A General Two-branch Decoder Architecture for Improving Encoder-decoder Image Segmentation Models

Sijie Hu, Fabien Bonardi, Samia Bouchafa and Désiré Sidibé
University Paris-Saclay, Univ. Evry, IBISC, 91020, Evry, France

Keywords: Multi-branch, Encoder-decoder, Complementary Learning, Supervised Learning, Semantic Segmentation.

Abstract: Recently, many methods with complex structures were proposed to address image parsing tasks such as image segmentation. These well-designed structures are hardly to be used flexibly and require a heavy footprint. This paper focuses on a popular semantic segmentation framework known as encoder-decoder, and points out a phenomenon that existing decoders do not fully integrate the information extracted by the encoder. To alleviate this issue, we propose a more general two-branch paradigm, composed of a main branch and an auxiliary branch, without increasing the number of parameters, and a boundary enhanced loss computation strategy to make two-branch decoders learn complementary information adaptively instead of explicitly indicating the specific learning element. In addition, one branch learn pixels that are difficult to resolve in another branch making a competition between them, which promotes the model to learn more efficiently. We evaluate our approach on two challenging image segmentation datasets and show its superior performance in different baseline models. We also perform an ablation study to tease apart the effects of different settings. Finally, we show our two-branch paradigm can achieve satisfactory results when remove the auxiliary branch in the inference stage, so that it can be applied to low-resource systems.

1 INTRODUCTION

Semantic segmentation can be formulated as the task of labeling all pixels in an image with semantic classes. Most state-of-the-art semantic segmentation models are based on the encoder-decoder architecture or its variants. Specifically, the encoder extract information from the original input, and the decoder integrate previously extracted information and recover semantic information from it. In recent years, researchers commit to exploring different network architecture (Simonyan and Zisserman, 2014; He et al., 2016) to learn a more general representation, then deployed to the image segmentation task (Chen et al., 2018; Wang et al., 2020).However, a general representation extracted by the encoder means that the decoder need to decrease the gap between task free representation and task dependency information.

In order to improve the parsing ability of decoder, DeeplabV3+ (Chen et al., 2018) through pyramid pooling integrate the contextual information at multiple scales. FCN (Long et al., 2015) use skip-connection to fuse feature maps of different layers. (Li et al., 2019; Li et al., 2018) try to explore the interrelationships between features through attention mechanisms. It is worth noting that some recent works start exploring the two-branch structure in the decoder (Fu et al., 2019; Yuan et al., 2020). They capture meaningful information by carefully designing different branches. Unfortunately, existing two-branch structures were elaborately designed, thus hard to port to other types of decoders, and the

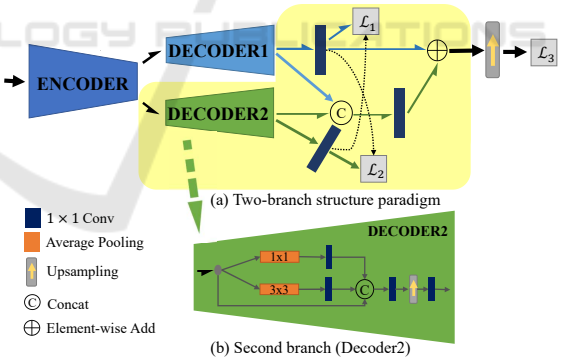


Figure 1: Overview of our proposed two-branch architecture. The output of the encoder is divided into two groups, which are represented by two 'half arrows'. Then each group is input to each branch separately and followed by a residual-liked module to fuse the outputs of two branches.

Hu, S., Bonardi, F., Bouchafa, S. and Sidibé, D.
A General Two-branch Decoder Architecture for Improving Encoder-decoder Image Segmentation Models.
DOI: 10.5220/0010818800003124
In *Proceedings of the 17th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2022) - Volume 5: VISAPP*, pages 374-381
ISBN: 978-989-758-555-5; ISSN: 2184-4321

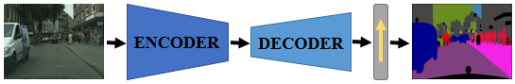


Figure 2: Encoder-Decoder paradigm.

degradation of model performance caused by removing a branch is also unacceptable. Or they were just designed for post-processing and are challenging to train end-to-end. On the other hand, with the continuous improvement of the encoder's representation ability, making full use of the information extracted by the encoder is still an open question. Therefore, we have reason to suspect that the existing encoder-decoder-based models do not fully integrate the information extracted by the encoder. We verified this view through experiments.

