Spatially-Adaptive Filter Units for Compact and Efficient Deep Neural Networks

Domen Tabernik[1] · **Matej Kristan**[1] · **Aleš Leonardis**[2,1]

Received: 15 February 2019 / Accepted: 11 December 2019

Abstract Convolutional neural networks excel in a number of computer vision tasks. One of their most crucial architectural elements is the effective receptive field size, which has to be manually set to accommodate a specific task. Standard solutions involve large kernels, down/up-sampling and dilated convolutions. These require testing a variety of dilation and down/up-sampling factors and result in non-compact networks and large number of parameters. We address this issue by proposing a new convolution filter composed of *displaced aggregation units* (DAU). DAUs learn spatial displacements and adapt the receptive field sizes of individual convolution filters to a given problem, thus reducing the need for hand-crafted modifications. DAUs provide a seamless substitution of convolutional filters in existing state-of-the-art architectures, which we demonstrate on AlexNet, ResNet50, ResNet101, DeepLab and SRN-DeblurNet. The benefits of this design are demonstrated on a variety of computer vision tasks and datasets, such as image classification (ILSVRC 2012), semantic segmentation (PASCAL VOC 2011, Cityscape) and blind image de-blurring (GOPRO). Results show that DAUs efficiently allocate parameters resulting in up to four times more compact networks in terms of the number of parameters at similar or better performance.

Keywords Compact ConvNets · Efficient ConvNets · Displacement units · Adjustable receptive fields

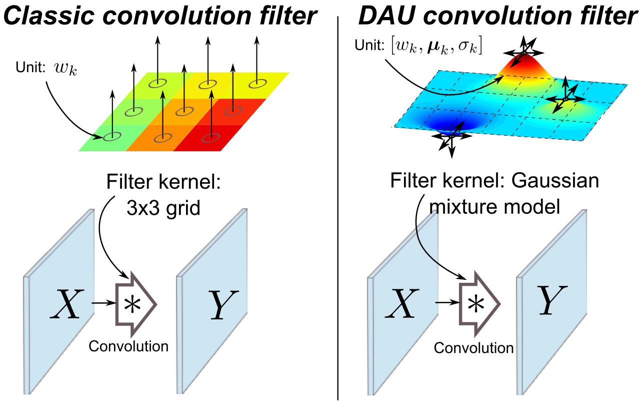


Fig. 1 A classic convolution filter with a fixed-grid filter kernel (left) is replaced with a convolution filter composed of several displaced aggregation units (DAUs) whose sub-pixel positions are learned (right).

1 Introduction

Deep convolutional neural networks (ConvNets) (He et al. 2016a; Xie et al. 2017; Redmon and Farhadi 2017; Kaiming et al. 2017) have demonstrated excellent performance across a broad spectrum of computer vision tasks, such as image classification (He et al. 2016a), semantic segmentation (Ronneberger et al. 2015; Chen et al. 2016a), image restoration, and blind image de-blurring (Tao et al. 2018). Early works showed that deep features pre-trained for one task (e.g., classification (Simonyan and Zisserman 2015)) can be applied with some success to other tasks (e.g. semantic segmentation (Shelhamer et al. 2016)). Direct application to another task, however, is sub-optimal and architectural changes are required (Chen et al. 2016b).

One of the crucial task-dependent architectural elements is the effective receptive field size of neurons (Luo et al. 2016). In image classification, relatively small convolution filters are used and the receptive field size is increased by

D. Tabernik[1], M. Kristan[1]
[1]Faculty of Computer and Information Science
University of Ljubljana, Slovenia
E-mail: domen.tabernik@fri.uni-lj.si
E-mail: matej.kristan@fri.uni-lj.si

A. Leonardis[2,1]
[2]School of Computer Science,
University of Birmingham, UK
E-mail: a.leonardis@cs.bham.ac.uk

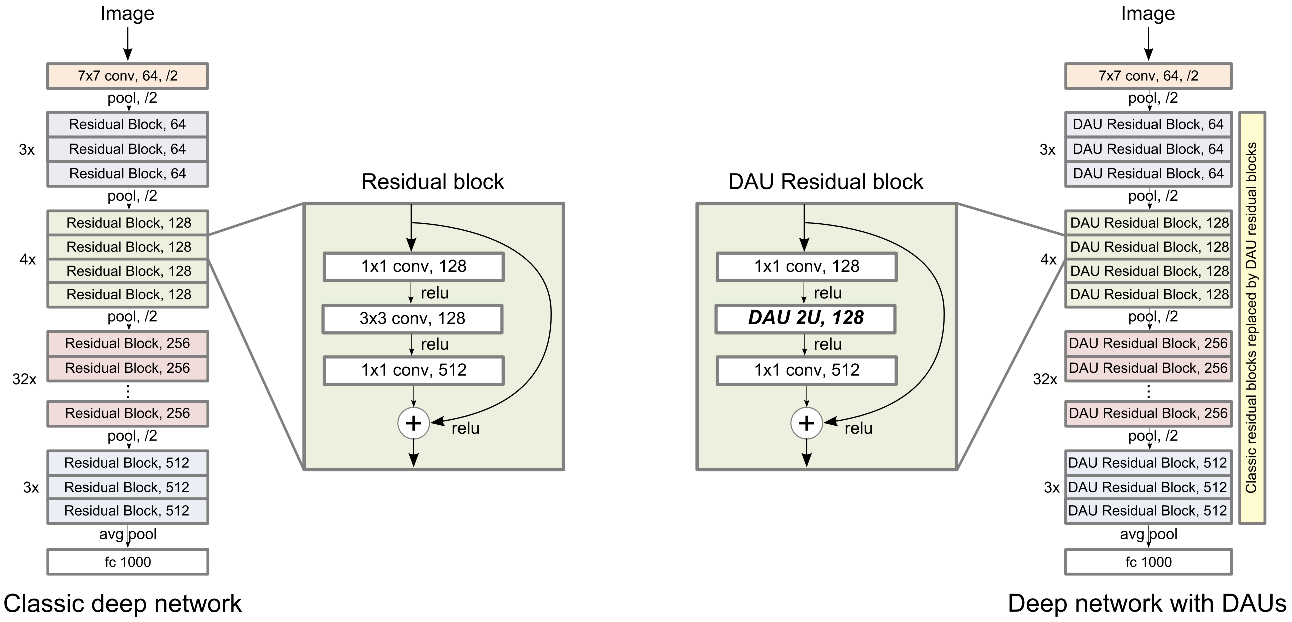


Fig. 2 Example of the state-of-the-art architecture (ResNet101) where classic convolution is replaced with the displaced aggregation units. In DAU residual block, DAUs with 2 units replace all 3 × 3 convolutions from the classic residual block.

gradual sub-sampling of features in consecutive layers. However, the output resolution obtained by such process is too coarse for tasks that require per-pixel predictions (e.g., semantic segmentation).

Standard strategies to address this issue are based on (i) removing the pooling layers or down-sampling, and using large convolution filters or (ii) keeping the filters small and appending a network with (gradual) up-sampling layers and skip connections (Ronneberger et al. 2015). Both strategies significantly increase the number of parameters leading to non-compact networks and inefficient learning. Jeon and Kim (2017) have proposed deforming small convolution kernels, but these deformations result in negligible change of the receptive field sizes.

The problem of parameter dependence on the filter size was partially addressed by Chen et al. (2016b) with dilated convolutions. The dilation factor is manually set and fixed, which may be sub-optimal for a given application. Chen et al. (2016b) thus proposed atrous spatial pyramid pooling (ASPP), which is composed of parallel processing paths, each path with a different dilation factor. Since ASPP entails a non-negligible increase of parameters, the authors propose adding it only as a post-processing block on top of deep features. There are several drawbacks of using fixed dilations. First, the pattern of the dilation is a regular expansion of a grid-based filter. The pattern is fixed prior to learning and cannot change. Other patterns might be more appropriate for a given task, but the search would lead to a combinatorial explosion of possible filters to be tested during learning, which is practically infeasible. Secondly, large dilations significantly violate the Nyquist theorem (Amidror 2013), resulting in gridding artifacts as demonstrated by Yu et al. (2017).

