9th International scientific conference
Technics and Informatics in Education – TIE 2022
16-18 September 2022

Session: *Engineering Education and Practice*
Original scientific paper
DOI: 10.46793/TIE22.165B

Getting Started with Wall Segmentation

Mihailo Bjekić [1]* and Ana Lazović [2]
[1] Everseen, Belgrade, Serbia
[2] Daon, Belgrade, Serbia
* mihailobjekic@gmail.com

Abstract: *In recent years, convolutional neural networks have been widely used in the area of semantic segmentation. In this paper, semantic segmentation network for detecting walls of indoor scenes is presented. Given an image of an indoor scene, the network automatically locates the wall regions in the image. In other words, walls are distinguished from the furniture, windows, curtains, and other possible indoor elements. Encoder-decoder structure of the semantic segmentation module is used. Specifically, PSPNet is used, one of the most common semantic segmentation algorithms. Model is trained on a new indoor scene dataset made from the publicly available ADE20K dataset, consisting of only two semantic labels: wall and no wall.*

Keywords: *semantic segmentation; indoor scenes; encoder-decoder; ADE20K; PSPNet.*

1. INTRODUCTION

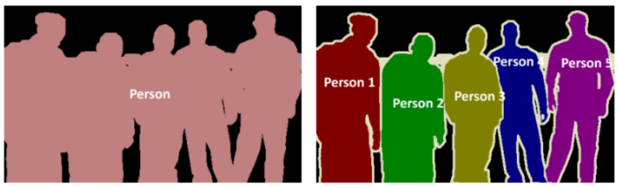
The rapid development of deep neural network architectures, availability of databases and increase of processing power have made it possible to solve more complex tasks in the field of computer vision. One such task is image segmentation [1]. Image segmentation is a process of classifying each pixel of an image to one of the predefined categories. Hence, image segmentation can be considered as a classification on pixel level. Contrary to classification, where the model identifies what is in an image, image segmentation model also performs localization. Image segmentation has two variants: semantic segmentation and instance segmentation [2].

Due to its capabilities, image segmentation can be used in different areas, such as autonomous driving [3], agriculture [4], robotic navigation [5], medical imaging [6], satellite imagery [7], scene understanding [8], etc. The main area of interest in this paper is indoor scene parsing.

Scene parsing is a process of segmenting and parsing an image into different image regions associated with semantic categories [9]. As it predicts class label, location, and shape of the object in an image, it provides complete understanding of a scene. Our goal is to develop a system for segmenting walls in images of indoor scenes. Indoor semantic segmentation is a challenging task due to the high variability of data. This variability is the result of indoor scenes often being cluttered, with a lot of illumination variation [10]. Also, there is often similarity between walls and other semantic parts, such as ceilings, that makes it more difficult to distinguish between these classes.

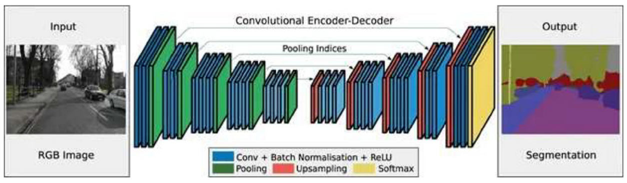
2. Semantic segmentation

Semantic segmentation is the process of assigning class label to every pixel in an image. It treats multiple objects of the same class as a single entity. Instance segmentation, the other type of image segmentation, treats multiple objects of the same class as distinct objects. In the Fig. 1 difference between semantic segmentation and instance segmentation is shown.



2.1 Architectures

The most commonly used architecture for semantic segmentation is symmetric. This architecture consists of an encoder and a decoder, followed by a pixel-wise classifier, as shown in the Fig. 2.



Typical semantic segmentation algorithms that use this structure are SegNet [12], U-Net [13], DeepLab [14], etc.

The encoder part of the architecture is typically a pre-trained classification network that is used for extracting complex semantic features. As preserving image dimensions throughout the entire network is computationally expensive, encoder performs downsampling of the input resolution. The output of the encoder structure is a low-resolution feature map that is learned to be efficient at discriminating between classes. Due to the downsampling of the input image, a lot of information is lost.

