This WACV 2022 paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

D^{2}Conv3D: Dynamic Dilated Convolutions for Object Segmentation in Videos

Christian Schmidt Ali Athar Sabarinath Mahadevan Bastian Leibe
Computer Vision Group, RWTH Aachen University
christian.schmidt4@rwth-aachen.de {athar,mahadevan,leibe}@vision.rwth-aachen.de

Abstract

Despite receiving significant attention from the research community, the task of segmenting and tracking objects in monocular videos still has much room for improvement. Existing works have simultaneously justified the efficacy of dilated and deformable convolutions for various image-level segmentation tasks. This gives reason to believe that 3D extensions of such convolutions should also yield performance improvements for video-level segmentation tasks. However, this aspect has not yet been explored thoroughly in existing literature. In this paper, we propose Dynamic Dilated Convolutions (D^{2}Conv3D): a novel type of convolution which draws inspiration from dilated and deformable convolutions and extends them to the 3D (spatio-temporal) domain. We experimentally show that D^{2}Conv3D can be used to improve the performance of multiple 3D CNN architectures across multiple video segmentation related benchmarks by simply employing D^{2}Conv3D as a drop-in replacement for standard convolutions. We further show that D^{2}Conv3D out-performs trivial extensions of existing dilated and deformable convolutions to 3D. Lastly, we set a new state-of-the-art on the DAVIS 2016 Unsupervised Video Object Segmentation benchmark. Code is made publicly available at `https://github.com/Schmiddo/d2conv3d`.

1. Introduction

The task of segmenting objects from monocular video sequences has received significant attention from the research community in recent years, mainly because of useful applications in self-driven cars, autonomous robots, *etc*. Several existing approaches [23, 44, 4] for this task follow a two-step paradigm where objects are first segmented in individual image frames, followed by a second temporal association step. Such methods leveraged the availability of accurate image-level instance segmentation networks [26, 15] and various cues for temporal association (*e.g*. attention, optical flow, Re-ID) [23, 44, 38, 29]. More recently however, methods have emerged [16, 1, 24] which use 3D convolutions to jointly reason over spatial and temporal dimensions,

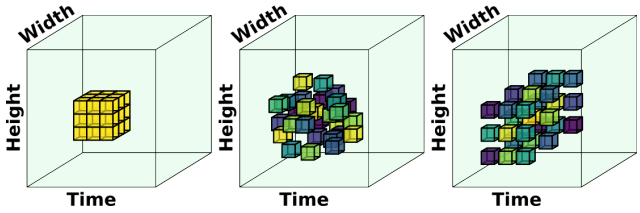

Figure 1: Comparison of regular convolutions (left), modulated deformable convolutions [43] (middle), and D^{2}Conv3D (right). Note that D^{2}Conv3D predicts a distinct spatiotemporal dilation for every point in the volume. Different colors indicate different modulation values.

resulting in improved performance for various video object segmentation related tasks.

In parallel to the aforementioned developments in the video domain, another research area in computer vision was focusing on improving the performance of image-level segmentation networks. To this end, one line of reasoning considers the limited receptive field of convolution operations as a drawback and aims to mitigate it. Even though a restricted receptive field is useful for weight sharing and imparting translation invariance, it is also a limitation for dense segmentation tasks where a wider view of the feature map can be beneficial. Chen *et al*. published a series of works [7, 8, 9] that use *atrous convolutions* (also called *dilated convolutions*) for semantic segmentation in images. Dilated convolutions effectively add padded zeros between the values in the convolutional kernel, thus enlarging the receptive field without incurring computational overhead or increasing the parameter count. Chen *et al*. argued that the high degree of spatial downsampling usually applied in CNNs is detrimental for dense segmentation tasks. They instead maintained feature maps at a higher resolution and used dilated convolutions to capture a larger receptive field.

Another method for enhancing the receptive field of convolutions is the idea of *deformable convolutions* [11] (DCNv1). Here, the convolutional kernel can be arbitrarily shaped depending on the input feature map, as opposed to being a regular grid as in standard or dilated convolutions. Practically, this is realized by using the input feature map to predict offsets (or *deformations*) to the sampling locations

of the convolution operation. The underlying idea here is to enable the network to dynamically adapt the kernel based on the input image. Zhu *et al*. [43] further extended this by adding a dynamic *modulation parameter* which scales the kernel weight value for each sampling location (DCNv2). By simply using deformable convolutions as a drop-in replacement for standard convolutions, it was shown that the performance of a variety of network architectures could be improved for object detection and segmentation.

Keeping these developments in mind gives rise to a question: Can 3D dilated/deformable convolutions retrace the success story of their 2D counter-parts and deliver improvements for video segmentation tasks? In this paper, we show that the answer is 'yes'. To this end, we propose a novel type of convolution called D^2Conv3D (**D**ynamic **D**ilated **3D Conv**olutions), which combines elements from dilated and deformable convolutions by dynamically learning a multiplicative scaling factor for the sampling locations of a convolutional kernel. Additionally, we also a predict a modulation parameter which dynamically scales the kernel weights based on the input features. We show that D^2Conv3D outperforms trivial extensions of dilated and deformable convolutions to 3D. Fig. 1 provides an illustrative comparison between: (i) standard 3D convolutions, (ii) a 3D extension of the modulated deformable convolutions proposed by Zhu *et al*. [43], (iii) our proposed D^2Conv3D.

In summary, our contributions are as follows:

- We propose a novel D^2Conv3D operator which can be used as drop-in replacements for standard convolutions in 3D CNNs to improve their performance on video segmentation tasks.
- We experimentally justify the efficacy of D^2Conv3D by applying it to two different 3D CNN based architectures [1, 24] and evaluating them on five different benchmarks [25, 6, 37, 34, 28].
- We set a new state-of-the-art on the DAVIS 2016 Unsupervised challenge [25] by achieving a $\mathcal{J}\&\mathcal{F}$ score of 86.0%.