In this paper, the above issues with the effective receptive field size and filter pattern are addressed by introducing a novel formulation of the convolution filter. In particular, the standard CNN convolution filter, which is a regular grid of values, is replaced by a continuous parametric function. Specifically, a mixture of weighted Gaussian kernels is proposed for the functional form (see Figure 1). Each Gaussian acts as a unit that aggregates feature responses locally at its displacement. During learning the displacement of units is optimized along with other parameters—hence we call these *displaced aggregation units* (DAU). The number of parameters in the convolution filter is thus fixed by the number of units and does not increase with the receptive field size.

Our main contribution is a DAU convolution filter (Figure 1), which incorporates three novel concepts into the deep networks: (a) *decoupling of the parameter count* from the receptive field size, (b) *learning of the receptive field of each convolution filter* in the network, (c) and *automatic adjustment of the spatial focus* on the sub-feature from a previous layer through explicit modeling of the unit's position. By following those concepts the DAU-ConvNets exert compactness in terms of the number of parameters, efficient use of parameters, and allow adaptation to specific tasks without manually testing various complex dilation factors and filter patterns. Those properties directly contribute to the improved performance for various computer vision tasks.

The benefits of DAUs are demonstrated on a range of computer vision tasks (i.e., image classification, semantic image segmentation and blind image de-blurring) by replacing the standard convolution filters (see Figure 2). We empirically verify that the resulting novel deep models:

- enable automatic and *efficient allocation of parameters* for spatial attention while requiring as few as 25% of parameters compared to the standard ConvNets,
- address *a spectrum of different tasks* without ad-hoc manual tuning of receptive fields,
- *eliminate the need for dilated convolutions* with hand-crafted dilation factors, and
- *enable a novel analysis* of parameter allocation and spatial coverage in ConvNets.

This paper extends our (preliminary) work published in two conference papers (Tabernik et al. 2016, 2018), which considered only shallow architectures on small datasets. The DAU formulation is extended with details to cover the general form as well as its efficient formulation. Additional computer vision tasks are considered (i.e., blind image de-blurring), significantly deeper architectures are used (ResNet50, ResNet101 and DeepLab, SRN-DeblurNet) and several datasets (Citycape and GOPRO) are used to support the empirical findings. Implementations of DAU convolution filters in standard deep learning toolboxes[1] are available to the research community along with DAU modifications of popular deep architectures used in this paper.

The remainder of the paper is structured as follows. Section 2 provides a review of most closely related works. The DAU convolution filter is introduced in Section 3. A comprehensive empirical analysis of DAU convolution filter parameters and displacements is given in Section 4. We demonstrate DAUs on standard computer vision tasks: classification (Section 5), semantic segmentation (Section 6) and blind image de-blurring (Section 7). We conclude with a discussion in Section 8.

2 Related Work

The receptive field has been considered as an important factor for deep networks in several related works (Luo et al. 2016; Chen et al. 2017). Luo et al. (2016) measured an effective receptive field in convolutional neural networks and observed that it increases as the network learns. They suggest an architectural change that foregos a rectangular window of weights for sparsely connected units. However, they do not show how this can be implemented. Our proposed approach is in direct alignment with their suggested changes as our displaced aggregation units are a direct realization of their suggested sparsely connected units.

The importance of deforming filter units has also been indicated by recent work of Dai et al. (2017). They implemented spatial deformation of features with deformable convolutional networks, while a general non-euclidean based formulation for a non-grid based input data was later proposed by Chang et al. (2018). Both explicitly learn feature displacements but learn them on a per-pixel location basis for input activation maps and share them between all channels. Dai et al. (2017) also applies deformations only to the last few convolution layers on regions generated by Mask R-CNN (Kaiming et al. 2017) with the effective outcome of normalizing scale-changes relative to the pixel position in the region of interest. Our model instead learns different displacements for different channels and shares them over all pixel locations in the input activation map. DAUs are applied to all layers with the goal of decoupling the receptive field size from the kernel size thus reducing the number of parameters and simplifying the architecture. Formulation of both methods also makes them conceptually complementary to each other.

Deforming filter units has also been explored by Jeon and Kim (2017), which, as opposed to deformable convolutions, apply deformation on filter units similarly as to our approach. They use bilinear interpolation similar to ours to get displacements at a sub-pixel accuracy, however, their limitation is in relying on 3×3 filters. They can neither displace them to more than a single neighboring pixel nor adapt them during the learning stage to an arbitrary position as we do. They also increase their parameter count as they still use 9 units per filter. We show that significantly fewer units are needed.

Works by Luan et al. (2017) and Jacobsen et al. (2016) changed the filter definition using different parametrization techniques. Both decompose filter units into a linear combination of edge filters. They show a reduction in parameters per filter but their models do not allow displacements of filter units to arbitrary values. Their models have fixed receptive fields defined as hyperparameters and cannot be learned as ours. This also prevents any further analysis on distribution of displacements and receptive field sizes which is possible with our model with the explicit modeling of the unit's displacements.

Several papers also studied the importance of number of parameters. Eigen et al. (2014) performed an extensive evaluation and studied where parameters should be allocated, either for additional layers or for additional features. They concluded that it is more useful to allocate them for additional layers, but their study was limited by convolutional filter design and did not study how many should be allocated in spatial dimensions. Others have also observed inefficient allocation of parameters. For instance, Jaderberg et al. (2014) performed a low-rank analysis of pre-trained model weights and showed significant compression rate, while Iandola et al.

[1] A low-level CUDA implementation of the DAU convolution filters are available in Caffe as well as Tensorflow at: https://github.com/skokec/DAU-ConvNet-caffe and https://github.com/skokec/DAU-ConvNet

(2016) proposed a more efficient network with 50-times less parameters. Most of the approaches reduce number of parameters through compression or architectural design change. Such compression techniques are complementary to ours and can be applied to our model as well.

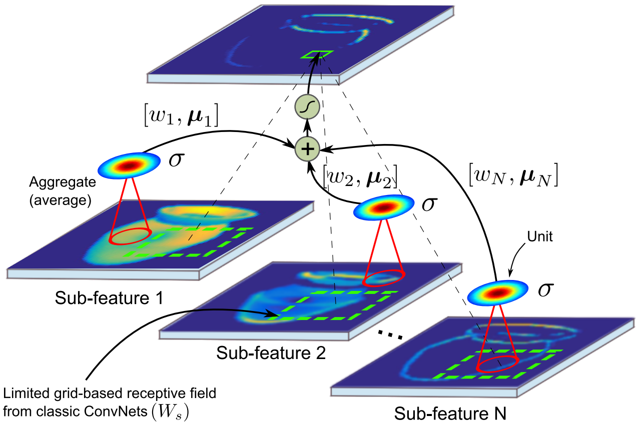

$[w_1, \boldsymbol{\mu}_1]$
σ
Aggregate
(average)
$[w_2, \boldsymbol{\mu}_2]$
σ
$[w_N, \boldsymbol{\mu}_N]$
Unit
σ
Sub-feature 1
Sub-feature 2
Sub-feature N
Limited grid-based receptive field
from classic ConvNets (W_s)

Fig. 3 Convolution filter with displaced aggregation units (DAUs) is composed of several displaced units that aggregate within a constrained area of the underlying sub-features from the lower layer.

3 Displaced Aggregation Units (DAU)

We start by defining displaced aggregation units (DAUs) in their most general form. The derivatives required for learning in standard deep learning frameworks are presented in Section 3.1 and an efficient formulation for fast inference is derived in Section 3.2.

