Binary DAD-Net: Binarized Driveable Area Detection Network for Autonomous Driving

Alexander Frickenstein*[1], Manoj-Rohit Vemparala*[1], Jakob Mayr*[1],
Naveen-Shankar Nagaraja[1], Christian Unger[1], Federico Tombari[2,3], Walter Stechele[2]

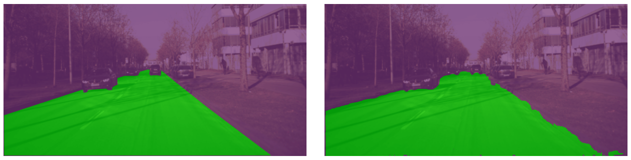
***Abstract*— Driveable area detection is a key component for various applications in the field of autonomous driving (AD), such as ground-plane detection, obstacle detection and maneuver planning. Additionally, bulky and over-parameterized networks can be easily forgone and replaced with smaller networks for faster inference on embedded systems. The driveable area detection, posed as a two class segmentation task, can be efficiently modeled with slim binary networks. This paper proposes a novel *binarized driveable area detection network (binary DAD-Net)*, which uses only binary weights and activations in the encoder, the bottleneck, and the decoder part. The latent space of the bottleneck is efficiently increased ($\times$32$\rightarrow$$\times$16 downsampling) through binary dilated convolutions, learning more complex features. Along with automatically generated training data, the binary DAD-Net outperforms state-of-the-art semantic segmentation networks on public datasets. In comparison to a full-precision model, our approach has a $\times$14.3 reduced compute complexity on an FPGA and it requires only 0.9MB memory resources. Therefore, commodity SIMD-based AD-hardware is capable of accelerating the binary DAD-Net.**

I. Introduction

Artificial Intelligence is often seen as the key enabler for fully autonomous driving due to the recent unprecedented success of deep learning (DL). However, two other key factors also need to be addressed in the field of AD.

Verifiable Software: With safety in mind, the AD stack is often modularized [1] into sensing and mapping, perception, and (path) planning blocks. Adhering to the modularization nature of AD, we pose the driveable area detection (DAD) as a two class segmentation task. In view of the DAD task, the segmentation task for topologically open contours, like roads, is relatively easier than object segmentation where missing a part can have adverse effects, see Fig. 1. Moreover, compared to a multi-class segmentation the two class driveable area detection offers a higher precision due to better separability between their respective features.

Efficient Application: Secondly, AD relies on a real-time system which implicitly imposes resource constraints (memory, bandwidth, run-time) on the underlying algorithms. Due to the real-time requirement, the performance of CNNs also need to be measured w.r.t. the power consumption (apart from standard metrics). Recent literature studies lightweight CNNs based on network optimization, i.e. pruning [2] and quantization [3]. One extreme is represented by Binary Neural Networks (BinaryNets) by constraining weights and activations to ± 1. Computationally, they use SIMD-based logical gate (bit-wise) operations instead of full-precision multiply-accumulate (MAC) operations. BinaryNets, are light weight w.r.t. the memory demand and computational cost, offer a trade-off between efficiency vs. accuracy.



(a) Manually labeled data. (b) Prediction of binary DAD-Net.

Fig. 1. Segmentation output of binary DAD-Net using only ± 1 weights and activations on proprietary fleet data. The misclassified driveable area pixels in (b) can be easily regularized by a groundplane detection algorithm.

Considering these two enabling factors for autonomous driving systems naturally make the two class driveable area detection [4] a perfect case for BinaryNets significantly reducing the memory demand and the computational complexity. To best of our knowledge binary DAD-Net is the first work fully binarizing a semantic segmentation model for the DAD task. The contribution of this work are summarized as follows:

- Efficiently optimized various blocks in the model's encoder, bottleneck and decoder, combining structural and local binary approximation schemes. A detailed ablation study is provided investigating different variants in binary DAD-Net.
- The proposed binary model performs similar to the full-precision network gaining $14.3\times$ computational efficiency and $15.8\times$ memory saving for Cityscapes dataset on the DAD task.
- The performance of binary DAD-Net is increased when pre-trained on automatic annotations.

II. Related work

In the literature, several methods have been proposed to address the task of semantic segmentation, see Sec. II-A. Efficient training and perception, *i.e.* binary neural networks, are detailed in Sec. II-B.