2. Related Work

Image-level Segmentation: Dense prediction tasks such as segmentation need to predict full resolution output maps and, at the same time, use multi-scale context for effective reasoning. Existing approaches for such tasks [1, 7, 8, 9, 15, 39] utilize dilated convolutions for this purpose, which dilate the convolutional kernel by a fixed factor to increase the receptive field, thus mitigating the need for downsampling the image features. Atrous Spatial Pyramid Pooling (ASPP) [7] goes a step further by using multiple dilation rates on the same feature map to capture a multi-scale feature representation, and has been successfully used for both instance and semantic segmentation tasks [1, 15].

Although dilated convolutions and ASPP help capture objects of different sizes, the convolutional kernels have fixed geometric structures since the dilation rates are constant. Several existing works [11, 12, 43] attempt to adapt these kernels by learning offsets or other transformation parameters from the image features. Spatial Transformer Networks (STN) [17] learn deformations of the sampling grid, for a regular convolution operation, from the input feature map and warp the sampling grid based on the learnt deformation parameters. Deformable Convolutional Network (DCNv1) [11] on the other hand apply learned offsets to the sampling locations of a regular convolution, thereby enhancing its capability of capturing non-rigid transformations. DCNv1 can adapt to varying object sizes and scene geometry, and is shown to be effective for image-level tasks such as object detection and segmentation [11]. Nevertheless, the sampling locations learned by DCNv1 often spread beyond the region of interest, which can lead to unnecessary feature influences. To overcome this issue, Zhu *et al*. introduced DCNv2 [43] where, in addition to the offsets, a dynamic modulation parameter is learned which scales the kernel weights. This parameter gives the convolution kernels additional freedom to adjust the influence of the sampled regions. A teacher network based on R-CNN [14] is then used to train this modulation mechanism, where the teacher provides additional guidance to learn a more focused feature representation.

D^2Conv3D, similar to deformable convolutions [11, 43], can be directly plugged-in to any existing architecture and improve its performance. However, unlike deformable convolutions, D^2Conv3D works with 3D models and can be used effectively for segmentation tasks in videos. In addition, the modulation mechanism used in D^2Conv3D does not need additional supervision from a teacher network as in DCNv2 [43].

Video Processing using 3D Convolutions: Videos can be interpreted as 3D data with the third dimension being time. To leverage temporal context effectively, video classification networks [18, 19, 30, 33] successfully use 3D-CNNs and show their superior performance. However, unlike segmentation tasks, these networks do not need large resolution feature maps, and hence the increase in computational overhead caused by 3D-CNNs is still manageable. Recent works in the field of Unsupervised Video Object Segmentation [16, 24], which target *foreground-background* segmentation, have also adapted 3D-CNNs to improve the segmentation performance. Hou *et al*. [16] uses an encoder-decoder architecture based on a variant of 3D-CNNs called R2plus1D [32], and insert an ASPP after the last layer of the encoder. However, they adopt a relatively shallow network to compensate for the additional computation needed by ASPP, which in turn affects the final performance. Mahadevan *et al*. [24] on the other hand employ a much deeper

channel-separated 3D-CNN [31] as backbone with much fewer parameters in combination with their novel 3D Global Convolutions and 3D Refinement modules in the decoder, and achieve state-of-the-art results. In this paper, we show that augmenting [24] with D^{2}Conv3D further improves the network performance even with significantly less training data.

Instance Segmentation in Videos: Multi-instance Segmentation in Videos has recently emerged as a popular field due to its applicability in autonomous driving and robotics. Some of the popular tasks in this domain are Video Object Segmentation (VOS) [6, 25], Video Instance Segmentation (VIS) [37], and the more recent Occluded Video Instance Segmentation (OVIS) [28]. Here the primary goal is to segment all object instances in a video and associate them over time. For VIS and OVIS, there is an additional task of classifying the predicted tracks into one of the predefined object categories. Multi Object Tracking and Segmentation (MOTS) [34] is another similar task that focuses on autonomous driving scenes and requires segmenting and tracking cars and pedestrians in these scenarios.

Popular methods [37, 34, 44, 40, 23] that tackle the aforementioned tasks typically first generate image-level instance proposals, and then associate them either by using multiple cues such as optical flow and re-id, or by learning some kind of pixel affinity based on attention mechanism [44, 23]. Most of these methods employ 2D dilated convolutions in the backbone, and process each frame separately, thereby not effectively making use of the larger temporal context. Bertasius *et al*. recently proposed MaskProp[4], which modifies Mask R-CNN [15] with a mask propagation branch to adapt it to videos The features from the middle frame of an input video clip are aligned with the remaining frames using learnt spatial offsets similar to [5]. Unlike D^{2}Conv3D, the mask propagation branch in MaskProp only operates on two frames, and it does not use learnt temporal dilation. STEm-Seg [1] is another architecture that is relevant to our work. It processes an input video clip and generates spatio-temporal embeddings that can be directly clustered to obtain temporally consistent instance segmentation masks. STEm-Seg is a bottom-up approach, and uses a decoder comprising of entirely 3D convolutions for this purpose. In this paper, we show that by plugging in D^{2}Conv3D to just the decoder of STEm-Seg further improves its performance.

3. Method

Our proposed D^{2}Conv3D predicts a dilation scaling factor and a modulation value for every pixel in the input feature map. Before explaining this in detail, we will first briefly recap the details of existing 2D deformable convolutions [11, 43] in Sec. 3.1:

3.1. Deformable Convolutions in 2D

Let $\mathbf{X} \in \mathbb{R}^{H \times W}$ denote the feature map for an image with resolution $H \times W$ (we ignore the channel dimension for ease of notation). Let $\mathbf{X}(\mathbf{p})$ denote the value of $\mathbf{X}$ at coordinates $\mathbf{p} = (p_y, p_x)$. Furthermore, let $\mathbf{W}$ denote the weights of a given convolutional kernel with K entries, and let $\mathcal{S} \in \mathbb{R}^{K \times 2}$ denote the sampling region for the convolution. E.g. for a standard 3×3 convolution, $K = 9$, and the sampling region $\mathcal{S} = \{(-1,-1), (-1,0), ..., (1,0), (1,1)\}$.