The activation map of the i-th feature (input into the current layer of neurons) is defined in the standard ConvNet as

$$Y_i = f(\textstyle\sum_s W_s * X_s + b_s), \tag{1}$$

where for each s-th input channel, b_s is a bias, $*$ is a convolution operation between the input map X_s and the filter W_s, and $f(\cdot)$ is a non-linear function, such as ReLU or sigmoid (LeCun et al. 1998). In convolution networks applied to images, the X_s and Y_i are two-dimensional features, while W_s is two-dimensional convolution filter. We refer to the individual weight value of a filter W_s in the standard ConvNet as its unit.

We re-define the filters W_s as a mixture of localized aggregated feature responses from the input feature map (see Figure 3). A Gaussian function is chosen for the analytic form of the aggregation units, although any other differentiable function that models the aggregation and displacement can be used. The resulting displaced aggregation unit (DAU) convolution filter is thus written as a mixture of K units

$$W_s = \sum_{k=0}^{K} w_k G(\boldsymbol{\mu}_k, \sigma_k), \tag{2}$$

where the unit's displacement and aggregation perimeter are specified by the mean $\boldsymbol{\mu}_k$ and standard deviation σ_k, respectively, and w_k is the input amplification factor (i.e., the unit weight). Note that parameters w_k, $\boldsymbol{\mu}_k$ and σ_k depend on the specific s-th input channel, as well as, on the specific i-th output channel, but in the interest of clarity we omit these in the notation. We refer to σ_k as the aggregation perimeter since values of the Gaussian function at 3σ become small and its contribution will be negligible. Therefore, 3σ represents an approximate cutoff point of the unit's aggregation.

The displaced aggregation unit, denoted by $G(\cdot)$, is implemented with a normalized Gaussian. To avoid discretization errors in $G(\boldsymbol{\mu}_k, \sigma_k)$ when implementing continuous function in a discrete convolution filter kernel, we replace the normalization factor computed in the continuous space with one computed in the discretized space, leading to our final aggregation unit $G(\mathbf{x}; \boldsymbol{\mu}_k, \sigma_k)$

$$G(\mathbf{x}; \boldsymbol{\mu_k}, \sigma_k) = \frac{1}{N(\boldsymbol{\mu}_k, \sigma_k)} \cdot \exp(-\frac{\|\mathbf{x} - \boldsymbol{\mu_k}\|^2}{2\sigma_k^2}), \tag{3}$$

where $N(\boldsymbol{\mu}_k, \sigma_k)$ is the normalization term, i.e.,

$$N(\boldsymbol{\mu}_k, \sigma_k) = \textstyle\sum_{\mathbf{x}} \exp(-\frac{\|\mathbf{x}-\boldsymbol{\mu_k}\|^2}{2\sigma_k^2}). \tag{4}$$

The proposed DAU-based convolution filter is similar to a standard Gaussian mixture model, but we do not enforce $\sum w_k = 1$ since the DAU weight $w_k \in [-\infty, \infty]$ can take any value.

3.1 Learning DAU Convolution Filter

Learning the parameters of individual DAU consists of learning the displacement $\boldsymbol{\mu}_k$, the aggregation perimeter σ_k and the weight w_k. The number of DAUs, K, in the convolution filter is a hyper-parameter that has to be set prior to learning. Since DAUs are analytic functions, the filter parameters are fully differentiable and compatible with the standard ConvNet gradient-descent learning techniques based on backpropagation.

Parameters are thus optimized by computing the gradients w.r.t. the cost function $l(y, \bar{y})$, which leads to three different types of gradients. By applying the chain rule, we define the gradient for the weight $\frac{\partial l}{\partial w_k}$ as a dot-product of the back-propagated error and the input feature X_s convolved with the k-th DAU, i.e.,

$$\frac{\partial l}{\partial w_k} = \sum_{n,m} \frac{\partial l}{\partial z} \cdot \frac{\partial z}{\partial w_k} = \sum_{n,m} \frac{\partial l}{\partial z} \cdot \sum_{\mathbf{x}} X_s * G(\mathbf{x}; \boldsymbol{\mu}, \sigma), \tag{5}$$

where n, m run over width and height of the image, x runs over discretized kernel positions, while $z = \sum_s W_s * X_s + b_s$

and $\partial l / \partial z$ is the back-propagated error. Note that only the s-th channel of input features are used since the weight w_k appears only in W_s. The back-propagated error for layer n follows the standard approach:

$$\frac{\partial l}{\partial z_s^n} = \frac{\partial l}{\partial z_s^{n+1}} * rot(W_s), \tag{6}$$

where the back-propagated error from the higher layer $n+1$ is convolved with the 180°rotated filter $rot(W_s)$ which can be computed from Eq. (2). We can similarly apply the chain rule to obtain the gradient for the mean and the standard deviation,

$$\frac{\partial l}{\partial \mu_k} = \sum_{n,m} \frac{\partial l}{\partial z} \cdot \sum_{\mathbf{x}} X_s * \frac{\partial G(\mathbf{x}; \boldsymbol{\mu_k}, \sigma_k)}{\partial \mu_k}, \tag{7}$$

$$\frac{\partial l}{\partial \sigma_k} = \sum_{n,m} \frac{\partial l}{\partial z} \cdot \sum_{\mathbf{x}} X_s * \frac{\partial G(\mathbf{x}; \boldsymbol{\mu}_k, \sigma_k)}{\partial \sigma_k}, \tag{8}$$

where the derivatives of the Gaussian are

$$\frac{\partial G(\mathbf{x}; \boldsymbol{\mu}, \sigma)}{\partial \mu} = w \frac{N(\boldsymbol{\mu}, \sigma) \cdot \frac{g(\mathbf{x}; \boldsymbol{\mu}, \sigma)}{\partial \mu} - g(\mathbf{x}, \theta) \cdot \frac{\partial N(\boldsymbol{\mu}, \sigma)}{\partial \mu}}{[N(\boldsymbol{\mu}, \sigma)]^2}, \tag{9}$$

$$\frac{\partial G(\mathbf{x}, \boldsymbol{\mu}, \sigma)}{\partial \sigma} = \tilde{w} \frac{N(\boldsymbol{\mu}, \sigma) \cdot \frac{g(\mathbf{x})}{\partial \sigma} - g(\mathbf{x}; \boldsymbol{\mu}, \sigma) \cdot \frac{\partial N(\boldsymbol{\mu}, \sigma)}{\partial \sigma}}{[N(\boldsymbol{\mu}, \sigma)]^2}. \tag{10}$$

3.2 Efficient Inference and Learning of DAUs

DAUs can be efficiently implemented in ConvNets by exploiting the translational invariance property of the convolution. The displacement of a Gaussian relative to the filter manifests in a shifted convolution result, i.e.,

$$\begin{aligned} f * G(\boldsymbol{\mu}_k, \sigma) &= f * \mathcal{T}_{\boldsymbol{\mu}_k}[G(\sigma)] \qquad (11) \\ &= \mathcal{T}_{\boldsymbol{\mu}_k}[f * G(\sigma)], \qquad (12) \end{aligned}$$

where $\mathcal{T}_x(g, y) = g(y - x)$ is translation of function $g(\cdot)$ and $G(\sigma)$ is a zero-mean Gaussian. Thus, the activation map computation can be written as:

$$Y_i = f\left(\sum_s \sum_k w_k \mathcal{T}_{\boldsymbol{\mu}_k}(G(\sigma) * X_s) + b_s \right). \tag{13}$$

This formulation affords an efficient implementation by pre-computing convolutions of all inputs by a single Gaussian kernel, i.e., $\tilde{X}_s = G(\sigma) * X_s$, and applying displacements by $\boldsymbol{\mu}_k$ to compute the aggregated responses of each output neuron. The size of the blurring kernel is determined by the standard deviation $(2 \cdot \lceil 3\sigma \rceil + 1)$, however, large kernels do not add much computational cost since blurring represents only $1\% - 3\%$ of the whole computational cost. The efficient implementation requires sharing of the same aggregation perimeter σ value among all units of the same layer. In a preliminary study, we have determined that this constraint is compensated for by the other free parameters in DAUs and performance is not affected. In fact, further constraints can be applied to the aggregation perimeters, which are empirically analyzed in Section 4.1.