A. *Driveable Area Detection*

The task of detecting the driveable area from images has been extensively studied. Monocular camera systems [5] use a homography computed from two consecutive images which provides information about the ground plane. Similarly, [6]

*Authors contributed equally

[1] BMW Group, Autonomous Driving, Munich, Germany, `<Firstname>.<Lastname>@bmw.de`

[2] Technical University of Munich, Munich, Germany, `tombari@in.tum.de`, `walter.stechele@tum.de`

[3] Google Inc. , Zurich, Switzerland

utilize a single camera and a variety of features including the homography in combination with a SVM learning approach to solve the task. Early systems focused only on the driveable area, whereas current deep learning approaches usually address the problem of full-image segmentation which includes the driveable area. One of the first prominent semantic segmentation models proposed was the Fully Convolutional Network (FCN), is successfully adopted by Shelhamer et al. [7]. It is shown that these networks are difficult to train from scratch and require a pre-trained classification model (encoder). Another important aspect of FCN are the skip connections which capture the intermediate features from the high level feature maps during the up-sampling stage. This method paved path to further, more structured models such as UNet [8]. This structured up-sampling provides higher accuracy than single $\times 8$ up-sampling, i.e. FCN. However, this increases the computational complexity. DeepLab, proposed in [9], utilizes dilated convolution instead of down-sampling the feature maps maintaining the sufficient receptive field. The pooling or strided convolution is avoided for the last set of feature maps. This would increase the computational costs as the convolution is performed on larger feature maps. The encoder network is downsampled by a factor of 8/16 instead of 32. The down-sampled featured maps are then passed to a spatial pyramid pooling module, which consists of parallel dilated convolution with different rates followed by concatenation and point-wise convolution. This module produces better segmentation results by extracting multi-scale information. Multi-class semantic segmentation has a negative effect on the precision of the driveable area detection algorithm and their vast number of MAC operations making the application impractical for embedded systems.

B. Binary Neural Networks

Binarization of CNNs attempts to constrain weights and activations to just ± 1. However, Binary Neural Networks observe a degradation in accuracy compared to their full-precision counterpart. BinaryNets proposed by Hubara et al. [10] relies on deterministic binarization functions and the STE estimator [11] during training. The degradation in accuracy of full-precision to binary weights can be reduced by suitable approximation techniques. In the binary model XNOR-Net, introduced by Rastegrati et al. [12], the real-valued weights and activations are estimated by introducing scaling factors alongside with the binary weights and activations. CompactBNN, proposed by Tang et al. [13], focuses on improving the approximation towards the activations, as they observe that binarizing activations is more challenging compared to the weights. [13] also propose the trainable parametric ReLU as activation function to further improve the training. Recently, ABC-Net [14] projects both, the full-precision weights and activations into corresponding linear combinations of its binary approximation with individual shifting and scaling factors. Further, they argue that a BinaryNets with multiple binary weight and activation bases is more suited for embedded systems than an equivalent fixed-point quantized CNN. The MAC operation consumes $> 8\times$ more power than a bit-wise operation using 45nm CMOS technology [15]. All publications mentioned above have consequently improved BinaryNets for image classification. Binary object detection models are studied by Hanyu et al. [16]. Zhuang et al. [17] extend the approximations further towards the structure level and proposes GroupNet with multiple binary bases. To the best of our knowledge this is the only work in the domain of BinaryNets reporting results to semantic segmentation. GroupNet introduces the Binary Parallel Atrous Convolution (BPAC) module. The BPAC module consists of multiple dilated convolutions with various dilation rate (up to 16), which causes irregular memory accesses (inefficient) and an higher power-consumption of the memory controller [2]. Moreover, introducing multiple binarizations indices in the binary DAD-Net is not beneficial for the DAD task, as discussed in Tab. II of the experimental section.

III. Binary Drivable Area Detection Network

The proposed driveable area detector is inspired by autoencoder-based networks with skip connections, *i.e.* DeepLabV3 [9]. As the name implies, binary DAD-Net has binary representations in all three parts of the model: the encoder, bottleneck (latent space) and decoder. The modules are detailed in Sec. III-A-III-C. Binary DAD-Net adopts the binarization scheme of Rastegari et al. [12] as discussed in Sec. III-D. The structure of binary DAD-Net is given in Fig. 2.

A. Binary Encoder for Feature Extraction

Binary Convolution: Without loss of generality, an activation $A_{l-1} \in \mathbb{R}^{h \times w \times c_i}$ is considered as an input to a convolutional layer $l \in [1, L]$. In the case of $l = 1$, the activation A_1 is the input image I. Moreover the weights $W_l \in \mathbb{R}^{k \times k \times c_i \times c_o}$ are the trainable parameters of the layer. The sign-function binarizes the real-valued activations $H_{l-1} \approx \texttt{sign}(A_{l-1})$. In the inference-stage the weights are considered to $B \approx \texttt{sign}(W) \in \{-1, +1\}$. Moreover, the activations are normalized using BatchNormalization [18]. Scale factors α and β, introduced in [12], find better estimations for $W \approx \alpha B$ and $A \approx \beta H$, see Eq. 1. The first convolutional layer is not binarized due to very few trainable parameters and computations compared to the remaining binary DAD-Net's layers.

$$A_l = \texttt{Conv}(W_S^l, f_S^{l-1}) \approx \alpha\beta\texttt{Conv}(B^l, H^{l-1}) \qquad (1)$$

Binary Residual Block: The residual block, introduced by He et al. [19], can be easily binarized learning more complex features as a regular binary convolutional layer, see Sec. IV-B. In detail, the binary residual block is built of consecutive binary convolutional layers including BatchNormalization and a non-linear activation. The shortcut connections in binary residual blocks are an adequate technique to overcome the gradient mismatch problem. Also for the binary version, these blocks combine the information obtained from the previous layer through fusing the identity connections with the output of the current layer.