In deformable convolutions [11], this sampling region is shifted by a set of offsets predicted for every point in $\mathbf{X}$ which we denote with $\Delta S \in \mathbb{R}^{H \times W \times K \times 2}$. Zhu *et al*. [43] additionally also predict a set of modulation parameters $\mathbf{M} \in \mathbb{R}^{H \times W \times K}$. If we let $\mathbf{Y}$ denote the feature map obtained after applying the deformable convolution, then the value of $\mathbf{Y}$ at coordinates $\mathbf{p_0}$ is calculated as follows:

$$\mathbf{Y}(\mathbf{p}_0) = \sum_{\mathbf{p}_n \in \mathcal{S}} \mathbf{M}(\mathbf{p}_0, \mathbf{p}_n) \cdot \mathbf{W}(\mathbf{p}_n) \cdot \mathbf{X}(\mathbf{p}_0 + \mathbf{p}_n + \Delta S(\mathbf{p}_0, \mathbf{p}_n)) \quad (1)$$

Here, $\mathbf{M}(\mathbf{p}_0, \mathbf{p}_n)$ and $\Delta S(\mathbf{p}_0, \mathbf{p}_n)$ are used to denote the modulation value and sampling offset, respectively, predicted for point $\mathbf{p}_0$ in $\mathbf{X}$ at sampling location $\mathbf{p}_n$ in the kernel. Thus, deformable convolutions are able to dynamically attend to spatial locations outside of the fixed sampling region $\mathcal{S}$ which standard convolutions are bound to follow.

Note that Eq. 1 is a generalization of the standard convolutions operation: if $\Delta S(\mathbf{p}_0, \mathbf{p}_n) = 0$ and $\mathbf{M}(\mathbf{p}_0, \mathbf{p}_n) = 1$, the deformable convolution reduces to a standard convolution. The modulation parameters in $\mathbf{M}$ are sigmoid activated and thus lie in the range $[0, 1]$, but the offsets in ΔS are unconstrained and may be fractional. Therefore, bilinear interpolation is applied to sample the input feature map [11, 43]. Points outside the feature map are assumed to have a value of 0.

3.2. D^{2}Conv3D - Dynamic Dilated Convolutions

The fixed grid structure of convolutions imposes useful inductive bias for computer vision tasks due to the regular grid structure of rasterized images. However, convolutional filters cannot adapt to changes in the underlying geometry of a scene. By contrast, deformable convolutions can adapt to changes in scene geometry, but lose the inductive bias imposed by the fixed grid structure of a standard convolution kernel.

For video tasks, the size of the temporal dimension of an input spatio-temporal volume is usually orders of magnitude smaller than the spatial dimensions. This means that a trivial extension of DCN [11, 43] to the 3D (spatio-temporal) domain often results in sampling locations that lie outside the spatio-temporal volume of the feature map.

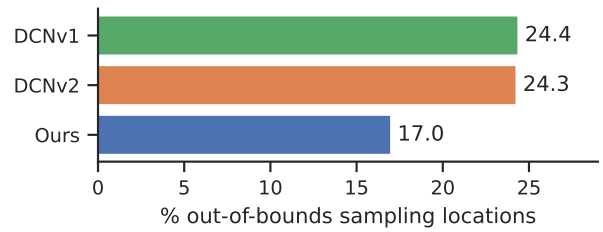

Figure 2: Average percentage of sampling locations outside the input volume during inference on DAVIS'16.

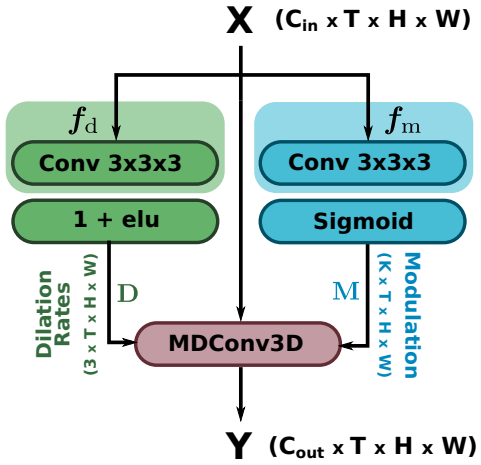

Figure 3: Architecture of the D^{2}Conv3D block. f_d and f_m are the names of the convolution layers which produce $\mathbf{D}$ and $\mathbf{M}$, respectively.

D^{2}Conv3D on the other hand, strikes a compromise between these two types of convolutions: it maintains the grid structure of a convolutional kernel, but allows the kernel to be dilated dynamically and independently along each of the three dimensions. Furthermore, compared to DCNv1 and DCNv2, sampling locations predicted by D^{2}Conv3D are better aligned with the input feature. This is shown in Fig. 2, where it can be seen that D^{2}Conv3D generates far less out of bounds sampling locations. We refer to the supplementary material for further analysis of out-of-bounds sampling behaviour.

Architecture. In the 3D spatio-temporal domain of temporal dimension T, features can be redefined as $\mathbf{X} \in \mathbb{R}^{T\times H\times W}$, the sampling region of a convolution as $\mathcal{S} \in \mathbb{R}^{K\times 3}$, and the point coordinates $\mathbf{p} = (p_t, p_y, p_x)$. E.g. a $3 \times 3 \times 3$ convolution has $K = 27$ entries.

Our proposed D^{2}Conv3D is a novel type of convolution that keeps the grid structure of normal convolutions while applying learnt dilations, and can be applied to spatio-temporal features. In contrast to deformable convolutions [11, 43] that learn K additive offsets for the sampling region, we learn 3 multiplicative factors, one each for the (t, y, x) dimensions, and apply them to the coordinates in the sampling region $\mathcal{S}$. D^{2}Conv3D can thus be seen as dilated convolutions with dynamically learned dilation rates. We will henceforth use the term 'dilation map' to refer to the set of dilated rates predicted for $\mathbf{X}$, *i.e.* $\mathbf{D} \in \mathbb{R}^{T\times H\times W\times 3}$.