Due to discretization, the Eq. (13) is accurate only for discrete displacements $\boldsymbol{\mu}_k$. We address this by re-defining the translation function in Eq. (13) as a bilinear interpolation,

$$\hat{\mathcal{T}}_x(g, y) = \sum_i \sum_j a_{i,j} \cdot g(y - \lfloor x \rfloor + [i, j]), \tag{14}$$

where $a_{i,j}$ are bilinear interpolation weights. This allows computing sub-pixel displacements and can be efficiently implemented in CUDA kernels.

The aggregation perimeter constraints and the displacement re-formulation also make the learning more efficient. Only two parameters have to be trained per DAU, i.e., the weight w_k and the spatial displacement μ_k, while the aggregation perimeter and the number of DAUs per convolution filter are hyperparameters[2].

Applying the efficient formulation to the learning of DAU convolution filter results in the following partial derivatives:

$$\frac{\partial l}{\partial w_k} = \sum_{n,m} \frac{\partial l}{\partial z} \cdot \sum_{\boldsymbol{x}} \hat{\mathcal{T}}_{\boldsymbol{\mu}_k}(X_s * G(\sigma)), \tag{15}$$

$$\frac{\partial l}{\partial \mu_k} = \sum_{n,m} \frac{\partial l}{\partial z} \cdot \sum_{\boldsymbol{x}} w_k \cdot \hat{\mathcal{T}}_{\boldsymbol{\mu}_k}(X_s * \frac{\partial G(\sigma)}{\partial \mu}), \tag{16}$$

where $\frac{\partial l}{\partial z}$ is back-propagated error.

Similarly to the inference, the gradient can be efficiently computed using convolution with zero-mean Gaussian (or its derivatives) and sampling the response at displacement specified by the mean values in the DAUs.

The backpropagated error for the lower layer is computed similarly to the classic ConvNets, which convolve the backpropagated error on the layer output with rotated filters. Since the DAUs are rotation symmetric themselves, only the displacements have to be rotated about the origin and Eq. (13) can be applied for computing the back-propagated error as well, yielding efficient and fast computation.

Computational cost Compared to the implementation of DAUs based on standard convolution (Tabernik et al. 2016) that discretize DAUs to large kernels, the efficient DAU implementation results in several times faster inference and an

[2] Note that reasonable aggregation perimeter value σ can in fact be estimated for a given problem by pre-training using the derivatives in Eq. (8), but using fixed value has proven sufficient. See Section 4.1 for the analysis of different choices of this parameter.

order of magnitude faster learning. However, the speed-up factor is dependent on the number of DAUs per channel K, and on the maximum displacement value. Considering the following input sizes:

$$X_s = [W \times H \times S]\,,\;\; Y_i = [W \times H \times F]\,,$$
$$G_k = \left[\hat{\mathcal{K}}_w \times \hat{\mathcal{K}}_h\right],$$

where S is the number of input channels, F is the number of output channels and K is the number of DAUs per input channel, then the computational cost for the efficient DAU implementation is $\mathcal{O}(4 \cdot S \cdot F \cdot K \cdot W \cdot H + S \cdot W \cdot H \cdot \hat{\mathcal{K}}_w \cdot \hat{\mathcal{K}}_h)$, where 4 relates to the bi-linear interpolation. Blurring with the Gaussian kernel $\hat{\mathcal{K}}_w \times \hat{\mathcal{K}}_h$ in the second term is performed F-times fewer than the first term, thus making the blurring part negligible for large number of input channels F. With the standard convolutional network using the $\hat{\mathcal{K}'}_w \cdot \hat{\mathcal{K}'}_h$ convolution kernel, the computational complexity is $\mathcal{O}(F \cdot S \cdot W \cdot H \cdot \hat{\mathcal{K}'}_w \cdot \hat{\mathcal{K}'}_h)$, and the speed-up factor γ becomes:

$$\gamma = \frac{\hat{\mathcal{K}'}_w \cdot \hat{\mathcal{K}'}_h}{4 \cdot K}. \tag{17}$$

With large DAU displacements resulting in bigger kernels for the standard convolution implementation, the speed-up of the efficient DAU implementation becomes more significant since kernel size $\hat{\mathcal{K}'}_w \cdot \hat{\mathcal{K}'}_h$ for the standard ConvNet must increase quadratically for larger displacements, while efficient DAU retains the same kernel size $\hat{\mathcal{K}}_w \cdot \hat{\mathcal{K}}_h$ and the same number of DAUs regardless of the displacements values.

4 Analysis of the Displaced Aggregation Units

An extensive empirical analysis was performed to gain insights into the properties of the displaced aggregation units CNN formulation (DAU-ConvNet). We have focused on two main parameters: (i) the DAU aggregation perimeter encoded by the variance of a Gaussian (Section 4.1) and (ii) the number of units per convolution filter (Section 4.2). We have then analyzed the learned convolution filters in terms of displacement distributions of the DAUs and thier receptive fields (Section 4.3), and analyzed the practical computational savings (Section 4.4).

4.1 Influence of the DAU Aggregation Perimeter

The aggregation perimeter of a single DAU is determined by the standard deviation, σ, of the corresponding Gaussian (Eq. 3). In our most general formulation, the standard deviation can be learned for each unit by backprop (Eq. 8), which in practice increases the computational complexity of the learning. The standard deviation plays several roles. On the one hand, it defines the region within which the DAU neuron aggregates features from a previous layer and on the other hand, it enables computation of smooth derivatives for DAU displacement optimization (Eq. 10) when σ is large enough to encompass neigbooring pixels. We explore the trade-off between learning all the standard deviations and fixing them to a reasonable value that affords sufficient aggregation as well as displacement optimization.

Table 1 Standard deviation σ hyperparameter evaluation on CIFAR10 classification task using a shallow DAU-ConvNet. Standard deviation has minor effect on classification performance.

Std. deviation $\sigma =$	0.3	0.4	0.5	0.6	0.7	0.8	Learned
DAU-ConvNet CIFAR10	82.9	83.4	**83.8**	83.6	82.9	82.8	**84.25**

The experiments were carried out on CIFAR10 (Krizhevsky 2009) classification problem using a network with three convolutional layers with DAU filters and three max-pooling layers followed by a fully-connected layer for predicting the image class. Batch normalization (He et al. 2014) was applied and weights were initialized using (Glorot and Bengio 2010). We trained the network with a stochastic gradient descent and a softmax loss function for 100 epochs using a batch size of 256 images. Learning rate was set to 0.01 for the first 75 epochs and reduced to 0.001 for remaining epochs. A momentum of 0.9 was also used.

In the first experiment, the number of DAUs per convolution filter was fixed to four, while the standard deviations were learned. The standard deviations were initialized randomly with a uniform distribution over $[0.3, 0.8]$. As reported in Table 1, DAU-ConvNet with learned standard deviation achieved a 84.25% accuracy. Top row in Figure 4 shows the distribution of standard deviations for each layer after learning. Distributions for the second and the third layer remained uniformly distributed even after the training, while in the first layer, the standard deviations converged towards smaller values. The distributions in Figure 4 were weighted by the learned unit's weight to reflect the changes of standard deviations only in the neurons that significantly contribute to the output. Comparison of the initial and final standard deviations in the bottom row of Figure 4 confirms that learning affected only the first layer, while the following two layers were negligibly affected. This indicates that DAU structure does not benefit considerably from learning the standard deviations and supports the use of simplification that leads to efficient inference and learning.