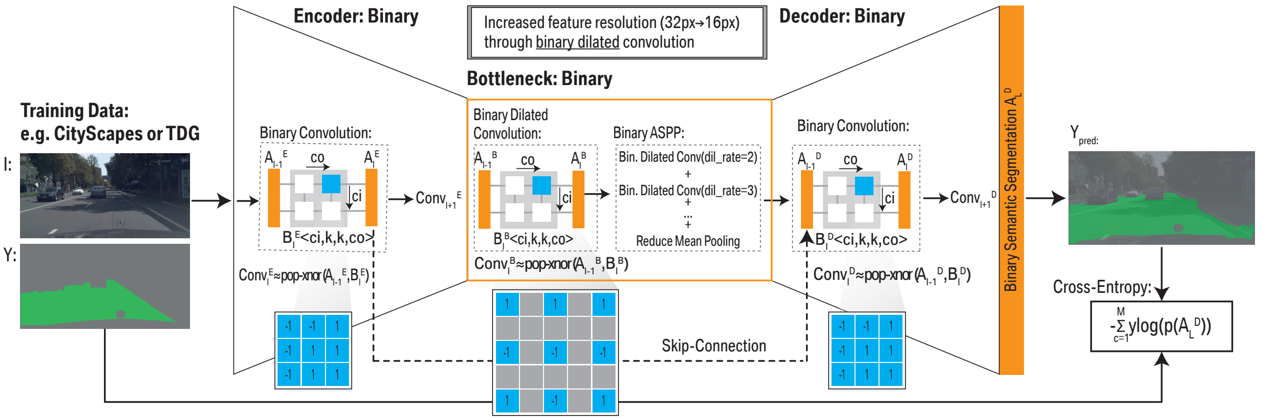
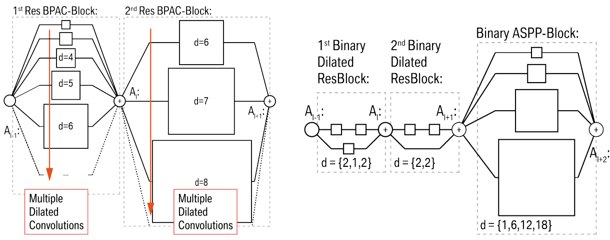

Encoder: Binary
Increased feature resolution (32px→16px) through binary dilated convolution
Decoder: Binary
Bottleneck: Binary
Training Data: e.g. CityScapes or TDG
I:
Y:
Binary Convolution:
co
ci
A_{l-1}^E
A_l^E
B_l^E<ci,k,k,co>
$Conv_{l+1}^E$
$Conv_l^E$≈pop-xnor(A_{l-1}^E,B_l^E)
Binary Dilated Convolution:
A_{l-1}^B
A_l^B
B_l^B<ci,k,k,co>
$Conv_l^B$≈pop-xnor(A_{l-1}^B,B_l^B)
Binary ASPP:
Bin. Dilated Conv(dil_rate=2)
+
Bin. Dilated Conv(dil_rate=3)
+
...
+
Reduce Mean Pooling
Binary Convolution:
A_{l-1}^D
A_l^D
B_l^D<ci,k,k,co>
$Conv_{l+1}^D$
$Conv_l^D$≈pop-xnor(A_{l-1}^D,B_l^D)
Binary Semantic Segmentation A_L^D
$Y_{pred:}$
Cross-Entropy:
$-\sum_{c=1}^{M} y\log(p(A_L^D))$
Skip-Connection
-1 -1 1
-1 1 1
-1 1 1
1 1 1
-1 -1 -1
1 -1 1

Fig. 2. Overview of binary DAD-Net. The binarized network consists of three parts, namely an encoder, a bottleneck and a decoder. Binary dilated convolution in the bottleneck ensures an extended feature resolution. The feature response is efficiently increased from 32 to 16 neurons through binary dilated convolutions.

B. Binary Bottleneck with Enlarged Receptive Field

The bottleneck layers in the segmentation architecture retain the lowest spatial dimensions obtained from the encoder, likewise to an auto-encoder. Similar to the binary convolutional layer, weights and activations are binarized for the bottleneck, see central building block of Fig. 2.

Inspired by DeepLabv3 [20], to increase the receptive field of a convolutional layer, dilated convolution introduces zeros to the weights of the respective layer. The distance between two neighboring weights is called dilation rate, see also the bottleneck's weights (gray parts) of Fig. 2. A typical binary dilated convolution block consists of 1) binarization of the activations, 2) binary convolution, 3) BatchNormalization and 4) non-linear activations such as ReLU. It is important to apply the non-linear activation after the normalization (unit variance and zero mean) to prevent the feature map from losing too much information. Our observation for dilated convolution fits previous investigations for vanilla binary convolution layer [12], [13], [14].