The decoder part of the architecture is a network whose main role is to recover details from the feature map. Input to the decoder is the output of the encoder. Decoder can also use additional feature maps from middle layers of the encoder using skip connections. This helps the decoder to prevent loss of information that is imposed by the encoder. The decoder upsamples encoded features to the resolution of the input image and outputs the segmentation mask.

2.2. Loss functions

The most widely used loss function for the classification task is a cross-entropy loss. Since semantic segmentation is pixel level classification, loss function that is often used, is pixel-wise cross-entropy loss [15]. This loss examines each pixel of an image individually, after which, an averaging over all pixels is done. This can be a problem if different classes are not equally represented in an image, because the most prevalent class will dominate during training. Hence, the cross-entropy loss is not a good choice in the case of imbalanced classes. One of the potential solutions is to use weighted cross-entropy loss, where each class is assigned with the appropriate weight. Larger weights are assigned to the less represented classes, which leads to the decrease in influence of the more represented classes.

Focal loss is an improved version of cross-entropy loss that makes the model focus on "difficult" examples by assigning them the larger weights. In the case of semantic segmentation, difficult examples are the pixels for which the model prediction (probability of belonging to the genuine class) is small, such as pixels of a background with noisy texture, pixels of partially cluttered objects, etc.

Another popular loss function, that successfully deals with the problem of imbalanced data in semantic segmentation, is dice loss. However, this loss only addresses the foreground-background imbalance, but ignores imbalance between "easy" and "difficult" examples. It is based on the dice coefficient that is a measure of overlap between two masks.

2.3. Metrics

The best-known metrics for evaluating semantic segmentation models are pixel accuracy (PA) and intersection over union (IoU).

Pixel accuracy is a ratio between the amount of correctly classified pixels and total number of pixels in the image. In the case of multiple classes, mean pixel accuracy (mPA), which represents the class average accuracy, is used. It is not recommended to use this metric in the case of imbalanced class datasets because only the correct classification of the dominant class will yield a high accuracy.

Intersection over union calculates the ratio between the overlap between the ground truth and the output segmentation mask, and their union. In the case of multiclass datasets, mean intersection over union (mIoU) is used. mIoU is calculated by averaging the IoU over all classes.

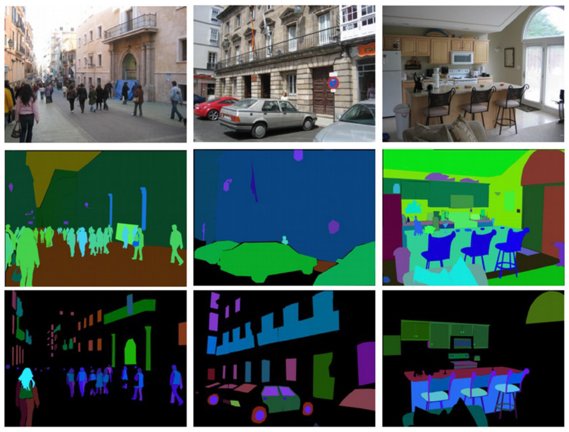
3. Wall segmentation

Wall segmentation is a special case of semantic segmentation. The task is to classify each pixel in one of two classes: wall and no wall. The goal is to distinguish walls from the rest of the room: ceilings, windows, paintings, doors, furniture, floors…

Wall segmentation is not an easy task. The wall edges are usually hard to detect due to the similarity with other semantic parts of the indoor scene. Also, often there are blurred parts of an image, representing items hanging on the wall that are difficult to localize, thus making it difficult to segment walls.

3.1. Dataset

In this paper, a modification of the ADE20K dataset is used [9]. The original ADE20K dataset consists of more than 20000 images of both indoor and outdoor scenes, annotated with 150 different categories. Each image has an associated segmentation mask. Most objects are also annotated with their parts. Examples of images from the ADE20K dataset with their associated segmentation masks are shown in the Fig. 3.