As noted in Section 3.2, the learning and inference of DAU-ConvNets can be made efficient by fixing the standard deviations in DAUs in the same layer. Table 1 also reports the results obtained by fixing the standard deviations to $\{0.3, 0.4, 0.5, 0.6, 0.7, 0.8\}$ in all layers. The results

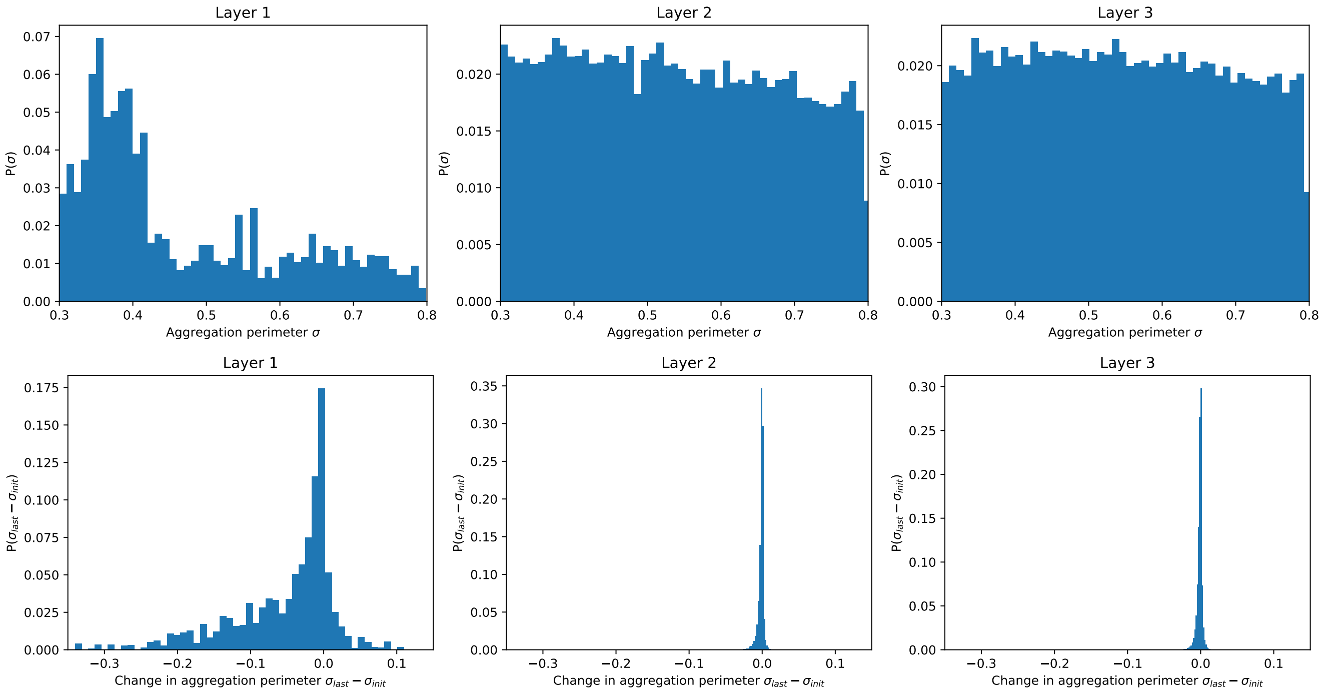


Fig. 4 Distribution of learned DAU perimeters (σ) in the top row, and changes in the learned DAU perimeters from the initialization in the bottom row, both for model trained on the CIFAR10 dataset for each three layers in the network.

show that the classification rates vary by approximately 1%, which means that the specific value of the perimeter negligibly affects the classification performance as long as it is set to a reasonable value. This means that unit's displacements and weights compensate for the chosen standard deviations. Comparing the performance of the DAU-ConvNet with the standard deviations fixed to 0.5 and the DAU-ConvNet with the individually learned standard deviations (*"Learned"* in Table 1), we also observe a negligible difference of 0.5%. Note that individually learned standard deviations prevent the use of efficient DAU implementation and require the implementation based on standard convolutions (Tabernik et al. 2016). Such implementation is significantly slower (cf. Section 3.2) and prevents the use of DAUs in very deep modern architectures.

Fixing the perimeters reduces the computational complexity of learning and very deep DAU-ConvNets can be trained. Thus in the remaining part of the experiments, we have fixed the DAU standard deviations to 0.5.

4.2 Influence of the Number of DAUs

With the standard deviations fixed, each DAU contains three parameters: two parameters for 2D displacement and a weight, which are learned from the data. A discrete parameter that has to be manually set, however, is the number of DAUs in the DAU convolution filter. This parameter thus determines the total number of parameters to be learned in the DAU-ConvNet and we analyze its impact on performance here using the ILSVRC 2012 (Russakovsky et al. 2015) image classification task with a moderately deep standard ConvNet architecture.

In a classical ConvNet, the units are equivalent to pixels in the convolution filters. Several research papers investigated the influence of the number of parameters in classic ConvNets with respect to the number of layers, number of features and filter sizes (Eigen et al. 2014), but could not analyze the impact of the number of units independently from the convolution filter size. The classic ConvNets are limited to a minimum of 9 parameters per filter, which corresponds to a 3 × 3 filter. Receptive fields may be increased with the dilated convolution (Holschneider et al. 1990) without increasing the number of parameters, but any such change requires hard-coding the size and the pattern, which leads to a combinatorial explosion of possible convolution filters.

The DAU formulation of convolution filter, on the other hand, allows us to investigate filters with even smaller number of parameters without affecting the spatial coverage and the receptive field sizes, since these are learned from the data. In addition, the DAU convolution filter definition (Eq. 2) provides a straight-forward way to prune the units. During the training, the number of units is kept the same in all filters. After the training is finished, the units with very small weights are removed, which further reduces the number of parameters.

Table 2 Analysis of the number of parameters and units per filter with three variants of DAU-AlexNet: Large, Medium and Small. Rows also show the elimination of units based on their amplification value. In columns we report classification top-1 accuracy on ILSVRC2012 validation set, the number of DAU on all filters and the percentage of removed units.

Relative threshold	*Large DAU-AlexNet*			*Medium DAU-AlexNet*			*Small DAU-AlexNet*		
	Acc. (%)	# units	% removed	Acc. (%)	# units	% removed	Acc. (%)	# units	% removed
0	**57.3**	1,523,712	0	56.9	786,432	0	56.4	393,216	0
0.01	**57.3**	1,389,131	8	56.8	739,884	6	56.4	378,692	4
0.02	57.1	1,325,057	13	56.7	707,745	10	56.4	366,144	7
0.05	40.1	1,157,129	24	54.8	623,923	20	55.4	331,137	16
0.10	28.3	925,509	39	47.4	507,651	35	49.6	279,162	29
0.25	0.2	453,987	70	1.9	261,093	66	0.9	154,624	61

Table 3 Per-filter unit and parameter count with three variants of DAU-ConvNet: Large, Medium and Small. Note, a unit in DAU has three parameters while a unit in a classic ConvNet has a single parameter.

	Per-filter unit count			
	Large	Medium	Small	AlexNet
Layer 2	6	4	2	5 × 5
Layers 3-5	4	2	1	3 × 3
	Per-filter parameter count			
Layer 2	18	12	6	25
Layers 3-5	12	6	3	9

Architecture. AlexNet (Krizhevsky et al. 2012), composed of 7 layers (5 convolutional and 2 fully connected) was chosen for the baseline architecture for this experiment. We used a single-pipeline AlexNet, that does not require splitting into two streams as originally proposed (Krizhevsky et al. 2012). For simplicity, we refer to a single-pipeline AlexNet as a standard AlexNet. The local normalization layers, max-pooling and dropout on fully-connected layers were kept. The initialization was changed to a technique by Glorot and Bengio (Glorot and Bengio 2010). The baseline AlexNet was modified into a DAU-AlexNet by replacing discrete convolution filters in layers 2 to 5 with DAU convolution filters presented in Section 3.