To predict the dilation map, we input feature map $\mathbf{X}$ to a standard $3 \times 3 \times 3$ convolution f_d followed by an elu activation function [10] and addition by 1:

$$\mathbf{D} = 1 + \mathrm{elu}(f_\mathrm{d}(\mathbf{X})) \tag{2}$$

This forces the values in $\mathbf{D}$ to lie in the range $[0, \infty)$. We found that simply applying a ReLU activation to $f_\mathrm{d}(\mathbf{X})$ frequently results in zero gradients. By contrast, the formulation in Eq. 2 produces more well-behaved gradients during training, and also better results during inference (see Sec. 4.4).

Separately, a $3 \times 3 \times 3$ convolution f_m is applied to $\mathbf{X}$ followed by sigmoid activation to produce the modulation map $\mathbf{M} \in \mathbb{R}^{T\times H\times W\times K}$ (*c.f.* [43]). The value of the output feature map $\mathbf{Y}$ at point $\mathbf{p}_0$ with D^{2}Conv3D is then calculated as follows:

$$\mathbf{Y}(\mathbf{p}_0) = \sum_{\mathbf{p}_n\in\mathcal{S}} \mathbf{M}(\mathbf{p}_0, \mathbf{p}_n) \cdot \mathbf{W}(\mathbf{p}_n) \cdot \mathbf{X}(\mathbf{p}_0 + (\mathbf{p}_n \cdot \mathbf{D}(\mathbf{p}_0))) \tag{3}$$

Here, with some abuse of notation, we use $\mathbf{p}_n \cdot \mathbf{D}(\mathbf{p}_0)$ to denote the multiplication of the sampling location coordinates $\mathbf{p}_n$ with the tuple of 3 dilation rates in $\mathbf{D}$ at point $\mathbf{p}_0$.

Fig. 3 illustrates the architecture of a D^{2}Conv3D block, which comprises the actual D^{2}Conv3D layer, and also the two layers and activations required to produce $\mathbf{M}$ and $\mathbf{D}$. Note that, similar to Eq. 1, D^{2}Conv3D is also a generalization of the convolution operation - if the modulation parameter and dilation rates are unity, D^{2}Conv3D reduces to a standard convolution. Moreover, it can also specialize to a 2D, 1D, or point-wise convolution by predicting one or more dilation rates as zero. It is therefore possible to use an D^{2}Conv3D block as a drop-in replacement for 3D convolutions in existing pre-trained networks without modifying their behavior at the start of training. This can be done by simply copying the existing kernel weights to the D^{2}Conv3D layer, and initializing the weights and bias parameters in f_d and f_m with zeros.

3.3. Qualitative Analysis

The dilation rates and modulation values predicted by D^{2}Conv3D when it is used as an intermediate layer in a video object segmentation network are illustrated in Fig. 4 and Fig. 5. The visualizations shows the mean dilation rates (second row) and mean modulation values (third row). It is evident that the network learns to use different dilation rates

Figure 4: Qualitative results on DAVIS'16 [25]. Depicted are results from our model (top row), mean predicted dilation factors (middle row), and mean predicted modulation values. Lighter pixels denote higher values. Dilation factors are highest on the object, while modulation values are highest on the object boundary. Left sequence is *dance-twirl*, right sequence is *horsejump-high*.

to distinguish the foreground object from the background. Dilation rates are highest inside the object, medium on the object borders, and zero in the background. The modulation values, on the other hand, are highest on the object boundaries. This indicates that the network mostly focuses on refining the edges of the segmentation mask.

4. Experiments on Video Saliency

To justify the efficacy of D^{2}Conv3D, we apply it as a drop-in replacement for convolution layers in the 3D CNN architecture proposed by Mahadevan *et al*. [24]. Subsequently, we conduct several ablations and also compare our results to existing state-of-the arts. The experimental evaluation is performed on the validation set of the DAVIS'16 Unsupervised Video Object Segmentation benchmark [25]. The task here is to perform foreground segmentation of the 'salient' regions of the video. Here, saliency is subjectively defined as regions which undergo motion changes significant enough to capture the attention of the human eye. Note that even though the word 'unsupervised' occurs in the benchmark name, this is a fully supervised task where labeled training data is provided.

4.1. Network Architecture

We build on the baseline network architecture proposed by Mahadevan *et al*. [24] which has a compact encoder-decoder architecture composed entirely of 3D convolutions. It accepts an input video clip and outputs a foreground probability map for each pixel in the input clip. The encoder is an efficient channel-separated 3D variant of ResNet-152 [31] from which feature maps are extracted at four different spatially downsampled scales (4x, 8x, 16x, 32x). The decoder is made up of three so-called *Refinement modules* which upsample the given input feature map and combine it with the encoder feature map at the corresponding scale. To process arbitrarily long videos, the input video is split into multiple 8-frame clips with an overlap of 3 frames between successive clips. The output probability maps for the overlapping frames are subsequently averaged to get the final pixel foreground probabilities. We refer the reader to [24] for more details of the baseline.

Our Changes. We replace all convolution layers in the first two *Refinement modules* with D^{2}Conv3D. These two modules are the ones which upsample the current feature map and combine it with the encoder features at the 16x and 8x spatially downsampled scales. We will henceforth refer to these modules as `rf1` and `rf2`.

Moreover, we apply GroupNorm [36] to the outputs of the convolution layers in order to improve the gradient flow to deeper network layers. It should be noted that D^{2}Conv3D is slower than a standard convolution, so it is infeasible to apply it to high resolution feature maps (*e.g*. at the 4x spatial scale). We chose `rf1` and `rf2` as the replacement locations since this yields a good trade-off between performance and speed. We found that replacing layers in the encoder only provides a minor improvement that does not justify the increased computation/memory overhead.