To alleviate these problems, we propose a more general two-branch paradigm, composed of a main branch and an auxiliary branch for improving the structure of the decoder. At the same time, we design a simple yet efficient branch that can be flexibly integrated into existing encoder-decoder semantic segmentation systems to verify the effectiveness of the proposed two-branch structure. In order to enable two branches to learn complementary information, we customize a loss calculation method to supervise the learning process of each branch. With these ideas, different branches can learn complementary information adaptively instead of explicitly indicating the specific learning elements of different branches. In addition, learning complementary information can make the two branches compete with each other to a certain extent during the learning process, which can further improve performance. Moreover, compared with the counterpart of the original model, the ameliorated two-branch version reduces or maintains the number of parameters while improving performance.

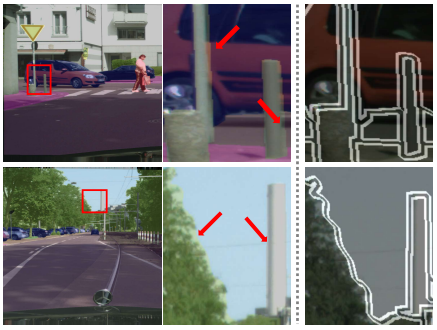
Our main contributions can be summarized as follows:

- We propose a general two-branch paradigm to enhance the capability of the decoder to parse the information extracted by the encoder without increasing the number of parameters.
- We propose the BECLoss that can supervise two-branch decoders to learn complementary information adaptively instead of explicitly indicating the specific learning elements to each branch.
- We design a simple yet efficient branch that can be flexibly integrated into the existing encoder-decoder framework to form a two-branch structure.

2 RELATED WORK

Encoder-decoder and Variants. As a general structural paradigm, encoder-decoder is widely used in the field of image segmentation. Such a structure usually first encode features from the input to a latent feature space, then gradually recover the information in the decoder. U-Net (Ronneberger et al., 2015) explored the potential relationship between the features of the encoding phase and their counterpart in the decoding phase through multiple skip-connections. SEMEDA (Chen et al., 2020) first learned to convert the label to an embedding space under the guidance of the boundary information, and then supervised the encoder-decoder structure under the learned subspace. PSPNet (Zhao et al., 2017) and Deeplab family (Chen et al., 2018; Chen et al., 2017) introduced dilated convolution in encoder for increasing the receptive field while maintaining the resolution, then several parallel pyramid pooling were followed to integrate information at different scales. Inspired by (Hu et al., 2018; Woo et al., 2018), attention mechanism and its variants are adopted in encoders or decoders (Li et al., 2019; Zhong et al., 2020) to improve performance. In (Li et al., 2018), attention was deployed in the decoding stage for re-calibrating the feature maps with learnable weights. In addition, the application of self-attention (Vaswani et al., 2017) in encoder has gradually become popular due to its capability of encoding distant dependencies for better feature extraction. SETR (Zheng et al., 2020) adapted a pure transformer encoder to extract features from an image seen as a sequence of patches then followed a decoder to restore the semantic information.

Multi-branch. Learning different information through multiple parallel data streams has been proved to have more advantages for representation and generalization. Specifically, HRNet (Wang et al., 2020) repeatedly exchanged the information across different resolutions by a series of parallel feature extraction streams in the encoding process to maintain high-resolution representations. Based on HRNet, (Tao et al., 2020) proposed a hierarchical multi-scale attention approach in which each data stream learned a specific image scale so that the model can consider the information of multiple input image scales when predicting. GSCNN (Takikawa et al., 2019) designed a two-stream structure, one for context information extraction, another one for boundary-related information extraction. Combined with attention, RAN (Huang et al., 2017) proposed a three-branch structure that performs the forward and backward attention learning processes simultaneously. Similarly, DANet (Fu et al., 2019) used a two-branch encoder to learn



(a) (b)

Figure 3: (a) Mis-labeled boundary pixels and (b) Extracted inner boundary.

the semantic relevance in spatial and channel feature spaces respectively. Unlike above works, SegFix (Yuan et al., 2020) proposed a post-processing scheme that predicted boundary and direction maps employing a two-branch decoder supervised by two boundary-related losses.

Encouraged by multi-branch learning, we propose a more general and easy-to-deploy two-branch paradigm, in which a new branch can be easily inserted into the original decoder to form a two-branch decoder and, as a result, improve the discriminating ability. Unlike previous works, we design a general paradigm and enable different branches to learn complementary information adaptively instead of explicitly indicating the specific learning elements of different branches.