Three variations of DAU-AlexNet are constructed: Large, Medium and Small. Different number of units and parameters per kernel for each variant are shown in Table 3 and follow approximate coverage of filter sizes from standard AlexNet with 5 × 5 filter sizes for the second layer and 3 × 3 filter sizes for the remaining layers. The Small DAU-AlexNet contains 400,000 DAUs, the Medium DAU-AlexNet contains 800,000 DAUs, and the Large DAU-AlexNet contains 1.5 million DAUs, which translates to 1.2 million, 2.3 million, and 4.5 million parameters, respectively. For reference, the baseline AlexNet contains 3.7 million parameters.

Dataset. The networks were trained on 1.2 million training images from ILSVRC 2012 (Russakovsky et al. 2015) and tested on 50,000 validation images. All images were cropped and resized to 227 pixels as in the reference AlexNet architecture (Krizhevsky et al. 2012). To keep the experiments as clean as possible, we did not apply any advanced augmentation techniques apart from mirroring during the training with a probability of 0.5.

Optimization. The networks were trained by stochastic gradient descent with the batch size of 128 for 800,000 iterations, or 80 epochs. The initial learning rate was set to 0.01 and was reduced by a factor of 10 every 200,000th iteration. A momentum with a factor of 0.9 was applied together with a weight decay factor of 0.0005. The weight decay was applied only on the weights w_k in DAUs but not to the displacement values $\boldsymbol{\mu_k}$.

Results and discussion. The results are reported in Table 2. We observe that all three DAU-AlexNets achieve classification accuracy of approximately 56-57%, which is comparable to the standard AlexNet (cf. Section 5). This performance is already achieved by a DAU-AlexNet with two or less units per convolution filter, resulting in 3 to 6 parameters per filter, which is significantly lower than in a standard AlexNet with 9 parameters for the smallest filter (i.e., 3 × 3) and 25 for a moderately large filter (i.e., 5 × 5). Note that the efficient use of parameters is possible since DAU-AlexNet learns the required convolution filter receptive field size through its adjustable units without increasing its parameters.

Further improvements are observed after the pruning – eliminating units with low weights. Table 2 shows that in all DAU-AlexNets, 7-13% of units can be removed without reducing the classification accuracy. The relation between the number of parameters and the classification performance is visualized in Figure 5. A steep increase in performance by the Small DAU-AlexNet shows that even pruning the smaller network can reduce some parameters while retaining fairly good results. Almost 50% accuracy can be maintained with less than 300,000 units, which is 10-fold less than in standard AlexNet. Furthermore, comparing to the less steep decline of the Large DAU-AlexNet shows that pruning is less effective than just learning with a limited set of units.

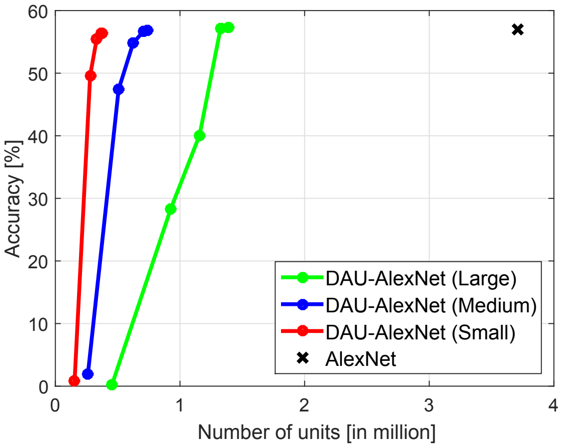


Fig. 5 Classification accuracy on ILSVRC 2012 dataset with respect to the number of DAUs in the network (in million). DAU-based AlexNet requires an order of magnitude fewer units than standard AlexNet at the same classification accuracy. Note that in DAU-AlexNet, the unit refers to one Gaussian component with a learnable weight and displacement, while in AlexNet, the unit refers to one weight in a kernel.

This indicates that larger network has encoded its information over many more units than are necessary.

4.3 Spatial Adaptation of DAUs

Next, we have investigated spatial distribution of the learned DAUs displacements from the convolution filter centers. The aim of the experiment was to expose whether certain displacement sizes are favored for a given task, and what are the indicated receptive field sizes of the convolution filters.

Such an experiment is practically unfeasible with classical ConvNets and requires a combinatorial sweep over alternative architectures with various hand-crafted filter designs. On the task of segmentation, for example, convolution filter receptive fields may be increased by dilated convolutions (Chen et al. 2016b), but the dilation factor has to be manually set. In contrast, the DAU convolution filters optimize their units with sub-pixel accuracy and can vary across the filters, thus no hand-crafting is required.

We investigate a 1D distribution of distances to the convolution filter centers as well as 2D distributions aggregated over all convolution filters. In both distributions a specific DAU contributes to the overall distribution proportionally to the unit absolute weight.

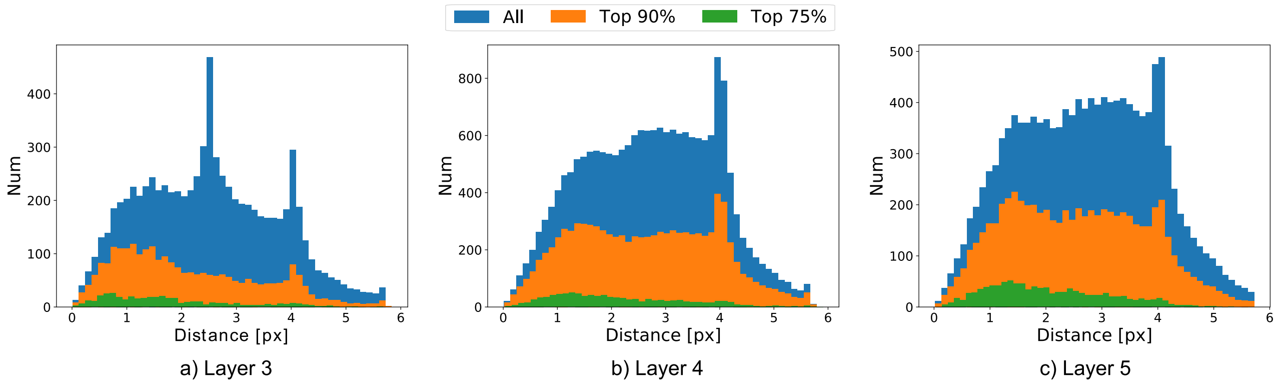
Architecture. A pre-trained Medium DAU-Alexnet architecture from the previous section was adapted to the segmentation task to perform a fine pixel-level class prediction as follows. The last fully-connected classification layer was replaced by the expansion and classification layer from Long et al. (2015) that entails a 1×1 classification layer and bilinear up-sampling with deconvolution layer to obtain pixel-wise mask. By removing the last two max-pooling layers we further increase the resolution which results in object boundaries maintained sharp. In this way the down-sampling factor is reduced from $32\times$ to $8\times$. Increasing the resolution of a pre-trained DAU-AlexNet model results in missaligned DAU positions, which were trained for a lower resolution. This is compensated for by proportionally increasing the displacements of DAUs in the layers with the increased resolution.

Dataset. The PASCAL VOC 2011 (Everingham et al. 2011) segmentation challenge was used. The training set was a combination of 1,112 training images from the PASCAL VOC 2011 segmentation challenge and 7,386 images collected by Hariharan et al. (2011). Performance was evaluated on the PASCAL VOC 2011 validation set excluding the images from Hariharan et al. (2011).