As the ADE20K dataset consists also of images that are not useful for the task of wall segmentation, it is modified so that it contains only images of interiors. Only a third of the original dataset are images of interest. Only three labels are kept: wall, no-wall and not annotated pixels. Examples of images from the modified dataset are shown in the Fig. 4.

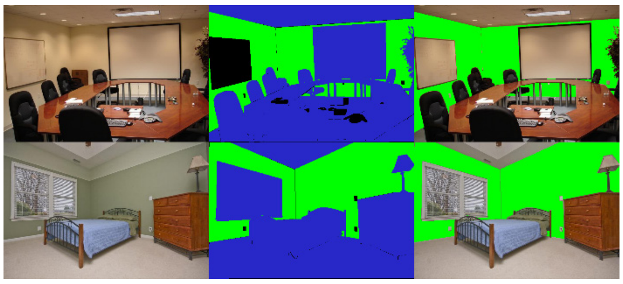


Figure 4. *Indoor images in the modified ADE20K dataset with associated segmentation masks (green - wall, blue - no wall, black - not annotated pixels)*

3.2. Model

In this paper an encoder-decoder semantic segmentation model based on PSPNet [16] is utilized. The encoder forms a feature map of low resolution from the given image, while the decoder upsamples the coarse feature map into a full-resolution map and produces the segmentation mask.

Encoder

Encoder is usually a modified convolutional neural network, typically used for classification tasks. In this paper ResNet-50 network is used. In [14], a variety of techniques for improving performance of the existing semantic segmentation architectures are proposed. These techniques reflect in obtaining finer results with less computational power. One of the improvements refers to the application of a dilated convolution, instead of a standard convolution within the encoder network.

Working with low-resolution feature maps leads to having less parameters of the model. Another advantage is having a large receptive field that enables extracting more context information. On the other hand, the main disadvantage of low-resolution feature map is the lack of spatial information that is very important for obtaining fine details for the task of semantic segmentation.

Dilated convolution enables having a large receptive field without increasing the number of parameters, while preserving spatial resolution. An example of the dilated convolution with a kernel size 3×3, with different dilation rates is given in Fig. 5.

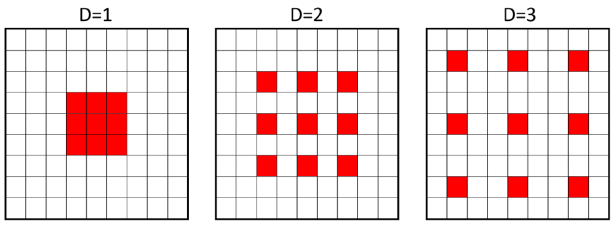


Figure 5. *Example of 3×3 dilated convolution with dilation rate D={1, 2, 3}*

In this paper, dilated ResNet-50 network is used. Following the work in [14], in the last two building blocks of the network, stride is reduced to 1 and all the following convolutions are replaced with dilated convolutions with a dilation rate D=2.

Decoder

The main part of the decoder is the pyramid pooling module (PPM) [16]. The entire structure of the used semantic segmentation model, with the PPM module, is shown in the Fig. 6.

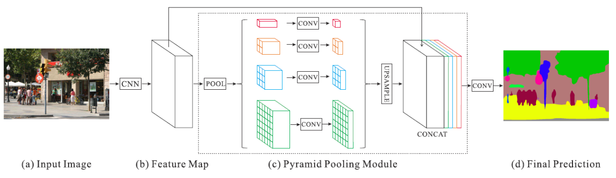


Figure 6. *Overview of the semantic segmentation model with PPM module [16]*

PPM gathers global context information by different region-based context aggregation. On top of the encoded feature map, adaptive pooling is applied, followed by 4-level pyramid pooling. Outputs of different levels of pyramid module are feature maps of different sizes. On top of each of these feature maps average pooling is used, followed by 1×1 convolution. The purpose of this convolution is to reduce the number of channels N times compared to the feature map produced by the encoder, where N is the number of pyramid levels. Obtained feature maps are upsampled to the size of the input feature map using bilinear interpolation. Finally, all four feature maps, along with the input feature map, are concatenated, thus obtaining a global feature map. It is followed by a convolutional layer in order to generate the final prediction map.