(a) BPAC based bottleneck.
(b) Binary DAD-Net bottleneck.

Fig. 3. Comparison between BPAC [17] and BinDAD bottleneck.

The central part of the DeepLab [20] inspired binary DAD-Net is the bottleneck, see Fig. 3(b) In detail, the bottleneck consists of two consecutive binary residual blocks and a binary atrous spatial pyramid pooling (ASPP)-block. The dilation rate d=2 for the residual blocks and $d = \{1, 8, 12, 18\}$ for the binary ASPP-block is employed. Different to previous residual blocks, the dilated residual blocks do not downsample the feature maps. Thus, the feature resolution of the binary bottleneck is efficiently increased. Upsampling by a factor of 16 instead of 32 is required.

Differently, structured BinaryNet [17] employs multiple parallel dilated convolution (8), called BPAC-block, in the binary dilated residual blocks. Their first residual block has dilation rates $d = \{2, 3, 4, 5, 6, 7, 8, 9\}$ and the second one has $d = \{6, 7, 8, 9, 10, 11, 12, 13\}$ for GroupNet with 8 bases. Zhuang et al. [17] skip the ASPP-block in their design, see also Fig. 3(a). In case of the DAD task we observe an accuracy degradation with this technique, see Tab. III. Moreover, the memory accesses for the BPAC based residual blocks becomes very inefficient.

C. Binary Decoder for Semantic Predictions

Third, to best of our knowledge binary DAD-Net is the first work also binarizing the decoder for driveable area detection, see right building block of Fig. 2. Employing only binary convolutions enlarges the output of the bottleneck to the size of the original input image I generating pixel-wise predictions for the task of driveable area detection. The binary decoder also consists of bilinear upsampling and a binary score layer. In detail, after the binary dilated convolution, described in the previous section, linear combination (binary 1×1 convolution) of the ASPP feature maps and the encoder skip connection (after the first residual block) is computed. Next, the feature maps are fused in two consecutive binary refinement blocks. The binary refinement blocks consist of 3×3 kernels, which is similar to the binary convolutional layer, described above. Instead of transpose convolutions, bilinear up-sampling enlarges the feature maps to the size of the input I. This is important as the binary transpose convolution would introduce additional operations and would lead to an accuracy degradation, see Sec. IV-B.

D. Training Scheme of Binary DAD-Net

Consider an L-layer BinaryNet (*i.e.* 29 for binary DAD-Net) f which takes I as the input image, trained on semantic labels Y, with real-valued latent weights $W \in \mathbb{R} : [-1, 1]$ as the trainable parameters. Y can refer to expensive hand-crafted annotations or to the automatic training data generator

(TDG) annotations which are described in the experimental results. The training algorithm is given in Algorithm 1.

Algorithm 1 Training an L-layer binary DAD-Net.

```
Require: a minibatch of images $I$ and lables $Y$, initialized
    weights $W$ and learning rate $\eta$.
    Forward propagation:
 1: for $l = 2$ to L do
 2:     Compute $B_l$ and $H_{l-1}$
 3:     $A_l \leftarrow Conv(B_l, H_{l-1})$                ▷ Eq. 1
    Optionally:
 4:     $S_l \leftarrow \texttt{MaxPool}(A_l)$
 5:     $O_l \leftarrow BatchNorm(S_l, \Theta_l)$
 6:     $A_l \leftarrow ReLu(O_l)$
 7: $\tilde{Y} \leftarrow A_L$
    Backward propagation:
 8: $gA_L \leftarrow \partial\mathcal{L}/\partial A_L$
 9: for $l = L$ to 1 do
10:     Optionally: $g\Theta_l, gO_l \leftarrow BackBN(gA_l, O_l, \Theta_l)$
11:     $gW_l, gA_{l-1} \leftarrow BackConv(gS_l, H_{l-1}, W_l)$   ▷ Eq.2
    Update the trainable parameter:
12: for $l = 1$ to L do
13:     $W_l \leftarrow Update(W_l, \eta, gw_l)$          ▷ Momentum
14:     $\Theta_l \leftarrow Update(\Theta_l, \eta, g\Theta_l)$
15:     $\eta \leftarrow \lambda\eta$
```

In detail, $B = \texttt{sign}(W)$ approximates W into the binary domain as $B \in \{-1, +1\}$. The gradient of the sign operation vanishes everywhere and therefore the gradient is estimated in order to update the real-valued weights during the training phase. In the simplest case, the estimated gradient could be obtained by replacing the derivative of sign with the identity function, see Eq. 2. This is referred as the straight through estimator [21].

$$g_w = g_{W_b} 1_{|w| \leq 1} \tag{2}$$

Thus, an efficient set of binary weights $\tilde{B} \in \{-1, +1\}$ is trained by minimizing the expected loss $\mathcal{L}$ according to the prediction $\tilde{Y}$ and annotations Y, as shown in the Eq. 3. $\tilde{B}$ is later used during forward propagation and for the inference on the embedded hardware.

$$\tilde{B} = \underset{W \in [-1,+1]^L}{\text{argmin}} \mathcal{L}(\mathbb{E}[(f(I, B)], Y) \tag{3}$$

Lines 1-7 show the equivalent forward path of binary DAD-Net, resulting in a semantic prediction $\tilde{Y}$. The loss is a combination of the pixel-wise cross entropy and a L2 regularization loss. The gradients are computed by minimizing the cost function $\mathcal{L}$ from line 8 to 11 (backpropagation). The STE of Eq. 2 is used to estimate the binary weights. The gradients gW and $g\Theta$ of the weights W and BatchNorm parameters Θ are applied by Update(), with Momentum optimizer, see lines 12-15. The loss also trains the scaling factors α and β associated with weights and the activations [12].