Optimization. The models were trained by mini-batch stochastic gradient descent for 65,000 iterations (150 epochs) with a batch size of 20 images. A fixed learning rate of 0.0002 was used, weight decay was set to 0.0005 and momentum to 0.9. Similar to Long et al. (2015), the added classification layer was initialized with zeros and a normalized per-pixel softmax loss was applied on pixels with valid labels.

Results and discussion. The DAU-AlexNet achieves segmentation accuracy comparable to the standard AlexNet with dilated convolutions (c.f. Section 6), which verifies that the network has properly adapted to the task. DAU displacement distributions were computed separately for layers 3, 4 and 5. In particular, three different distributions were computed. The first distribution considered locations of all DAUs, the second considered locations of the DAUs with weight at least 90% of the largest absolute weight and the third considered locations of the DAUs with the weight at least 75% of the largest absolute weight. The resulting 1D and 2D distributions are visualized in Figure 6 and Figure 7, respectively.

Two significant spikes are observed in the 1D distributions in Figure 6. One spike corresponds to 2.5 pixels displacement and the other to 4 pixels displacement. The spike at 2.5 pixels occurs only at the third layer and corresponds to DAU initialization points, which means that many units did not move significantly. This is confirmed by the high density regions in 2D distributions Figure 7, at initialization centers (red dots). However, a further inspection reveals that these units do not contribute in the inference, since their weights are very small. In fact, they disappear in the distributions of Figure 6 and Figure 7 with the DAUs corresponding to negligible weights removed. This is in effect the self-pruning property of DAU-ConvNets which was observed in Section 4.2. The spikes at initialization points are not apparent at the 4th and the 5th layers in the corresponding distributions (Figure 6 and Figure 7). This means that in



a) Layer 3

b) Layer 4

c) Layer 5

Fig. 6 Distance-to-center distributions collected from displacement of DAUs. Distributions are shown per-layer and after keeping only units corresponding to the top percentage of absolute weights (blue, orange, green).

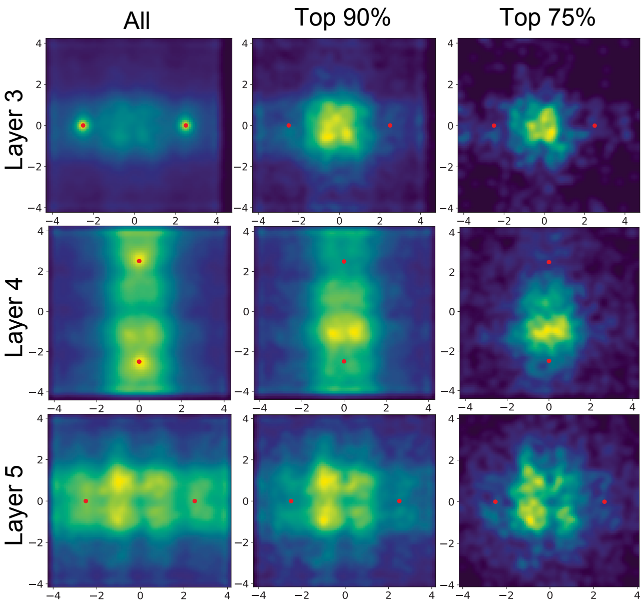


Fig. 7 2D distributions of displacements collected from DAUs. Red dots indicate initialization points. Distributions reported for layer 3, 4 and 5 in top, middle and bottom row, respectively. The three columns show the distributions after keeping only units corresponding to the top percentage of absolute weights.

this particular problem the DAUs are redundant only at the 3rd layer.

The second spike at 4 pixels is significant and does not disappear when removing DAUs with small weights (Figure 6). The spike occurs due to a limitation of our implementation that constraints the receptive field size, which in our case is set at four pixels in both spatial dimensions[3]. Still, a significant number of those units have large weights, which suggests that even larger receptive fields would be observed if unconstrained by the implementation specifics of GPU processing.

The overall shape of the displacement distribution is consistent across all layers (Figure 6). This indicates a preference to densely cover locations 1-2 pixels away from the center for the segmentation task. Some units with large weights are located far away from the center, which indicates a need to cover large receptive fields albeit with a lower density. The same conclusion is drawn from 2D spatial distributions in Figure 7.

The Effective Receptive Field. We further visualize the receptive field of the network by calculating the Effective Receptive Field (ERF) as introduced by Luo et al. (2017). The ERF measure calculates the effective receptive field of a single output pixel in a specific channel by back-propagating the error with only one active pixel in the corresponding channel. We report ERF averaged over all the channels for each layer and depict ERF as contour plots that well capture the extent of the receptive field. A contour represents the area with a fixed percentage of the influence to the output neuron. For instance, all pixels within 75% line represent 75% of the whole influence on the output neuron.

The effective receptive fields for DAU-AlexNet and standard AlexNet trained on the semantic segmentation are reported in Fig. 8. In all three layers, the DAU-ConvNet consistently demonstrates larger receptive field sizes than the ConvNet. The most noticeable difference is demonstrated in the 5th layer in the bottom row, where in the standard ConvNet the 99.9% of the influence to the output neuron is concentrated at only approximately 60 pixels from the center, while in DAU-CovNet, this influence is extended by nearly a factor of two.

4.4 Computational cost

As show in Section 3.2, the computational cost of the DAU model is dependent only on the number of DAUs per channel, and not on the size of the convolution kernel, as in the standard convolution. In practice, we found $K = 2$ and 4

[3] Our current implementation in CUDA allows only distances up to 4 or 8 pixels. This limitation can be overcome by modifying the implementation.

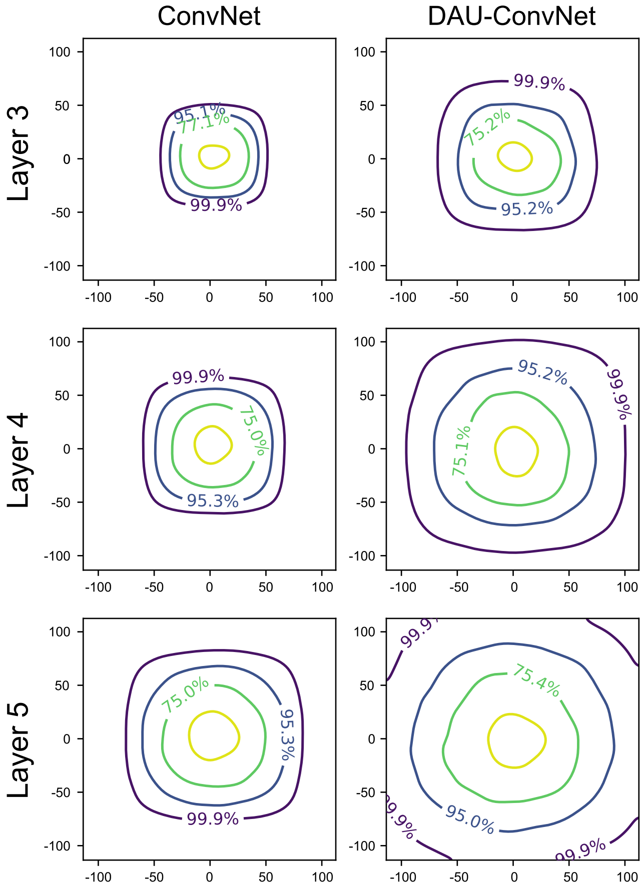


Fig. 8 The effective receptive fields (ERF) in a contour plot visualization for layers 3, 4 and 5 in standard AlexNet (left) and DAU-AlexNet (right) trained for the semantic segmentation. The size of the visualization patch is 227 × 227 pixels. Note the inner (yellow) contour presents a 25% influence area.

pixel displacements (corresponding to $\hat{\mathcal{K}'}_w \cdot \hat{\mathcal{K}'}_h = 9 \times 9$) are sufficient for large networks such as AlexNet or ResNet. This results in a theoretical speed-up of $\gamma = 10.125$. Profiling an efficient DAU implementation on a shallow 3-layer architecture with $K = 2$, $\hat{\mathcal{K}'}_w \cdot \hat{\mathcal{K}'}_h = 9 \times 9$, 192 output features and 32 × 32 input resulted in 3.25 times faster inference and 12.54 times faster learning on NVIDIA RTX 2080 Ti compared to the implementation based on the standard convolution. Note that in the efficient DAU the sigma was not learned, therefore adding additional 1.33-times speed-up compared to the theoretical speed-up (Eq. 17), while the difference to the theoretical speed-up for the inference points to the overhead cost and inefficiencies in our CUDA implementation compared to the CuDNN implementation.