3 METHODOLOGY

In this section, we first systematically describe the two-branch decoder paradigm, then design a simple yet efficient branch that can be applied as a plug-in to existing encoder-decoder frameworks to turn them into our proposed two-branch architecture. Finally, we introduced a new loss calculation method that can be used to supervise branch learning complementary information.

3.1 Two-branch Structure Prototype

In an image segmentation model, existing encoder-decoder architectures can be simply represented in Figure 2. Our proposed encoder-decoder based two-branch variant is depicted in Figure 1. As shown in Figure 1 (a), raw data is first input into the encoder for feature extraction, then encoded features are input to two branches separately, followed by a residual-liked module to integrate information from different

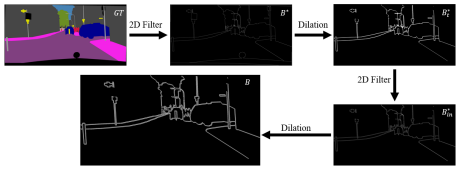


Figure 4: Ground-truth inner boundary extraction process.

branches adaptively. For the fusion of two branch features, we use the output of the penultimate layer of each decoder instead of the last layer to retain more information. Specifically, in the residual path, we first concatenate the output features of two branches, next follow a 1×1 convolution to reduce the channels. Then features are combined with the output of the first branch by an element-wise addition operation. The final output is up-sampled to recover resolution if needed.

3.2 Additional Branch Setting

In this part, we design a simple branch that can be deployed into an encoder-decoder framework to form a two-branch decoder architecture. As shown in Figure 1 (b), the branch takes the encoded features as input. Similarly to (Zhao et al., 2017), we utilize a parallel average pooling module, each path consisting of an average pooling operator and a 1×1 convolution operator. We concatenate the output of each path to get a multi-scale feature representation and followed by another 1×1 convolution. Then, we get the output of this branch through an up-sampling operation and a 1×1 convolution operation. Finally, we divide the encoded features into two groups along the channel axis, and each grouped feature is entered into a specific branch.

3.3 BECLoss

In supervised learning, loss function plays a crucial role in the optimization of the network. Thus, we further propose a novel loss computation strategy that can efficiently optimize this two-branch structure. Moreover, (Chen et al., 2020; Takikawa et al., 2019) have proved that introducing boundary information in the loss helps to improve the inherent sensitivity of the network to boundary pixels. Thus, we introduce boundary information in the proposed loss to help the model learn boundary features during the training stage, which is verified in ablation experiments.

We name this well-designed loss BECLoss. Specifically, BECLoss takes three inputs: outputs of the first branch X^1 and the second branch X^2 and ground-truth map GT. We assume batch size as 1, thus the shape of $X^k(k = 1,2)$ is $C \times H \times W$ and C, H

and W indicate the number of predicted classes, high and width of input images, respectively. First, we get the probability distribution $S^k \in \mathbb{R}^{H \cdot W \times C}$ which can be computed as:

$$S_i^k = \frac{exp(X_i^k)}{\sum_j^C exp(X_i^k[j])} \tag{1}$$

where $i = 0 \ldots H \times W - 1$ denotes the index of pixels, $j = 0 \ldots C - 1$ denotes the index of channels. Then, we compute the probability map of ground truth label $P^k \in \mathbb{R}^{H \cdot W \times 1}$ as:

$$p_i^k = S_i^k[gt_i] \tag{2}$$

where gt_i is the i^{th} pixel in GT. Following, we define a mask M^1 for indicating all the pixels whose probability in P^1 is less than a threshold τ. M^1 indicates the pixels that are difficult to predict in the first branch. With the computed M^1 and P^2, we filter out all pixels in X^2 whose probability is less than a threshold τ:

$$M_i^1 = \begin{cases} 1 & \text{if } P_i^1 < \tau \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

where $i = 0 \ldots H \times W - 1$ denotes the index of pixels.

In order to standardize the loss definition, we use $\mathcal{L}^1$ to indicate the boundary enhanced loss computed from X^1, and $\mathcal{L}^2$ to indicate a partial loss that we get from X^2. In $\mathcal{L}^1$ and $\mathcal{L}^2$ we only consider the pixels which are hard to predict in the first branch in order to utilize the additional branch to assist in the prediction of these pixels. In addition, we use a hyperparameter γ to control the influence of boundary information $B \in \mathbb{R}^{W \times H}$ (detailed in 3.4) to the loss of the first branch, we get $\mathcal{L}^1 \in \mathbb{R}^{H \cdot W \times 1}$:

$$\mathcal{L}_i^1 = -log(P_i^1) \times (1 + \gamma \cdot B_i) \times M_i^1 \tag{4}$$

where $i = 0 \ldots H \times W - 1$ denotes the index of pixels. Following, we compute the partial loss $\mathcal{L}^2 \in \mathbb{R}^{H \cdot W \times 1}$:

$$\mathcal{L}_i^2 = -\log(P_i^2) \times M_i^1 \tag{5}$$

Finally, the BECLoss can be written as a weighted average sum of $\mathcal{L}^1$ and $\mathcal{L}^2$:

$$\mathcal{L}_{BEC} = \frac{\sum_i (\mathcal{L}_i^1 + \eta \cdot \mathcal{L}_i^2)}{\sum M_i^1} \tag{6}$$

where η is a hyperparameter used to control the ratio of $\mathcal{L}^2$ in $\mathcal{L}_{BEC}$.

The two branches can automatically learn complementary information which helps the proposed model to further learn a more appropriate way to combine the outputs of the two branches.

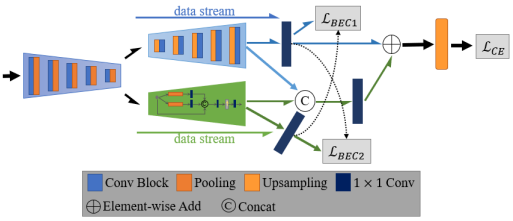


Figure 5: Architecture of modified SegNet with two decoders (SegNetT).

3.4 Ground-truth Boundary

In this part, we explain how we get a ground-truth boundary map from a ground-truth label map. Introducing approximate boundary information in the loss can improve the model's sensitivity to physical boundaries, which improves the prediction accuracy in the boundary area. However, there are always labeled error pixels in the hand-labeled ground truth map, which are especially obvious at the boundary region, as shown in Figure 3(a). In order to alleviate this problem, Figure 4 illustrates the inner boundary extraction process. Concretely, we first extract the boundary map B^* from the original ground-truth label map by a filter f that sets all pixels that do not have 8 identically-labeled neighbor pixels as 1, and other pixels as 0. Then we thicken the boundary by a 7×7 dilation operator and get boundary map B_t^*. Finally, we get the inner boundary B_{in}^* by applying the same filter f on B_t^* again and followed by another 3×3 dilation operator, as shown in Figure 3(b).

3.5 Joint Loss

The proposed BECLoss is designed for optimizing the network with two branches. The purpose is to guide the two branches to learn complementary information. It can naturally be combined with other losses for training the whole network. Therefore, the network is trained to minimize a joint loss function:

$$\mathcal{L} = \mathcal{L}_{CE} + \alpha \cdot \mathcal{L}_{BEC1} + \beta \cdot \mathcal{L}_{BEC2} \tag{7}$$

Specifically, $\mathcal{L}_{CE}$ is cross-entropy loss, $\mathcal{L}_{BEC1}$ and $\mathcal{L}_{BEC2}$ are proposed BECLoss for first and second branch, respectively. α and β are weights parameters of the two BECLoss.

4 EXPERIMENTAL RESULTS

In this section, we conduct experiments on Cityscapes dataset (Cordts et al., 2016) and Freiburg Forest dataset (Valada et al., 2016). In the following, we first

Table 1: Comparison in terms of IoU vs different baselines on the cityscapes val set with 11 semantic class labels.

Methods	sky	building	road	sidewalk	fence	vegetation	pole	vehicle	traffic sign	person	bicycle
SegNet	91.83	88.47	95.52	72.76	40.02	91.22	52.95	89.45	65.57	77.2	68.98
SegNetT (ours)	**93.35 (+1.52)**	**90.89 (+2.42)**	**96.65 (+1.13)**	**77.28 (+4.52)**	**49.72 (+9.7)**	**92.37 (+1.15)**	**61.54 (+8.59)**	**92.86 (+3.41)**	**75.64 (+10.07)**	**81.61 (+4.41)**	**75.06 (+6.08)**
DeepLabv3+	93.99	90.9	97.29	80.47	54.73	91.92	56.56	93.05	71.78	78.9	73.79
DeepLabv3+T (our	93.95	**91.99 (+1.09)**	**97.62 (+0.33)**	**82.31 (+1.84)**	**54.85 (+0.12)**	**92.55 (+0.63)**	**62.69 (+6.13)**	**94.17 (+1.12)**	**77.87 (+6.09)**	**82.4 (+3.5)**	**76.59 (+2.8)**
HRNet	94.31	92.06	97.67	82.31	54.94	92.59	63.31	94.34	76.37	82.27	75.4
HRNet-T (ours)	**94.83 (+0.52)**	**92.68 (+0.62)**	**97.98 (+0.31)**	**84.3 (+1.99)**	**56.28 (+1.34)**	**93.06 (+0.47)**	**67.17 (+3.86)**	**94.83 (+0.49)**	**79.98 (+3.61)**	**84.35 (+2.08)**	**77.09 (+1.69)**

Table 2: Improvements with two-branch decoder on Freiburg Forest val set.