IV. Experiments and Results

The following section is structured as follows: the datasets, training procedure and performance metrics for the benchmarks of binary DAD-Net are introduced in Sec. IV-A; in Sec. IV-B, the configuration space of binary DAD-Net is explored to get more insight in order to the improve the performance, finally in Sec. IV-C, the binary DAD-Net is compared to SotA semantic segmentation models.

A. Benchmark Datasets and Automatic Annotations

Cityscapes: The CityScapes dataset [22] consists of 2975 training images, 500 validation images, and 1525 test images including their corresponding ground truth labels. Ground truth labels for the test set are not publicly available. The images of size 2048 × 1024 show German street scenes along with their pixel-level semantic labels of 30(19) classes. However, for training the raw images are down-sampled to a size of 1024×512. For an elaborate comparison of the ground plane detection, the human labeled ground truth validation data from the road and the parking area class are considered as driveable area. The remaining classes are assigned to the non-driveable area, i.e. pedestrians, sidewalks and cars.

KITTI Road: The KITTI Road dataset [23] consists of 289 images with manually annotated ground truth labels. This data is split into 259 images for training and 30 images for validation. The raw images are down-sampled using nearest neighbour scaling algorithm from (1242 × 375) to (1024 × 256). The experimental setup for the KITTI Road dataset is similar to the CityScapes dataset.

Automatic Annotations: The training data generator (TDG), proposed by Mayr et al. [24] automatically generates annotations for the task of driveable area segmentation. In case of the automatically labeled KITTI data, the data amount is increased compared to manual labeling. In total, we use automatic annotations generated from 10900 images instead of 259 images as training dataset. This dataset also improves the performance binary DAD-Net by providing a good initialization, which is further fine-tuned on the corresponding DAD task.

Training Procedure: The binary DAD-Net is trained using the Momentum optimizer with a base learning rate $\eta = 0.01$, the momentum $\gamma = 0.9$ and weight decay $\lambda = 0.0005$. The learning is dropped by a factor of 0.9 every 8 epochs. For the DAD task, all the models are trained for 240 epochs and the results are reported after retraining the batch statistics.

Performance Metrics: The metrics reported in this experiments correspond to mean Intersection-over-Union (IoU), Average Precision (AP), False Positive Rate (FPR) and False Negative Rate (FNR) as used in the KITTI Road challenge [23]. For applications as autonomous driving, it is crucial that the perception models have real-time capability. The modern deep learning inference engines such as NVIDIA-T4 GPU [25] and Xilinx FPGAs [26] with DSP48 blocks support SIMD-based bit-wise operations. In particular, a single DSP48 block can perform two 16-bit fixed-point multiplications or 48 XNOR operations at once [27]. The normalized compute complexity (NCC), allowing an implementation-wise comparison, is defined as the optimal utilization of MAC and XNOR operations in one compute unit. The DSP48 block serves as a reference implementation to compute NCC for the further experiments.

B. Configuration Space Exploration

The requirement of the following analysis is to determine appropriate modules for binary DAD-Net (encoder and decoder), a local binary approximation (XNOR|CompactBNN|ABC), structural approximation schemes of the bottleneck (BPAC|Dilation|ASPP) and good initialization scheme(ImageNet|Automatic annotations). Analysis is performed on CityScapes dataset. **Encoder/Decoder Selection:** The Tab. I compares the mIoU on CityScapes dataset, the number of operations and the memory demand for storing the parameter of different encoder-decoder configurations. A detailed comparison is shown in the supplementary material. The models of Tab. I are trained in full-precision in order to select the right configuration. Based on a high mIoU, the lowest number of operations and an appropriate memory demand, ResNet18 is chosen as encoder and DeepLabV3 as decoder for the DAD task. Moreover, the standard bottleneck of DeepLabV3 only consists of dilated convolutions and has no transpose convolution in the decoder, enabling an efficient binarization for binary DAD-Net.

TABLE I
SELECTION OF A SOTA ENCODER/DECODER CONFIGURATION.