4.2. Training

We initialize the encoder with weights from a publicly available model which is trained for video action classification on Kinetics400 [20] and Sports-1M [19]. The weights of layers which predict offsets and modulation parameters are initialized to zero. All other weights in the decoder are initialized randomly. The network is trained for 20 epochs on 8-frame clips sampled randomly from the DAVIS'17 [27] training set without any data augmentation. We use the Lovasz-Hinge loss [2] and train the network using using the Adam optimizer [21] with a learning rate of 10^{-5}, which is reduced by a factor of 0.1 after 10 epochs. Additionally, we employ gradient clipping by limiting the L2-norm of the gradient to a maximum value of 10.

Variant	Mod.	$\mathcal{J}\&\mathcal{F}$	FPS	Mem. (GB)
Baseline [24]	–	82.2	–	–
Revised Baseline	–	83.5	4.96	1.92
ASPP	–	82.6	4.50	1.96
DCNv1 [11]	✗	84.4	4.41	1.95
DCNv2 [43]	✓	84.8	4.36	1.96
(I) S ($\|\mathbf{D}\| = 1$)	✓	84.2	4.49	1.94
(II) S ($\|\mathbf{D}\| = 2$)	✓	84.3	4.46	1.94
(III) S+T ($\|\mathbf{D}\| = 3$)	✗	85.0	4.46	1.93
(IV) S+T ($\|\mathbf{D}\| = 3$)	✓	**85.5**	4.46	1.94

Table 1: Ablation on offset modality. Runtime and memory were measured during inference on an NVIDIA GTX1080ti. Mod: Modulation, S: spatial, S+T: spatio-temporal

4.3. Ablations on Deformations

We report the results of our first set of ablations in Tab. 1. The most comparable baseline setting from [24] which uses the same (pre-)training data as we do achieves a $\mathcal{J}\&\mathcal{F}$ score of 82.2. For a fair comparison, we re-train this baseline with our training settings and added normalization layers. This 'Revised Baseline' achieves a $\mathcal{J}\&\mathcal{F}$ score of 83.5.

ASPP. Here, we replace the convolution layers in `rf1` and `rf2` with ASPP blocks [7], which comprise multiple dilated convolutions in parallel, but with a fixed dilation rate. Doing so actually reduces the $\mathcal{J}\&\mathcal{F}$ score from 83.5 to 82.6.

Deformable Convolutions. Next, we use 3D extensions of existing deformable convolutions in `rf1` and `rf2`. Using DCNv1 [11] improves the $\mathcal{J}\&\mathcal{F}$ by 0.9% over the revised baseline (83.5 $\rightarrow$ 84.4), while using DCNv2 [43] provides a 1.3% improvement (83.5 $\rightarrow$ 84.8). This clearly shows that dynamically adapting the convolution sampling locations is beneficial for video segmentation tasks.

D^{2}Conv3D Variants. Finally, we replace the convolutions in `rf1` and `rf2` with variants of dynamic dilated convolutions. In (I), a single dilation rate is predicted for the two spatial dimensions, fixing the temporal dilation to 1. This improves the $\mathcal{J}\&\mathcal{F}$ from 83.5 in the revised baseline to 84.2, but still lags behind DCNv1 and DCNv2. In (II) we allow different dilation rates for the two spatial dimensions, but the temporal dilation remains fixed. Doing so yields an insignificant improvement of 0.1% $\mathcal{J}\&\mathcal{F}$ over variant (I).

In variant (III), we predict separate dilation rates for the 3 spatio-temporal dimensions, but do not predict modulation parameters. This variant achieves 85.0 $\mathcal{J}\&\mathcal{F}$ which is 1.5% higher than the revised baseline (83.5), and 0.6% higher than DCNv1 variant (84.4) which also does not use modulation parameters. Finally, variant (IV) is the full D^{2}Conv3D with modulation parameters (as explained in Sec. 3.2). This achieves a $\mathcal{J}\&\mathcal{F}$ score of 85.5 which is 2% higher than the revised baseline, and 0.7% higher than the second-best performing DCNv2 variant. The fact that variant (IV) achieves a 1.8% higher $\mathcal{J}\&\mathcal{F}$ compared to variant (II) shows the effectiveness of the dynamic temporal dilation rate predicted by the network.

Activation	None	1 + ReLU	ReLU	1 + elu
$\mathcal{J}\&\mathcal{F}$	83.8	83.9	85.1	**85.5**

Table 2: Ablations on dilation map activation function.

In light of these results, we can see that for video segmentation tasks, D^{2}Conv3D out-performs 3D extensions of existing deformable convolutions despite having fewer parameters (our dilation map $\mathbf{D}$ has 3 channels, whereas the offset maps for DCNv1 and DCNv2 have 81 channels for a $3 \times 3 \times 3$ convolution). This justifies our earlier claim that D^{2}Conv3D allow the network to adapt to geometric scene variations in the input features while retaining the useful inductive bias associated with the grid shaped structure of a convolutional kernel.

4.4. Ablations on Dilation Activation Function

Tab. 2 examines the effect of using different activation functions to predict the dilation map $\mathbf{D}$. Using no activation achieves a $\mathcal{J}\&\mathcal{F}$ score of 83.8. Note that this setting allows the network to predict a negative dilation rate, which mirrors the convolution kernel along that dimension. '1 + ReLU' restricts the range of dilations to $[1, \infty)$. Looking at Fig. 4, we see that the network often specializes D^{2}Conv3D to a point-wise convolution for background regions by predicting very small dilation rates (close to zero). Since '1 + ReLU' disallows such behavior, the $\mathcal{J}\&\mathcal{F}$ reduces by 1.2% to 83.9 compared to the 85.1 achieved by just applying a ReLU. Finally, our chosen '1 + elu' activation (85.5 $\mathcal{J}\&\mathcal{F}$) out-performs ReLU (85.1 $\mathcal{J}\&\mathcal{F}$) by 0.4% due to improved gradient flow for small dilation rates.