Methods	BaseNet	Trail	Grass	Veg.	Sky	Obst.	Mean IoU (%)	Parms. (M)
SegNet	Vgg16	84.15	85.55	88.97	91.28	0	69.99	29.4
SegNetT (ours)		**88.55 (+4.4)**	**88.96 (+3.41)**	**0.91 (+1.94)**	**2.63 (+1.35)**	**47.93 (+47.93)**	**81.79 (+11.8)**	18.6
DeepLabv3+	Res50	83.03	86.11	89.96	92.16	36.1	77.48	26.6
DeepLabv3+T (ours)		**88.02 (+4.99)**	**88.93 (+2.82)**	**91.02 (1.06)**	**2.83 (+0.67)**	**52.87 (+16.77)**	**82.73 (+5.25)**	27.5
HRNet	Hrnet-W18	84.79	86.49	89.79	91.96	38.44	78.29	9.6
HRNet-T (ours)		**88.74 (+3.95)**	**89.35 (+2.86)**	**91.14 (+1.35)**	**92.6 (+0.64)**	**53.17 (+14.73)**	**83 (+4.71)**	9.6

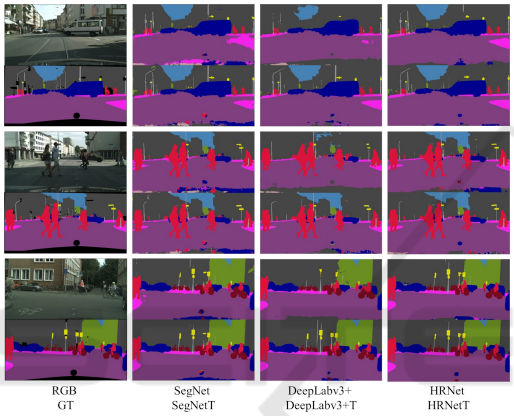


Figure 6: Qualitative results on the Cityscapes val set with 11 semantic class labels.

modify some classic image semantic segmentation algorithms to build their two-branch decoder counterpart, then compare the proposed two-branch architecture with the original network. Finally, we carry out a series of ablation experiments on Freiburg Forest dataset. Our models are trained on one Nvidia Tesla P100 GPU with mixed precision settings.

4.1 Datasets

Cityscapes. The Cityscapes dataset is a large-scale database for urban street scene parsing. It contains 5000 finely annotated images captured from 50 cities with 19 semantic object categories, in which 2875 images are used for training, 500 and 1525 images are used for validation and testing separately. All images are provided with a resolution of 2048×1024. We followed (Valada et al., 2019) and report results on the reduced 11 class label set.

Freiburg Forest. The Freiburg Forest dataset is an unstructured forested environments dataset. It contains 6 segmentation classes, i.e., sky, trail, grass, vegetation, obstacle, and void. The dataset contains 325 images with pixel level hand-annotated ground truth map. We follow (Valada et al., 2019) and use the same train and test splits provided by the dataset.

4.2 Implementation Details

In order to comprehensively test, we deploy proposed two-branch decoder on three classic baseline networks, namely, SegNet (Badrinarayanan et al., 2017), DeeplabV3+ (Chen et al., 2018), and HRNet (Wang et al., 2020). Two-branch SegNet is shown in Figure 5. We divide the output of the encoder into two groups, one of which is input to the original data stream, and another is input to the additional data stream. In our two-branch implementation, we denote the upper branch in the decoder as the original data stream, the lower branch as the additional data stream. Next, we follow the residual-liked module to fuse the two outputs while deploying the BECLoss and cross-entropy loss during the training. More concretely, we supervise the learning process of the two branches through $\mathcal{L}_{BEC1}$ and $\mathcal{L}_{BEC2}$, and the combination of two outputs are guided by $\mathcal{L}_{CE}$. We follow the same way to implement the counterpart of DeeplabV3+ and HRNet. Note that we only take the backbone in the original model as an encoder, and the rest as the decoder. In practice, we use Resnet50, Vgg16 and HRNet-W18 as backbones.