Encoder	Decoder	mIOU [%]	GOPs	Mem. [MB]
VGG16	FCN8	96.75	222.1	268.5
VGG16	UNet	95.92	234.0	272.2
ResNet18	FCN8	97.05	**20.93**	22.60
ResNet18	UNet	97.50	23.58	23.26
ResNet18	**DeepLabV3**	**97.54**	29.77	**14.64**

Local Binarization Scheme: In this section three binarization schemes are analyzed for the DAD task. The results are given in Tab. II. By adopting XNOR binarization [12], the model is trained with one weight and activation base including scaling factors α and β. For details see Sec. III. Differently, CompactBNN [13] introduces 3 activation bases and therefore increases the computational complexity of binary convolutional layers. For the DAD task no accuracy improvement is observed. Finally, the binarization scheme of ABC-Net [14] introduces multiple activation and weight bases (*i.e.* 3×3) to the binary convolution. If only the encoder is binarized with different local approximation schemes, the mIoU remains almost the same (See row. 1, 2, 3). Contrary, if the bottleneck and the decoder are binarized, multiple binarizations result in a degraded driveable area detection. Moreover, NCC and memory demand are increased compared to the XNOR binarization (See row. 4, 5, 6).
Bottleneck Configuration: In Tab. III different bottleneck configurations are analyzed. We replace the bottleneck in BinDAD with the BPAC module proposed in [17]. Introducing dilations in the bottleneck improves the mIoU for driveable area detection. Zhuang et al. [17] argues that BPAC modules capture different object scales making the ASPP obsolete. However, in the case of DAD, a dedicated ASPP block shows better accuracy. Moreover, the implementation of BPAC modules becomes inefficient in HW as the dilation rate increases due to irregular memory access on a general inference processor.

TABLE II
LOCAL BINARY APPROXIMATION OF THE ENCODER AND DECODER.

Encoder/Decoder	Binarization			mIOU	NCC	Mem.
	En.	De.	Scheme	[%]	[$\times10^9$]	[MB]
ResNet18+DeepLab	✓	✗	**XNOR**	**96.93**	7.30	4.83
ResNet18+DeepLab	✓	✗	Compact	96.83	7.96	4.83
ResNet18+DeepLab	✓	✗	ABC	96.85	9.94	5.54
Binary DAD-Net	✓	✓	**XNOR**	96.23	**0.73**	**0.92**
Binary DAD-Net	✓	✓	Compact	92.40	1.96	0.92
Binary DAD-Net	✓	✓	ABC	93.26	5.65	2.75

TABLE III
CHOOSING THE BOTTLENECK FOR BINARY DAD-NET.

Bottleneck config.	mIOU	NCC	Mem.
Binary DAD-Net w/o Dilations + ASPP	94.80	0.65	0.69
Binary DAD-Net w **Dilations** w/o ASPP	95.18	**0.65**	**0.69**
Binary DAD-Net w BPAC [17]	96.02	1.36	0.69
Binary DAD-Net w **Dilations w ASPP**	**96.23**	0.73	0.92

Automatic annotations In the field of AD, automatic annotations are a low priced alternative to costly hand labeled data enabling a competitive deployment. On the one hand, automatic annotations are noisy and have much higher variance than manually labeled finite datasets, e.g. CityScapes. On the other hand, BinaryNets are prone to a degradation in accuracy because of their limited learning capabilities. When binary DAD-Net is trained on automatic annotations, it achieves on-par accuracy compared to SotA semantic segmentation model, *i.e*. DeepLab. Referred to Tab.IV, TDG data have an mIoU of 69.7% with respect to the manually labeled validation dataset of KITTI (See row. 1). The training of DeepLabV3 on only automatic annotations (TDG) results in an mIoU of (86.1%). The 38× more efficient binary DAD-Net achieves on-par accuracy (See row. 2, 3).

TABLE IV
BINARY DAD-NET'S PERFORMANCE ON AUTOMATIC ANNOTATIONS.

Model	Taining Data	mIOU	Acc.
TDG [24]	TDG	69.70	87.73
DeepLabV3	TDG	86.11	90.49
Binary DAD-Net	TDG	85.33	90.49
Binary DAD-Net	CityScapes	96.23	96.13
Binary DAD-Net	**TDG + CityScapes**	**96.60**	**96.68**

Second, the training of BinaryNets is highly sensitive to initialization. In [9], the encoder is pre-trained on ImageNet and later fine-tuned on segmentation task. With the same initialization strategy, binary DAD-Net achieves an mIOU of (96.23%). However, when binary DAD-Net is fine-tuned on the pre-trained TDG data, there is an improvement of 0.37% in mIOU. (See, row 4, 5) in Tab.IV.

TABLE V

PERFORMANCE COMPARISON OF BINARY DAD-NETTO BASELINE MODELS. COMPUTATIONS REFERS TO MAC AND XNOR OPERATIONS. THE NCC METRIC (COLUMN 5) CONSIDERS AN EFFICIENT HW IMPLEMENTATION,E.G. WITH SIMD-BASED ACCELERATION OF XNOR OPS.