4.5. Comparison with State of the Art

Tab. 3 shows the results of our method (baseline [24] with D^{2}Conv3D) in comparison to existing state-of-the-art approaches on the validation set of the DAVIS'16 Unsupervised Video Object Segmentation. Our method achieves 85.5 $\mathcal{J}\&\mathcal{F}$ using a clip overlap of 3 frames during inference. If this overlap is increased to 7 frames, the $\mathcal{J}\&\mathcal{F}$ increases to 86.0 at the cost of slower run-time.

Our method out-performs the current state-of-the-art RT-Net [29] (85.2 $\mathcal{J}\&\mathcal{F}$) by 0.8% even though RTNet uses CRF post-processing and optical flow as an external cue. In fact, almost all other works either train on significantly more data [1, 24], or use additional performance improvement cues such as optical flow [29], CRF postprocessing [35, 22, 29] or other heuristics [38, 41]. By contrast, we use only the DAVIS'17 dataset for training and use no other

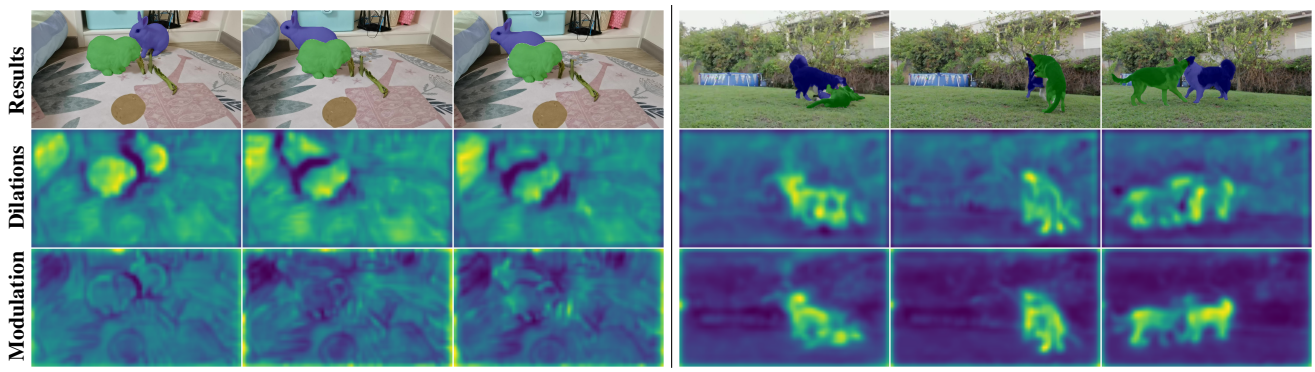

Figure 5: Qualitative results on OVIS [28]. Depicted are results from our model (top row), mean predicted dilation factors (middle row), and mean predicted modulation values. Left side: embedding decoder. Right side: semantic segmentation decoder.

Method	$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}$-mean	$\mathcal{F}$-mean
3D-CNN [16]	77.8	78.3	77.2
AGNN*‡ [35]	79.9	80.7	79.1
COSNet‡ [22]	80.0	80.5	79.5
STEm-Seg* [1]	80.6	80.6	80.6
ADNet*§ [38]	81.1	81.7	80.5
MATNet† [42]	81.6	82.4	80.7
DFNet*§ [41]	82.6	83.4	81.8
3DC-Seg** [24]	84.5	84.3	84.7
RTNet†‡ [29]	85.2	**85.6**	84.7
Ours	85.5	84.9	86.0
Ours (dense)	**86.0**	85.5	**86.5**

Table 3: Quantitative results on the DAVIS'16 unsupervised validation set. †Optical flow ‡CRF post-processing *Multi-scale inference **large-scale pretraining §heuristic post-processing

external cues, augmentations or post-processing techniques. Also note that our $\mathcal{F}$ score (86.5) is significantly higher than the second-highest (84.7), indicating that D^{2}Conv3D can produce highly accurate object boundaries.

5. Multi-Instance Segmentation in Video

To show the generalization capability of D^{2}Conv3D, we evaluate on four other benchmarks involving multi-instance segmentation in videos.

5.1. Network Architecture

We extend the network architecture of STEm-Seg [1], which is a single-stage approach that segments multiple object instances by clustering per-pixel embeddings in a given input video clip. STEm-Seg also has an encoder-decoder architecture, but unlike [24], it only has 3D convolutions in the decoder. For our experiments, we replace the convolution layers in the two 'deepest' blocks of the decoder (which process the 32x and 16x down-sampled feature maps) with D^{2}Conv3D. Furthermore, in order to reduce training time, we use a lighter ResNet-50 encoder backbone compared to the ResNet-101 used in the original paper [1]. All other details, including the training schedule, are kept identical.

5.2. Benchmarks

We compare the performance of STEm-Seg enhanced with D^{2}Conv3D on four popular, challenging benchmarks for multi-instance segmentation in videos. These are briefly described below:

KITTI-MOTS. KITTI-MOTS [34] is an extension of the popular KITTI dataset for Multi-Object Tracking (MOT) [13] which requires pixel-precise object masks as opposed to bounding boxes, hence the name MOTS (Multi-Object Tracking and Segmentation). It contains 21 lengthy videos captured from a moving vehicle wherein the task is to segment and track all *car* and *pedestrian* object instances. Performance is primarily assessed using the 'sMOTSA' measure [34], which is an extension of the CLEAR MOT metrics [3] to account for pixel-precise segmentation masks.

DAVIS'19 Unsupervised. The DAVIS 2019 Unsupervised Video Object Segmentation benchmark [6] requires all salient objects in the video to be segmented and tracked over time. The training and validation sets comprise 60 and 30 videos, respectively. Similar to the DAVIS'16 Unsupervised benchmark, the evaluation metrics here are the $\mathcal{J}$ and $\mathcal{F}$ scores, which are averaged into a single $\mathcal{J}\&\mathcal{F}$ metric. For this benchmark however, the $\mathcal{J}\&\mathcal{F}$ is computed separately for each object instance.