We initialize encoder with the weights pre-trained on ImageNet, this is totally the same as its original implementations (Badrinarayanan et al., 2017; Chen et al., 2018; Wang et al., 2020). We employ a cyclical exponent learning rate policy (Smith, 2017) where the min_lr and max_lr are set to $1e-5$ and $1e-2$, and cycle_length and step_size are set to 40 and 5 epochs respectively . Momentum and weight decay coefficients are set to 0.9 and 0.0005. If not specified, all models are trained with a mini batch size of 8. Fur-

Table 3: Improvements with two-branch decoder on Cityscapes val set with 11 semantic class labels.

Methods	BaseNet	Mean IoU (%)	Parms. (M)
SegNet	Vgg16	75.82	29.4
SegNetT (ours)		80.64 (+**4.82**)	18.6
DeepLabv3+	Res50	80.31	26.6
DeepLabv3+T (ours)		82.45 (+**2.14**)	27.5
HRNet-W18	Hrnet-W18	82.34	9.6
HRNet-W18T (ours)		83.9 (+**1.56**)	9.6

thermore, we configure the hyperparameter γ and η in BECLoss as 10.0 and 0.3. The scale α and β in Equation 7 are set to 2.0. For Cityscapes dataset, we set input image size to 384×768, thus random cropping (cropsize 384×768) is applied during training, and during testing, we use the original resolution of 1024×2048. For Freiburg Forest dataset, we resize the image to 384×768 during training and testing. All training images are augmented by random left-right flipping. We set 160 and 120 training epochs to Cityscapes datasets and Freiburg Forest dataset. In addition, as we compare the original models with their two-branch encoder counterpart, so we perform the same settings for each comparison pair to ensure fairness.

4.3 Experimental Evaluation

In this section, we provide an extensive evaluation of each component of our framework on two challenging outdoor datasets, namely Cityscapes dataset and Freiburg Forest dataset. We use the widely used intersection over union (IoU) to evaluate the performance of our approach.

4.3.1 Results on Cityscapes Dataset

Table 3 summarizes the results of our two-branch decoder with different baselines. We can see that our approach significantly improves the mean IoU. Specifically, our approach improves the mean IoU of original encoder-decoder frameworks, namely SegNet, Deeplabv3+, and HRNet, by 4.81, 2.14, and 1.56, respectively. In particular, our two-branch implementation of SegNet (SegNetT) dramatically reduces the number of parameters while significantly improving the performance. DeepLabv3+T and HRNet only slightly increase the parameters (0.9M) or keep the number of parameters while improving the model's performance. Our results also reflect that the original decoder does not fully use the information extracted by the encoder. In addition, table 1 illustrates the category-wise comparison between various baselines and their two-branch variants. We surprisingly find that our method has a significant improvement in the prediction accuracy of small-scale targets, like "pole", "traffic sign" and "person". Several segmentation results are shown in Figure 6, we can see that our two-branch variants perform better on those small-size-object classes in the images than the baseline models. Note that we may find the optimal hyperparameters to achieve better performances through grid search, but this is not the focus of this work.

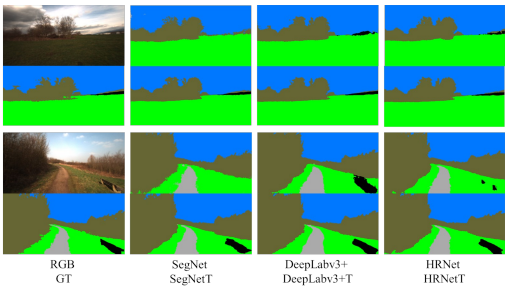


Figure 7: Qualitative results on the Freiburg Forest test set.

4.3.2 Results on Freiburg Forest Dataset

We carry out experiments on the Freiburg Forest dataset to further evaluate the effectiveness of our method. Quantitative results of Freiburg Forest are shown in Table 2. The baselines (SegNet, DeepLabv3+, HRNet) yield mean IoU 69.99%, 77.48%, and 78.29%. Our two-branch counterpart boosts the performance to 81.79%, 82.73%, and 83%. We can see that our methods outperform their baselines with notable advantage, especially for the class of "obstacle", which is hardest to segment because of its severe class imbalance. Several examples are shown in Figure. 7.