The number of pyramid levels, as well as the size of each level can be changed, according to the size of the feature map that is an input to the PPM. Using 4-level pyramid, the pooling filters cover the entire image, half of the image and the small regions of the image. This is a reason why information gathered by the PPM is more representative than information gathered by global average pooling. After the PPM module, the segmentation mask is upsampled to the resolution of the input image.

4. Experiments

The described semantic segmentation model was implemented in PyTorch [17].

The criterion function used for model training was sum of cross-entropy for each spatial position in the feature map. All pixels in an image, as well as the class labels (wall/no wall), have the same weight. Non-annotated pixels were ignored during training. Three different approaches to the model training were used.

4.1. Training

Optimization algorithm used for training is stochastic gradient descent (SGD). "Poly" learning rate strategy was used (1).

$$\alpha_{curr} = \alpha_{start} \cdot \left(1 - \frac{iter}{max\_iter}\right)^{0.9} \tag{1}$$

The starting learning rate was set to $\alpha_{start} = 0.02$, while the maximum number of iterations was set to $max\_iter = 100000$. Current iteration is given by $iter$. Number of epochs was 20 with 5000 iteration per epoch. The learning rate over iterations is shown in the Fig. 7.

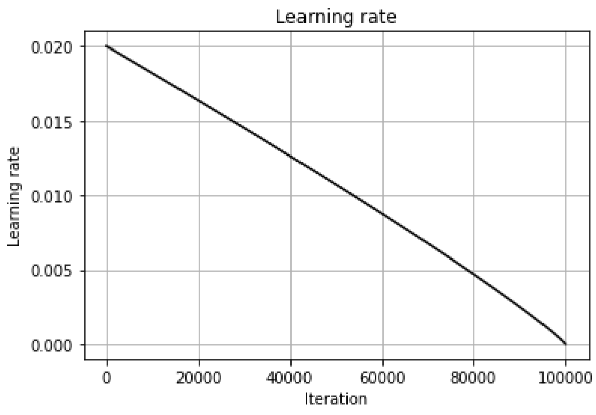


Figure 7. *Learning rate value during training*

For data augmentation, random mirror flip and random resize to the one of the pre-defined sizes were applied. As additional regularization, dropout with the parameter p=0.1 was performed before the last convolutional layer in the decoder part. Also, each batch consists of two images.

First approach to model training consists of two separate steps. Firstly, the model was trained on the entire ADE20K dataset (with all 150 classes), after which transfer learning, on the modified ADE20K dataset, was performed. In the first training, encoder was initialized with weights of the ResNet-50 model pre-trained on ImageNet, while the decoder was randomly initialized using Kaiming initialization. Transfer learning was performed by changing only the last output layer of the decoder (in order to enable classification into 2 classes, instead of 150), and training only this new layer, while freezing all previous. The model was trained for only one epoch after transferring the weights.

Second approach to model training, unlike the first approach, trained the entire decoder structure, not only the last layer, while the encoder weights were frozen. The changed model was trained for 5 epochs.

Third approach used the modified ADE20K dataset from the start. Unlike previous approaches, there was no transfer learning. After initializing the encoder with pre-trained ResNet-50 and random initialization, the model was trained end-to-end with two classes.

4.2. Results

Dataset used for model evaluations is a subset of the modified ADE20K validation dataset, consisting only of indoor images. Metrics used for model evaluation are pixel accuracy and intersection over union.

Evaluation results of models trained by the three different approaches are given in the Table 1.

Table 1. *Evaluation results on the validation set*

	First approach	Second approach	Third approach
PA [%]	84.82	86.24	90.75
IoU [%]	56.87	59.08	69.05

From the results given in Table 1, it can be seen that the best pixel accuracy and IoU are obtained by the third approach to model training, where model was specialized to classify only two classes from the start. It is important to note that high pixel accuracy doesn't always mean that the segmentation model performs good for each class, especially in the case of imbalanced class datasets. For that reason, IoU is the better metric.