YouTube-VIS. The YouTube Video Instance Segmentation dataset [37] consists of 2,883 videos with a total of more than 130k object instances. Here, in addition to segmenting and tracking object instances over time, a class label (from one of 40 known classes) also has to be assigned to each predicted instance. The evaluation measure is mean Average Precision (mAP).

Conv Type	Car				Pedestrian			
	sMOTSA	MOTSA	MOTSP	IDS	sMOTSA	MOTSA	MOTSP	IDS
Baseline	66.6	76.6	87.5	67	38.5	54.4	**77.1**	33
DCNv1	68.7	78.8	**87.7**	79	35.4	51.1	76.0	54
DCNv2	67.5	77.7	87.6	86	38.7	54.2	76.5	**28**
D^{2}Conv3D	**69.7**	**80.0**	87.6	**66**	**42.6**	**58.3**	76.7	46

Table 4: Performance improvements on KITTI MOTS. Baseline is STEm-Seg [1] with a ResNet50 backbone.

Conv Type	$\mathcal{J}\&\mathcal{F}$	$\mathcal{J}$-mean	$\mathcal{F}$-mean
Baseline	63.4	60.3	66.5
DCNv1 [11]	63.2	59.7	66.6
DCNv2 [43]	**64.6**	**61.0**	68.2
D^{2}Conv3D	**64.6**	60.8	**68.5**

Table 5: Performance improvements on DAVIS'19. Baseline is STEm-Seg [1] with a ResNet50 backbone.

Dataset	Conv Type	mAP	AP50	AP75	AR1	AR10
YVIS	Baseline	30.6	50.7	33.5	31.6	37.1
	DCNv1 [11]	31.7	50.8	34.0	31.9	37.8
	DCNv2 [43]	29.4	48.1	31.8	30.4	36.1
	D^{2}Conv3D	**32.3**	**51.3**	**34.7**	**32.2**	**38.1**
OVIS	Baseline	14.3	31.5	12.4	10.2	20.7
	DCNv1 [11]	**15.9**	**34.0**	13.2	**10.8**	**22.4**
	DCNv2 [43]	14.9	31.6	**13.8**	10.5	21.5
	D^{2}Conv3D	15.2	33.8	13.7	10.6	22.2

Table 6: Performance improvements on YoutubeVIS (YVIS) and OVIS. Baseline is STEm-Seg [1] with a ResNet50 backbone.

OVIS. Occluded Video Instance Segmentation [28] comprises 5,233 videos with labeled masks for 25 known object classes. The dataset is similar to YouTube-VIS in that it also uses mean Average Precision (mAP) as the evaluation measure, but is more challenging since it comprises longer videos where objects undergo significant occlusion.

5.3. Results

We compare the results of STEm-Seg with D^{2}Conv3D against the original baseline [1], and also against the case where 3D extensions of DCNv1 [11] and DCNv2 [43] are used instead of D^{2}Conv3D. On all four benchmarks, using D^{2}Conv3D in the decoder improves the results over the baseline, whereas DCNv1 and DCNv2 perform inconsistently, sometimes even degrading performance.

On the MOTS task (Tab. 4), using D^{2}Conv3D leads to a significant performance increase. For the *car* class, DCNv1 and DCNv2 improve the sMOTSA score over the baseline by 2.1 and 0.9, respectively, whereas D^{2}Conv3D yields a more profound improvement of 3.1. For the *pedestrian* class, D^{2}Conv3D improves the sMOTSA score over the baseline by 4.1 (38.5 $\rightarrow$ 42.6), whereas DCNv1 actually reduces the sMOTSA by 3.1 (38.5 $\rightarrow$ 35.4) and DCNv2 yields only a minor 0.2 improvement.

On DAVIS'19 (Tab. 5), D^{2}Conv3D improves the $\mathcal{J}\&\mathcal{F}$ to 64.6, which is 1.2% higher than the baseline (63.4). The 3D extension of DCNv2 achieves similar performance, while DCNv1 performs slightly worse than the baseline. Both D^{2}Conv3D and DCNv2 significantly improve the $\mathcal{F}$ measure, which indicates that modulating kernel weights improves the networks ability to accurately predict the contours of object instances.

On YoutubeVIS and OVIS (Tab. 6), D^{2}Conv3D improves over the baseline by 1.7% and 0.9% mAP, respectively. This indicates that D^{2}Conv3D improves segmentation quality for complex scenes with several occluded objects. DCNv1 improves result on YouTube-VIS by 1.1% mAP and achieves an even greater improvement on OVIS, where it boosts the mAP from 14.3 to 15.9, outperforming D^{2}Conv3D by 0.7%. On the other hand, DCNv2 improves over the baseline on OVIS, but performs worse on YoutubeVIS. We note that our training schedule for OVIS may be sub-optimal since OVIS was not evaluated in the original paper [1] and we simply used the same training setup and hyper-parameters as for YouTube-VIS.

6. Conclusion

In this paper, we presented D^{2}Conv3D, a new type of dynamic 3D convolution, and justified its efficacy by applying it to two different network architectures and five different video segmentation tasks. Furthermore, we showed that D^{2}Conv3D out-performs 3D extensions of existing deformable convolutions because it experiences fewer out-of-bounds sampling locations and preserves the useful inductive bias associated with rectangular convolutional kernels. Future work could explore shifting the kernel center position, and additional types of kernel shape deformation.

Acknowledgements. This project was funded, in parts, by ERC Consolidator Grant DeeVise (ERC-2017-COG-773161), EU project CROWDBOT (H2020-ICT-2017-779942). Computing resources for several experiments were granted by RWTH Aachen University under project 'thes0863'. We thank Paul Voigtlaender, István Sárándi, Jonas Schult and Alexey Nekrasov for helpful discussions.