4.4 Ablation Study

4.4.1 BECLoss and Boundary

All two-branch variants are implemented by replacing the decoder of the original network with our proposed two-branch decoder, and through our well-designed BECLoss to explicitly supervise the learning process of the model, the two branches can learn complementary information. In addition, we introduce boundary information into BECLoss to improve the inherent sensitivity of our models to boundary pixels. To verify the validity of our method, we conduct a group of ablations to analyze the influence of various factors within our method. We report the results over the segmentation baseline SegNet on Cityscapes and Freiburg Forest dataset in Table 4.

As shown in Table 4, two-branch decoder improves the performance remarkably. Compared with

Table 4: Ablation study on Cityscapes val set and Freiburg Forest test set. *Loss*1-*Loss*3 represent deployed loss in Figure 1, *B* indicates BECLoss enhanced by boundary information.

Methods	Loss1	Loss2	Loss3	B	Mean IoU (%)	
					Cityscapes	Freiburg
SegNet	\	\	CE	\	75.82	69.99
SegNetT	CE	CE	CE	\	78.54 (+2.72)	78.9 (+8.91)
SegNetT	BEC	CE	CE	N	79.54 (+3.72)	80.48 (+10.49)
SegNetT	CE	BEC	CE	N	79.07 (+3.25)	79.9 (+9.91)
SegNetT	BEC	BEC	CE	N	79.5 (+3.68)	81.43 (+11.44)
SegNetT	BEC	BEC	CE	Y	80.64 (+**4.82**)	81.79 (+**11.8**)

the baseline SegNet, employing two-branch decoder yields a result of 78.54% mean IoU on Cityscapes dataset and 78.9% mean IoU on Freiburg Forest dataset, which brings 2.72% and 8.91% improvement. In addition, when we gradually replaced the cross-entropy loss CELoss of loss1 and loss2 with the BECLoss we designed, the performance further improved to 79.5% and 81.43%. Furthermore, we notice that when we use only one BECLoss, the result very slightly exceeds the result of using two BECLoss, as shown in the third row and the fifth row, the result from 79.54% goes to 79.5% on Cityscapes dataset. After introducing boundary information to BECLoss, performance further increased to 80.64%. Results show that our proposed two-branch decoder and boundary enhanced BECLoss bring great benefit to scene parsing.

4.4.2 Single Branch

As mentioned in section1, the proposed two branches can compete during the training process, which prioritizes each branch to learn complementary knowledge that can boost the parsing ability and improve learning efficiency. Thanks to this property, the results are still far better than the original encoder-decoder structure even if we remove a branch during the inference process. Moreover, the number of parameters is less than the original one, which alleviates the challenging to deploy complex models into practical applications in many real scenarios due to computer resources and run-time limitations. As shown in Table 5, we use an extremely simple branch, illustrated in Figure 1(b), retraining on the Cityscapes dataset, and we named the trained model '*ED*'. Moreover, we test the output results of each branch separately on the trained two-branch decoder model. Specifically, we take SegNet as an example. In the inference process, we only keep the upper branch of the model in Figure 5, and the output result obtained corresponds to 'O^*'. 'D^*' corresponds to the result of only keep the lower branch. '*O&D*' goes to the result of original two-branch model. The results of the upper branch in our trained two-branch model are 80.49%, 82.35%, and 83.87%, which significantly exceeds

Table 5: Single branch test on Cityscapes val set with 11 semantic class labels. '*Enc.*' represent encoder, '*Dec.*' represent decoder. '*O*' indicates the decoder deployed in the original model. '*D*' the decoder in Figure 1(b), '*T*' indicates our two-branch decoder. 'O^*', 'D^*' and '*O&D*' mean the result from upper branch, lower branch and final branch separately.

Methods	Enc.	Dec.	Mean IoU (%)			Parms. (M)		
SegNet	Vgg16	*O*	75.82			29.4		
ED		*D*	65.34			15.3		
SegNetT (ours)		*T*	O^*	D^*	*O&D*	O^*	D^*	*O&D*
			80.49	67.34	80.64	18.4	14.9	18.6
DeepLabv3+	Res50	*O*	80.31			26.6		
ED		*D*	76.67			32.2		
DeepLabv3+ (ours		*T*	O^*	D^*	*O&D*	O^*	D^*	*O&D*
			82.35	77.3	82.61	25.3	25.7	27.5
HRNet	Res50	*O*	82.34			9.6		
ED		*D*	81.25			9.7		
HRNet-T (ours)		*T*	O^*	D^*	*O&D*	O^*	D^*	*O&D*
			83.87	76.83	83.9	9.6	9.6	9.6

the counterparts of original encoder-decoder models (75.82%, 80.31%, and 82.34%). The results of lower branch trained in two-branch model are also better than correspond one-branch trained model. At the same time, the number of parameters used dropped remarkably. In addition, we find that the residual-like module can effectively combine the outputs of the two branches to further improve the final result to 80.64%, 82.61%, and 83.9%, as shown in '*O&D*' columns, which means that the final results are not adversely affected. The results once again show that our method can make each branch learns complementary information.