Results of wall segmentation for all three approaches, with the corresponding pixel accuracy and IoU, for one image from the validation set are given below. In the Fig. 8 original image and ground truth are given, while in the Fig. 9 predicted segmentation masks are given.

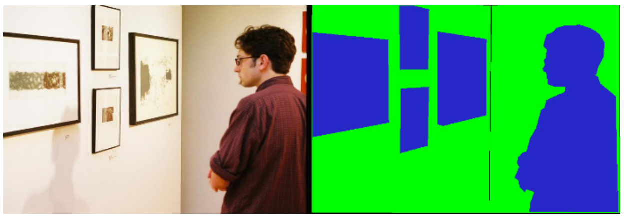


Figure 8. *Original image (left) and ground truth (right)*

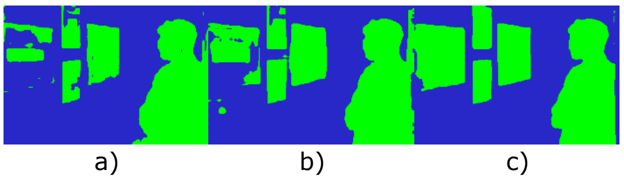


Figure 9. *Predicted segmentation masks: a) first approach, b) second approach, c) third approach*

Based on previous images, it can be seen that the first approach gives the worst results. A lot of pixels of paintings are classified as wall. There is an improvement using the second approach, but the third approach gives the best results.

In the Fig. 10, smoothed accuracy and smoothed loss of the best model on the train set during training, at each iteration, are shown.

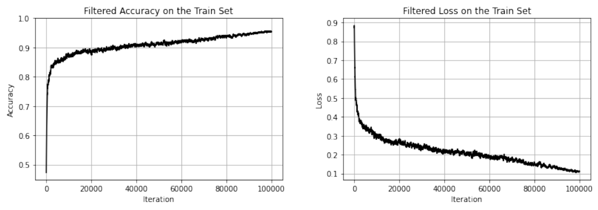


Figure 10. *Accuracy (left) and loss (right) of the best model on the train set during training*

In the Fig. 11, pixel accuracy and IoU on the validation set, for each epoch during training, are shown.

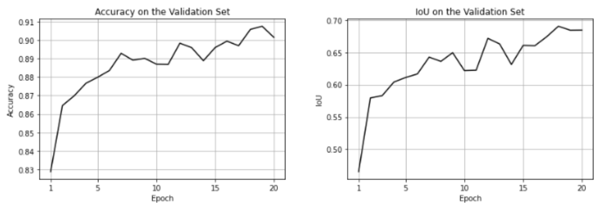


Figure 11. *Accuracy (left) and IoU (right) on the validation set during training*

5. Limits of the current approach

During model testing, it has been observed that there are different limitations imposed mostly by data quality. Some limitations are discussed in more detail below.

5.1. ADE20K scenes

All data in the ADE20K dataset is grouped into different scene categories, such as living room, bedroom, church, airport, etc. When creating a modified ADE20K dataset used for training, described in this paper, a subset of scene categories was selected. This selection was done under the assumption that images belonging to a certain category contain walls. There was no validation whether the selected images contain walls or not. As a result, there is a number of images in the final dataset that are not of interest for training the wall segmentation model. This may result in model performance degradation.

5.2. Annotation quality

During error analysis, it has been noticed that there are certain images with either wrongly annotated walls, or pixels of wall regions not annotated at all. Examples of these two cases are given in the Fig. 12.