Approach	**Datasets**	**Parameters** [MB]	**Computations** [GOP]	**NCC** $\times 10^9$	**mIOU** [%]	**Accuracy** [%]	**FPR** [%]	**FNR** [%]
FCN8s	CityScapes	22.60	20.93	10.47	96.94	97.01	1.62	5.71
DeepLabv3	CityScapes	14.63	26.54	14.89	97.30	97.29	**1.32**	5.41
UNet	CityScapes	23.26	23.58	11.79	**97.50**	**97.55**	1.59	**5.33**
FCN8s-XNOR	CityScapes	1.41	20.93	**0.66**	95.19	95.35	2.72	8.54
Binary DAD-Net (Ours)	**CityScapes+TDG**	**0.92**	29.77	0.73	96.60	96.68	1.96	6.16
FCN8s	KITTI Road	22.60	10.50	5.250	**95.43**	**97.71**	3.92	1.91
DeepLabv3	KITTI Road	14.63	20.93	10.46	94.45	96.34	**1.02**	2.12
UNet	KITTI Road	23.26	10.71	5.36	93.26	95.50	4.81	**1.51**
FCN8s-XNOR	KITTI Road	1.41	10.50	0.33	92.10	96.34	4.92	3.44
Binary DAD-Net (Ours)	KITTI Road	**0.92**	13.26	**0.27**	95.25	97.05	7.83	1.82

C. Comparison with the State-of-the-Art

In this section, the structure of the proposed binary DAD-Net is analyzed and evaluated against SotA models for semantic segmentation on both public datasets. Also a variety of encoder models with different binarization methodologies proposed in [12], [13] and [14] are employed.

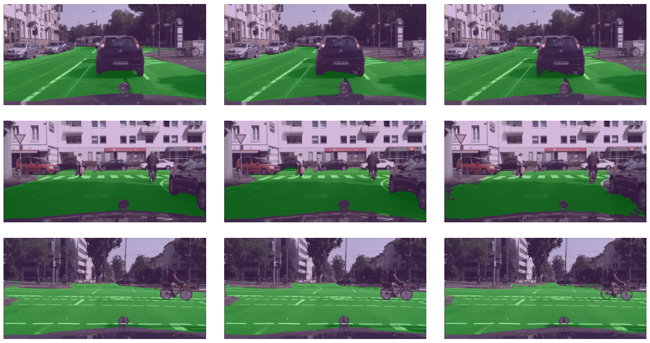
TABLE VI

A DEDICATED TWO CLASS DAD MODEL (RIGHT) IS COMPARED TO A MULTI-CLASS MODEL (LEFT) JUSTIFYING BINARY DAD-NET.

	All 19 classes		**Only DAD (2 class)**	
Metric\Model	**FP DeepLab**	**Binary DAD-Net**	**FP DeepLab**	**Binary DAD-Net**
Road IoU [%]	97.23	97.22	**97.55**	96.79
meanIoU [%]	63.43	58.12	**97.30**	96.60
Precision [%]	79.23	80.00	**97.36**	94.30
Recall [%]	71.12	71.21	**95.13**	93.53

Firstly, a justification of a dedicated two class driveable area detection is given in Tab. VI. The IoU of the individual road class of the full-precision segmentation network is 97.23%. However, training a model with many classes, i.e. 19, causes the precision and recall for the task of driveable area segmentation to degrade compared to the dedicated two class DAD task. This also holds for the binary implementation, see column 2 and 4. Moreover, binary DAD-Net achieves comparable results (only -0.70% mIoU) compared to the bulky full-precision DeepLabV3 on the DAD task.

Full-precision models have been proven to be dispensable for most prediction tasks, rendering 16-bit fixed-point representations an adequate alternative [28]. With respect to the memory requirements and the compute complexity, binary DAD-Net is compared to a 16-bit implementation rather than a 32-bit one. Tab. V shows the performance of different CNNs for driveable area detection, including binary DAD-Net. The models are trained on the CityScapes and the KITTI Road datasets separately. Binary DAD-Net achieves a mIoU of 96.60% on the CityScapes and 95.25% on the KITTI Road dataset which constitutes to an improvement of $+1.05\%$ (3.15%) compared to the previous best BinaryNet. The improvement over FCN8s-XNOR is due to the highly representational bottleneck block discussed in Sec. III-B. Apart from the comparison with BinaryNets, binary DAD-Net observes a slight accuracy degradation of -0.9 (-0.7) compared to the state-of-the-art full-precision segmentation networks, i.e. UNet (DeepLabV3). Fig. 4 displays some quantitative results on different scenarios in the CityScapes dataset. The misclassified pixels of binary DAD-Net, compared to the full-precision counterpart, can be easily regularized by a ground plane detection algorithm. Moreover, the predictions show that a minor accuracy degradation is negligible taking the performance advantages into consideration.



(a) DeepLabv3 [20]. (b) FCN8s [7]. (c) Binary DAD-Net.

Fig. 4. Quantitative results on different scenarios in CityScapes dataset. The last column shows the semantic predictions of binary DAD-Net.