5 CONCLUSION

In this paper, we present a general two-branch decoder paradigm composed of a main branch and an auxiliary branch for scene segmentation. This decoder paradigm can be directly applied in an encoder-decoder framework to refine and integrate the information extracted by the encoder efficiently. With this two-branch decoder, we further propose a boundary enhanced complementary loss named BECLoss to guide two branches to learn complementary information. Moreover, we designed a simple yet efficient branch deployed as the auxiliary branch in our two-branch decoder. The comparative experiment shows that the proposed two-branch decoder paradigm and BECLoss can significantly improve the performance of the original encoder-decoder model consistently on challenging outdoor datasets. In addition, although we added a branch to the decoder, it did not significantly increase the number of parameters, and the added branch can be removed in the inference process while still getting performance far beyond the original counterpart.

REFERENCES

Badrinarayanan, V., Kendall, A., and Cipolla, R. (2017). Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495.

Chen, L.-C., Papandreou, G., Schroff, F., and Adam, H. (2017). Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*.

Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., and Adam, H. (2018). Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 801–818.

Chen, Y., Dapogny, A., and Cord, M. (2020). Semeda: Enhancing segmentation precision with semantic edge aware loss. *Pattern Recognition*, 108:107557.

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., and Schiele, B. (2016). The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223.

Fu, J., Liu, J., Tian, H., Li, Y., Bao, Y., Fang, Z., and Lu, H. (2019). Dual attention network for scene segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3146–3154.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Hu, J., Shen, L., and Sun, G. (2018). Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141.

Huang, Q., Xia, C., Wu, C., Li, S., Wang, Y., Song, Y., and Kuo, C.-C. J. (2017). Semantic segmentation with reverse attention. *arXiv preprint arXiv:1707.06426*.

Li, H., Xiong, P., An, J., and Wang, L. (2018). Pyramid attention network for semantic segmentation. *arXiv preprint arXiv:1805.10180*.

Li, X., Zhong, Z., Wu, J., Yang, Y., Lin, Z., and Liu, H. (2019). Expectation-maximization attention networks for semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9167–9176.

Long, J., Shelhamer, E., and Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440.

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer.

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Smith, L. N. (2017). Cyclical learning rates for training neural networks. In *2017 IEEE winter conference on applications of computer vision (WACV)*, pages 464–472. IEEE.

Takikawa, T., Acuna, D., Jampani, V., and Fidler, S. (2019). Gated-scnn: Gated shape cnns for semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5229–5238.

Tao, A., Sapra, K., and Catanzaro, B. (2020). Hierarchical multi-scale attention for semantic segmentation. *arXiv preprint arXiv:2005.10821*.

Valada, A., Mohan, R., and Burgard, W. (2019). Self-supervised model adaptation for multimodal semantic segmentation. *International Journal of Computer Vision*, pages 1–47.

Valada, A., Oliveira, G., Brox, T., and Burgard, W. (2016). Deep multispectral semantic scene understanding of forested environments using multimodal fusion. In *International Symposium on Experimental Robotics (ISER)*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. *arXiv preprint arXiv:1706.03762*.

Wang, J., Sun, K., Cheng, T., Jiang, B., Deng, C., Zhao, Y., Liu, D., Mu, Y., Tan, M., Wang, X., et al. (2020). Deep high-resolution representation learning for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*.

Woo, S., Park, J., Lee, J.-Y., and Kweon, I. S. (2018). Cbam: Convolutional block attention module. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19.

Yuan, Y., Xie, J., Chen, X., and Wang, J. (2020). Segfix: Model-agnostic boundary refinement for segmentation. In *European Conference on Computer Vision*, pages 489–506. Springer.

Zhao, H., Shi, J., Qi, X., Wang, X., and Jia, J. (2017). Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2881–2890.

Zheng, S., Lu, J., Zhao, H., Zhu, X., Luo, Z., Wang, Y., Fu, Y., Feng, J., Xiang, T., Torr, P. H., et al. (2020). Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. *arXiv preprint arXiv:2012.15840*.

Zhong, Z., Lin, Z. Q., Bidart, R., Hu, X., Daya, I. B., Li, Z., Zheng, W.-S., Li, J., and Wong, A. (2020). Squeeze-and-attention networks for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13065–13074.