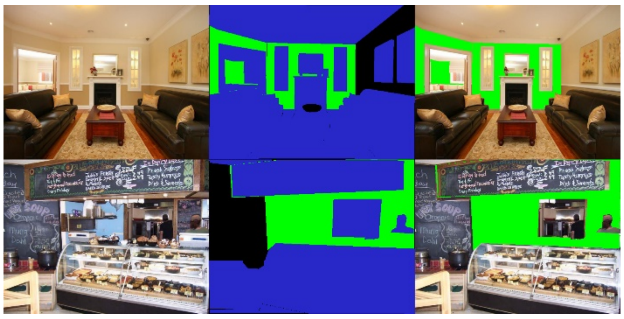


Figure 12. *Examples of low-quality annotations*

5.3. Overcluttered images

Another dataset related problem that may affect model quality is when scene in the image is cluttered with various object, as shown in Fig. 13.

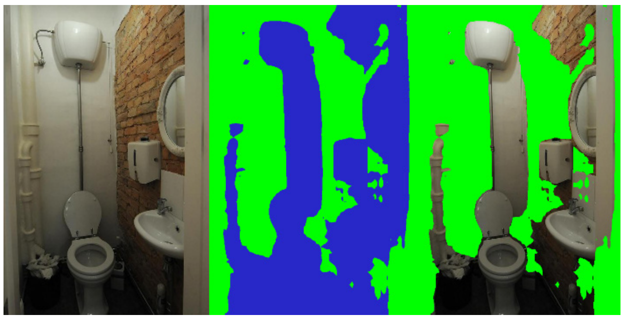


Figure 13. *Example of a cluttered scene and the model prediction*

5.4. Image resolution

The model is trained on the range of different resolutions and gives the best results for images of similar resolution. For images of substantially different resolution, the model does not behave as expected. When the input resolution is large, image should be downsampled to a lower resolution within the range the model was trained on. On the other hand, if the input resolution is too small, the model is not able to extract all the information, from the image, necessary for segmentation.

5.5. Difficult images

When it comes to semantic segmentation, human error performance is a good proxy for the bayes error [18]. So, if humans are not able to successfully distinguish between wall and no wall classes in an image, it cannot be expected from the model to perform well on this image. Example of such an image is given in the Fig. 14.

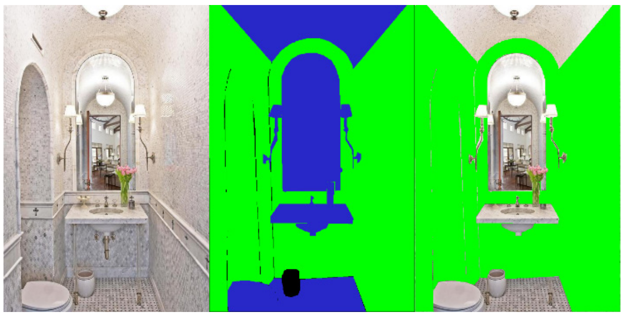


Figure 14. *Example of an ambiguity*

6. CONCLUSION

In this paper, a model structure for semantic segmentation of walls, was described. Encoder-decoder architecture was used. As the encoder, dilated ResNet-50 network was used. Building block of the decoder was pyramid pooling module in combination with bilinear interpolation. The model was trained on a modified ADE20K dataset, consisting only of interior scene images with two classes (wall and no wall). Three different approaches to model training were tested. The best approach was directly training the model on the modified ADE20K dataset, without transfer learning. Implementation of all approaches is provided in [17].

Wall segmentation is a complex task, due to strong occlusions, similarity with other semantic parts of the interior scenes, as well as different objects that occlude the wall and are hard to localize. During model development, different problems with the current setup of the project, were observed. In the future work, most of these problems can be overcome. When it comes to the selected images, validation of each image, whether it is an image of interest and contains walls, should be performed. Also, all images with bad mask annotations should be discarded. Regarding images with any ambiguities, these images should be treated carefully. All ambiguous images reflect on the model performance and their influence cannot be predicted. Each image should be separately reviewed whether to discard or keep.

Except data cleaning, future work may also consist of experimenting with different model architectures in order to increase validation metrics. Also, lighter models can be implemented with the goal to speed up the entire wall segmentation system. In future work, practical application of such a system can also be explored.

