Morel, M. Blanchon, R. Seulin, M. Rastgoo, and D. Sidibé, "Exploration of deep learning-based multimodal fusion for semantic road scene segmentation," in *Proc. 14th Int. Joint Conf. Comput. Vis., Imag. Comput. Graph. Theory Appl.*, 2019, pp. 336–343.

[9] P. Perera, M. Abavisani, and V. M. Patel, "In2I: Unsupervised multi-image-to-image translation using generative adversarial networks," in *Proc. 24th Int. Conf. Pattern Recognit. (ICPR)*, Aug. 2018, pp. 140–146.

[10] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A deep convolutional encoder–decoder architecture for image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 12, pp. 2481–2495, Dec. 2017.

[11] A. Valada, G. Oliveira, T. Brox, and W. Burgard, "Deep multispectral semantic scene understanding of forested environments using multimodal fusion," in *Proc. Int. Symp. Exp. Robot. (ISER)*, 2016, pp. 465–477.

[12] D. Maturana, P.-W. Chou, M. Uenoyama, and S. Scherer, "Real-time semantic mapping for autonomous off-road navigation," in *Field and Service Robotics*. Springer, 2018, pp. 335–350.

[13] P. Jiang, P. Osteen, M. Wigness, and S. Saripalli, "RELLIS-3D dataset: Data, benchmarks and analysis," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 1110–1116.

[14] G. Gresenz, J. White, and D. C. Schmidt, "An off-road terrain dataset including images labeled with measures of terrain roughness," in *Proc. IEEE Int. Conf. Auto. Syst. (ICAS)*, Aug. 2021, pp. 1–5.

[15] M. Wigness, S. Eum, J. G. Rogers, D. Han, and H. Kwon, "A RUGD dataset for autonomous navigation and visual perception in unstructured outdoor environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 5000–5007.

[16] J. Fritsch, T. Kuhnl, and A. Geiger, "A new performance measure and evaluation benchmark for road detection algorithms," in *Proc. 16th Int. IEEE Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2013, pp. 1693–1700.

[17] G. J. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla, "Segmentation and recognition using structure from motion point clouds," in *Proc. ECCV*, 2008, pp. 44–57.

[18] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 3213–3223.

[19] G. Neuhold, T. Ollmann, S. R. Bulo, and P. Kontschieder, "The mapillary vistas dataset for semantic understanding of street scenes," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 4990–4999.

[20] G. L. Oliveira, W. Burgard, and T. Brox, "DPDB-Net: Exploiting dense connections for convolutional encoders," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 4525–4531.

[21] S. Song, S. P. Lichtenberg, and J. Xiao, "SUN RGB-D: A RGB-D scene understanding benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 567–576.

[22] A. Valada, J. Vertens, A. Dhall, and W. Burgard, "AdapNet: Adaptive semantic segmentation in adverse environmental conditions," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2017, pp. 4644–4651.

[23] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez, "The SYNTHIA dataset: A large collection of synthetic images for semantic segmentation of urban scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 3234–3243.

[24] D. Saire and A. R. Rivera, "Empirical study of multi-task hourglass model for semantic segmentation task," *IEEE Access*, vol. 9, pp. 80654–80670, 2021.

[25] O. Mayuku, B. W. Surgenor, and J. A. Marshall, "Multi-resolution and multi-domain analysis of off-road datasets for autonomous driving," in *Proc. 18th Conf. Robots Vis. (CRV)*, May 2021, pp. 165–172.

[26] S. Jegou, M. Drozdzal, D. Vazquez, A. Romero, and Y. Bengio, "The one hundred layers tiramisu: Fully convolutional DenseNets for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jul. 2017, pp. 11–19.

SHAHRZAD SHASHAANI was born in Tehran, Iran, in 1998. She received the B.Sc. degree in computer engineering from K. N. Toosi University of Technology, in 2020, where she is currently pursuing the M.Sc. degree in artificial intelligence. Her research interests include artificial intelligence, machine learning, deep learning, computer vision, and image processing.

MOHAMMAD TESHNEHLAB was born in Borujerd, Iran, in 1957. He received the B.Sc. degree in electrical engineering from Stony Brook University, Stony Brook, NY, USA, in 1981, the M.Sc. degree in electrical engineering from Oita University, Japan, in 1991, and the Ph.D. degree from Saga University, Japan, in 1993. He is currently a Faculty Member with the Department of Electrical Engineering, K. N. Toosi University of Technology. He is also a member of the Industrial Control Center of Excellence and the Founder of the Intelligent Systems Laboratory (ISLab). His main research interests include intelligent systems and control, including artificial rough and deep neural networks, fuzzy systems, neural nets, optimization, and applications in the identification, prediction, classification, and control. He is the Head and the Co-Founder of the Intelligent Systems Scientific Society of Iran (ISSSI), and a member of the *International Journal of Information and Communication Technology Research* (IJICTR) Editorial Board.

AMIRREZA KHODADADIAN received the Ph.D. (Dr. rer. nat.) degree (Hons.) in applied mathematics from the University of Vienna, in 2017, and the postdoctoral degree from the Technical University of Vienna (TU Wien), in 2018. Afterward, he moved to Germany and works with the Institute of Applied Mathematics, Leibniz Universität Hannover. His research interests include scientific computing, using partial differential equations to model real-world problems, Bayesian inversion, and computational mechanics.

MARYAM PARVIZI received the Ph.D. (Dr. rer. nat.) degree from the Institute of Analysis and Scientific Computing, Technical University of Vienna, in 2021. Then, she started a Postdoctoral program with Leibniz Universität Hannover. In November 2021, she won the prestigious Alexander von Humboldt Fellowship. Since April 2022, she has been working on Alexander von Humboldt Fellowship project.

THOMAS WICK was born in Weidenau, Germany, in 1983. He received the B.Sc. and M.Sc. degrees in mathematics from the University of Siegen, Germany, in 2006 and 2008, respectively, and the Ph.D. degree in mathematics from Heidelberg University, Germany, in 2011. He is currently a Full Professor of scientific computing and the Director of the Institute of Applied Mathematics (IfAM), Leibniz Universität Hannover (LUH). Furthermore, he is a member of the Cluster of Excellence PhoenixD. His research interests include numerical methods for partial differential equations, multiphysics problems, adaptive finite elements, and computational mechanics.

NIMA NOII received the B.Sc. degree from the University of Portsmouth, the M.Sc. degree from the University of Nottingham, and the Ph.D. degree (Dr.-Ing) from the Technical University of Braunschweig. He has completed two postdoctoral studies with the Institute of Applied Mathematics and the Institute of Continuum Mechanics, Leibniz Universität Hannover.