CONCLUSION

This paper introduces a novel binary driveable area detector (binary DAD-Net) required in the field of autonomous driving. Binary DAD-Net is fully binarized, including the encoder, the bottleneck and the decoder. An elaborate study is performed to explore various components of Binary DAD-Net, namely the model structure, the binarization scheme and the ground-truth annotations for training. Along with automatically generated training data, binary DAD-Net achieves state-of-the-art semantic segmentation results $96.60\%(-0.7\%)$ on the CityScapes dataset. The proposed driveable area detector is very memory efficient, with only 0.9MB parameters (-15.9×). Moreover, Binary DAD-Net shows its superior performance w.r.t. an embedded implementation, by drastically reducing the computational complexity $(14.3\times)$ compared to previous work.

References

[1] Shai Shalev-Shwartz and Amnon Shashua. On the sample complexity of end-to-end training vs. semantic abstraction training. *CoRR*, abs/1604.06915, 2016.

[2] Alexander Frickenstein, Manoj Rohit Vemparala, Christian Unger, Fatih Ayar, and Walter Stechele. DSC: Dense-sparse convolution for vectorized inference of convolutional neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019.

[3] Manoj Rohit Vemparala, Alexander Frickenstein, and Walter Stechele. An efficient FPGA accelerator design for optimized cnns using openCL. In *Architecture for Computing Systems (ARCS)*, May 2019.

[4] Xiaolong Liu and Zhidong Deng. Segmentation of drivable road using deep fully convolutional residual network with pyramid pooling. *Cognitive Computation*, 10:272–281, Nov 2017.

[5] Manolis I.A. Lourakis and Stelios C. Orphanoudakis. Asian Conference on Computer Vision (ACCV). 1997.

[6] J. Yao, S. Ramalingam, Y. Taguchi, Y. Miki, and R. Urtasun. Estimating drivable collision-free space from monocular video. In *2015 IEEE Winter Conference on Applications of Computer Vision*, pages 420–427, Jan 2015.

[7] Evan Shelhamer, Jonathan Long, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(4):640–651, April 2017.

[8] O. Ronneberger, P.Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, volume 9351 of *LNCS*, pages 234–241. Springer, 2015.

[9] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision – ECCV 2018*, pages 833–851, Cham, 2018. Springer International Publishing.

[10] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems (NeurISP)*, pages 4107–4115. Curran Associates, Inc., 2016.

[11] Yoshua Bengio, Nicholas Léonard, and Aaron C. Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *CoRR*, abs/1308.3432, 2013.

[12] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. XNOR-net: Imagenet classification using binary convolutional neural networks. In *ECCV*, 2016.

[13] Wei N. Tang, Gang Hua, and Liang Wang. How to train a compact binary neural network with high accuracy? In *AAAI Conference on Artificial Intelligence (AAAI)*, 2017.

[14] Xiaofan Lin, Cong Zhao, and Wei Pan. Towards accurate binary convolutional neural network. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems (NeurISP)*, pages 345–353. Curran Associates, Inc., 2017.

[15] Song Han, Jeff Pool, John Tran, and William J. Dally. Learning both weights and connections for efficient neural networks. In *Advances in Neural Information Processing Systems (NeurISP)*, NIPS'15, pages 1135–1143, Cambridge, MA, USA, 2015. MIT Press.

[16] Siyang Sun, Yingjie Yin, Xingang Wang, De Xu, Wenqi Wu, and Qingyi Gu. Fast object detection based on binary deep convolution neural networks. *CAAI Trans. Intell. Technol.*, 3:191–197, 2018.

[17] B. Zhuang, C. Shen, M. Tan, L. Liu, and I. Reid. Structured binary neural networks for accurate image classification and semantic segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 413–422, June 2019.

[18] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015.

[19] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, June 2016.

[20] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision – ECCV 2018*, pages 833–851, Cham, 2018. Springer International Publishing.

[21] Yoshua Bengio, Nicholas Léonard, and Aaron C. Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *CoRR*, abs/1308.3432, 2013.

[22] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[23] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[24] Jakob Mayr, Christian Unger, and Federico Tombari. Self-Supervised Learning of the Drivable Area for Autonomous Vehicles. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 362–369. IEEE, 01.10.2018 - 05.10.2018.

[25] Nvidia. NVIDIA Turing GPU Architecture. In *https://www.nvidia.com/content/dam/en-zz/\newlineSolutions/design-visualization/technologies/turing-architecture/NVIDIA-Turing-Architecture-Whitepaper.pdf (Accessed: 28/02/20)*, 2017.

[26] XILINX 7 series dsp48e1 slice. In *https://www.xilinx.com/support/documentation/user_guides/ug479_7Series_DSP48E1.pdf (Accessed: 28/02/20)*, number UG479. Xilinx, Inc, 3 2018. v1.10.

[27] Alexander Frickenstein Walter Stechele Nael Fasfous, Manoj Rohit Vemparala. OrthrusPE: Runtime reconfigurable processing elements for binary neural networks. In *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, Grenoble, France, 2020.

[28] Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaev, Ganesh Venkatesh, and Hao Wu. Mixed precision training. 10 2017.