REFERENCES

[1] S. Minaee, Y. Boykov, F. Porikli, A. Plaza, N. Kehtarnavaz, D. Terzopoulos, "Image Segmentation Using Deep Learning: A Survey", *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, pp. 3523-3542, Jul. 2022.

[2] W. Gu, S. Bai and L. Kong, "A review on 2D instance segmentation based on deep neural networks", *Image Vis. Comput.*, vol. 120, Apr. 2022.

[3] L. Tran and M. Le, "Robust U-Net-based Road Lane Markings Detection for Autonomous Driving", in Int. Conf. System Science Eng., Jul. 2019, pp. 62-66.

[4] K. Singh, R. Rawat, and A. Ashu, "Image Segmentation in Agriculture Crop and Weed Detection Using Image Processing and Deep Learning Techniques", *Int. J. Res. Eng. Science Manage.*, vol. 4, no. 5, pp. 235–238, Jun. 2021.

[5] V. Koval, D. Zahorodnia and O. Adamiv, "An Image Segmentation Method for Obstacle Detection in a Mobile Robot Environment" in 9th Int. Conf. Adv. Comput. Information Techologies, Jun. 2019, pp. 475-478.

[6] N. Siddique, S. Paheding, C. P. Elkin and V. Devabhaktuni, "U-Net and Its Variants for Medical Image Segmentation: A Review of Theory and Applications", *IEEE Access*, vol. 9, pp. 82031-82057, Jun. 2021.

[7] B. Neupane, T. Horanont, and J. Aryal, "Deep Learning-Based Semantic Segmentation of Urban Features in Satellite Images: A Review and Meta-Analysis", *Remote Sensing*, vol. 13, no. 4, p. 808, Feb. 2021, Art. no. 808.

[8] S. Barchid, J. Mennesson, C. Djeraba, "Review of Indoor RGB-D Semantic Segmentation Convolutional Neural Network", in Int. Conf. Content-Based Indexing, Jun. 2021.

[9] B. Zhou, X. Puig, S. Fidler, A. Barriuso and A. Torralba, "Scene Parsing through ADE20K Dataset", in IEEE Conf. Comput. Vis. Pattern Recognit., Jul. 2017.

[10]T. Liu, Y. Wei, Y. Zhao, S. Liu and S. Wei, "Magic-Wall: Visualizing Room Decoration by Enhanced Wall Segmentation", *IEEE Trans. on Image Process.*, vol. 28, no. 9, pp. 4219-4232, Sept. 2019.

[11]P. Sharma. "Computer Vision Tutorial: A Step-by-Step Introduction to Image Segmentation Techniques (Part 1)." Analytics Vidhya. https://www.analyticsvidhya.com/blog/2019/04/introduction-image-segmentation-techniques-python (accessed Jul. 23, 2022).

[12]V. Badrianarayanan, A. Kendall and R. Cipolla, "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation", *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, pp. 2481–2495, Jan. 2017.

[13]O. Ronneberger, P. Fischer and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation", in Int. Conf. Med. Image Comput. Computer-Assisted Intervention, pp. 234–241, May 2015.

[14]L. C. Chen, G. Papandreou, I. Kokkinos, K. Murphyand A. L. Yuille, "DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs", *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, pp. 834–848, Apr. 2017.

[15]S. Jadon, "A survey of loss functions for semantic segmentation," in IEEE Conf. Comput. Intell. Bioinf. Comput. Biol., Oct. 2020, pp. 1-7.

[16]H. Zhao, J. Shi, X. Qi, X. Wang and J. Jia, "Pyramid Scene Parsing Network," in IEEE Conf. Comput. Vis. Pattern Recognit., 2017, pp. 6230-6239.

[17]M. Bjekić and A. Lazović. "Wall Segmentation". GitHub.com. https://github.com/bjekic/WallSegmentation (accessed Jul. 24, 2022).

[18]M. Mason. "Understanding Bayes Error: How a low cost machine learning strategy could have a big impact". LinkedIn.com. https://www.linkedin.com/pulse/understanding-bayes-error-how-low-cost-machine-learning-malcolm-mason (accessed Jul. 23, 2022).