References

[1] Ali Athar, Sabarinath Mahadevan, Aljoša Ošep, Laura Leal-Taixé, and Bastian Leibe. Stem-seg: Spatio-temporal embeddings for instance segmentation in videos. In *ECCV*, 2020.

[2] Maxim Berman, Amal Rannen Triki, and Matthew B Blaschko. The lovász-softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks. In *CVPR*, 2018.

[3] Keni Bernardin and Rainer Stiefelhagen. Evaluating multiple object tracking performance: the clear mot metrics. *JIVP*, 2008.

[4] Gedas Bertasius and L. Torresani. Classifying, segmenting, and tracking object instances in video with mask propagation. In *CVPR*, 2020.

[5] Gedas Bertasius, L. Torresani, and Jianbo Shi. Object detection in video with spatiotemporal sampling networks. In *ECCV*, 2018.

[6] Sergi Caelles, Jordi Pont-Tuset, Federico Perazzi, Alberto Montes, Kevis-Kokitsi Maninis, and Luc Van Gool. The 2019 davis challenge on vos: Unsupervised multi-object segmentation. *arXiv*, 2019.

[7] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *PAMI*, 2017.

[8] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

[9] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018.

[10] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv*, 2015.

[11] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *ICCV*, 2017.

[12] Mingyu Ding, Yuqi Huo, Hongwei Yi, Zhe Wang, Jianping Shi, Zhiwu Lu, and P. Luo. Learning depth-guided convolutions for monocular 3d object detection. In *CVPR Workshops*, 2020.

[13] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012.

[14] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

[15] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN. In *ICCV*, 2017.

[16] Rui Hou, Chen Chen, Rahul Sukthankar, and Mubarak Shah. An efficient 3d CNN for action/object segmentation in video. In *BMVC*, 2019.

[17] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and koray kavukcuoglu. Spatial transformer networks. In *NIPS*, 2015.

[18] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. *PAMI*, 2012.

[19] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *CVPR*, 2014.

[20] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv*, 2017.

[21] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *ICLR*, 2015.

[22] Xiankai Lu, Wenguan Wang, Chao Ma, Jianbing Shen, Ling Shao, and Fatih Porikli. See more, know more: Unsupervised video object segmentation with co-attention siamese networks. In *CVPR*, 2019.

[23] Jonathon Luiten, Paul Voigtlaender, and Bastian Leibe. Premvos: Proposal-generation, refinement and merging for video object segmentation. In *ACCV*, 2018.

[24] Sabarinath Mahadevan, Ali Athar, Aljoša Ošep, Sebastian Hennen, Laura Leal-Taixé, and Bastian Leibe. Making a case for 3d convolutions for object segmentation in videos. *BMVC*, 2020.

[25] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *CVPR*, 2016.

[26] Pedro O Pinheiro, Tsung-Yi Lin, Ronan Collobert, and Piotr Dollár. Learning to refine object segments. In *ECCV*, 2016.

[27] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alexander Sorkine-Hornung, and Luc Van Gool. The 2017 davis challenge on video object segmentation. *arXiv*, 2017.

[28] Jiyang Qi, Yan Gao, Yao Hu, Xinggang Wang, Xiaoyu Liu, Xiang Bai, Serge Belongie, Alan Yuille, Philip HS Torr, and Song Bai. Occluded video instance segmentation. *arXiv preprint arXiv:2102.01558*, 2021.

[29] Sucheng Ren, Wenxi Liu, Yongtuo Liu, Haoxin Chen, Guoqiang Han, and Shengfeng He. Reciprocal transformations for unsupervised video object segmentation. In *CVPR*, 2021.

[30] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *ICCV*, 2015.

[31] Du Tran, Heng Wang, Lorenzo Torresani, and Matt Feiszli. Video classification with channel-separated convolutional networks. In *ICCV*, 2019.

[32] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *CVPR*, 2018.

[33] Gül Varol, Ivan Laptev, and Cordelia Schmid. Long-term temporal convolutions for action recognition. *PAMI*, 2017.

[34] Paul Voigtlaender, Michael Krause, Aljosa Osep, Jonathon Luiten, Berin Balachandar Gnana Sekar, Andreas Geiger,

and Bastian Leibe. Mots: Multi-object tracking and segmentation. In *CVPR*, 2019.

[35] Wenguan Wang, Xiankai Lu, Jianbing Shen, David J Crandall, and Ling Shao. Zero-shot video object segmentation via attentive graph neural networks. In *ICCV*, 2019.

[36] Yuxin Wu and Kaiming He. Group normalization. In *ECCV*, 2018.

[37] Linjie Yang, Yuchen Fan, and Ning Xu. Video instance segmentation. In *ICCV*, 2019.

[38] Zhao Yang, Qiang Wang, Luca Bertinetto, Weiming Hu, Song Bai, and Philip H. S. Torr. Anchor diffusion for unsupervised video object segmentation. In *ICCV*, 2019.

[39] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. In *International Conference on Learning Representations*, 2015.

[40] Xiaohui Zeng, Renjie Liao, Li Gu, Yuwen Xiong, Sanja Fidler, and Raquel Urtasun. Dmm-net: Differentiable mask-matching network for video object segmentation. In *ICCV*, 2019.

[41] Mingmin Zhen, Shiwei Li, Lei Zhou, Jiaxiang Shang, Haoan Feng, Tian Fang, and Long Quan. Learning discriminative feature with crf for unsupervised video object segmentation. In *ECCV*, 2020.

[42] Tianfei Zhou, Jianwu Li, Shunzhou Wang, Ran Tao, and Jianbing Shen. Matnet: Motion-attentive transition network for zero-shot video object segmentation. *TIP*, 2020.

[43] Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets v2: More deformable, better results. In *CVPR*, 2019.

[44] I. E. Zulfikar, J. Luiten, and B. Leibe. UnOVOST: Unsupervised Offline Video Object Segmentation and Tracking for the 2019 Unsupervised DAVIS Challenge. *CVPR Workshops*, 2019.