5 Application to Classification

The generality of DAU-ConvNets is demonstrated on several computer vision tasks with state-of-the-art CNN architectures. In this section, DAU-ConvNets are empirically analyzed on the ILSVRC 2012 classification task using the AlexNet and ResNet models, while the following sections (Section 6 and Section 7) demonstrate application to semantic segmentation and de-blurring.

Table 4 Results on ILSVRC 2012 validation set using AlexNet architecture and corresponding number of parameters on convolutional layers. Top-1 accuracy reported.

Network architecture	Top-1 accuracy (%)	Number of parameters on conv. layers
DAU-AlexNet	56.89	**2.3** million
AlexNet	**56.99**	3.7 million

5.1 AlexNet with DAUs

The first experiment involved evaluation on a classic architecture AlexNet (Krizhevsky et al. 2012) from Section 4.2. We compare the baseline AlexNet to Medium DAU-AlexNet (Table 3), which contains less than 70% of parameters than the baseline.

Table 4 reports accuracy for both methods together with the number of free parameters in the convolution layers. The DAU-AlexNet and the baseline AlexNet converge to comparable performance, close to 57%. The DAU-version of AlexNet achieved comparable performance to the classical AlexNet with over 30% fewer parameters and analysis in Section 4.2 shows that further reduction is possible at negligible performance loss. The overall comparable performance supports the hypothesis that DAUs do not lose expressive power on the account of their simple functional form.

5.2 Residual Networks with Displaced Aggregation Units

Next, we evaluated DAUs on ResNet50 and ResNet101 (He et al. 2016a) classification architectures. In particular, we evaluated ResNet v2 (He et al. 2016b), which applies batch normalization and activation before convolution for learning stabilization.

ResNet was modified into a DAU-ResNet by replacing all 3 × 3 convolutions with DAU convolution filters containing only two units (see Figure 2). This includes all layers except the first layer with 7 × 7 kernels and bottleneck layers with 1 × 1 kernels. We also implemented down-sampling with max-pooling instead of using convolutions with a stride[4]. This was performed on all levels except on the first one, where standard convolution was retained. The same down-sampling with max-pooling was included in the standard ResNet for a fair comparison. In DAU-ResNet, the displaced

[4] DAU layers with stride operation are not yet implemented.

Table 5 Results on segmentation task using a PASCAL VOC 2011 validation set. Per-class mean-IU and averaged mean-IU over all classes are reported.

Network arch.	bg	arpln.	bcyle.	bird	boat	bottle	bus	car	cat	chair	cow	d.tble	dog	horse	m.bike	person	p.plant	sheep	sofa	train	tv.	mIoU
DAU-AlexNet	**86.1**	**58.5**	**29.7**	**55.0**	**41.7**	**47.2**	**61.3**	**56.3**	**57.9**	**14.1**	**47.1**	**27.3**	47.8	**36.7**	**54.7**	**63.9**	**28.9**	53.0	**19.3**	59.8	**45.3**	**47.22**
AlexNet-dilation	85.8	54.6	27.2	51.8	39.0	45.2	56.3	54.2	57.4	12.4	43.8	26.1	**50.6**	35.6	54.1	61.1	26.9	**53.6**	18.9	**60.2**	42.5	45.57

Table 6 Results on ILSVRC 2012 validation set using deep residual network architecture and corresponding number of parameters on convolutional layers. Top-1 accuracy reported.

Network arch.	Top-1 acc. (%)	Number of parameters (in million)		
		Conv/DAU	Bottlenecks	Total
ResNet50	**74.08**	11.3 M	14.2 M	25.5 M
DAU-ResNet50	74.06	**7.5** M	14.2 M	**21.7** M
ResNet101	**75.39**	21.3 M	23.1 M	44.5 M
DAU-ResNet101	74.89	**14.2** M	23.1 M	**37.4** M

aggregation units were initialized randomly with uniform distribution on interval $[-1.5, 1.5]$, following the observation of displacement distribution in Section 4.3. Units were restricted to move up to 4 pixels away from the center, resulting in receptive field size of up to 9×9 pixels relative to the previous layer. This restriction was enforced only due to technical limitations in current DAU implementation.

Optimization. Both architectures were trained by stochastic gradient descent. The same optimization hyper-parameters were used in DAU and classic ResNet, i.e., learning rate of 0.1, momentum of 0.9, weight decay of 10^{-4} and a batch size of 256. Learning rate was reduced four times by a factor of 10 at 30th, 60th, 80th and 90th epoch. In DAUs, the weight decay was applied only to weights but not to offset; however, 500-times larger learning rate for offset was used during the training to compensate for orders of magnitude different values compared to the weights.

Classification Results with ResNet. Results for networks with 50 and 101 layers are reported in Table 6. The DAU version achieves the same performance as the classical ConvNet counterparts on ResNet50 as well as ResNet101. This result is achieved with a 30% reduction of parameters allocated for convolutions in spatial coverage of DAU-ResNet. The reduction in the overall number of parameters is slightly lower since the residual network allocates half of the parameters for 1×1 bottleneck layers, which are not replaced with DAUs.

6 Application to Semantic Segmentation

Classic ConvNet architectures designed for classification require hand-crafted structural modifications in the form of hand-tuned dilated convolutions to achieve high-quality results on other task like semantic segmentation. In particular, dilated convolution with several manually set dilation sizes are placed at certain layers in the ResNet when adapting for semantic segmentation (Chen et al. 2016a). Such changes are not required for DAU counterparts, since these simultaneously learn the filter receptive field sizes and content to the task at hand. A semantic segmentation task on PASCAL VOC 2011 and Cityscape datasets using three popular deep learning architectures, AlexNet, ResNet101 and DeepLab was chosen to demonstrate this.

6.1 Semantic Segmentation with AlexNet

We first evaluated a classic architecture – the AlexNet model. An AlexNet architecture modified for semantic segmentation from Section 4.3 was evaluated on PASCAL VOC 2011 segmentation dataset. Modification includes increased resolution at the last two layers and scaled displacements in the corresponding DAU convolution filters. The baseline AlexNet was similarly modified for the semantic segmentation task, but instead of scaling the displacements, we dilated convolution filters with the same factor. The layers after the first removed max-pooling use a dilation of two (layers 3, 4 and 5) and the layers after the second-removed-max-pooling use a dilation factor of four (layers 6 and 7).

Segmentation results. The performance of DAU-AlexNet compared to the baseline AlexNet with dilation is shown in Table 5. DAU-AlexNet consistently outperforms the baseline AlexNet with dilation across all measures. The mean IoU and per-pixel accuracy are improved by approximately 2%. Looking at the per-class mean IU, we observe the improvement is consistent over all categories, with the exception of ”dog”, ”sheep” and ”train”.

6.2 Semantic Segmentation with Residual Networks

DAUs were further evaluated on a very deep residual network with 101 layers (ResNet101) (He et al. 2016a). ResNet101 was modified in the same way as AlexNet in the

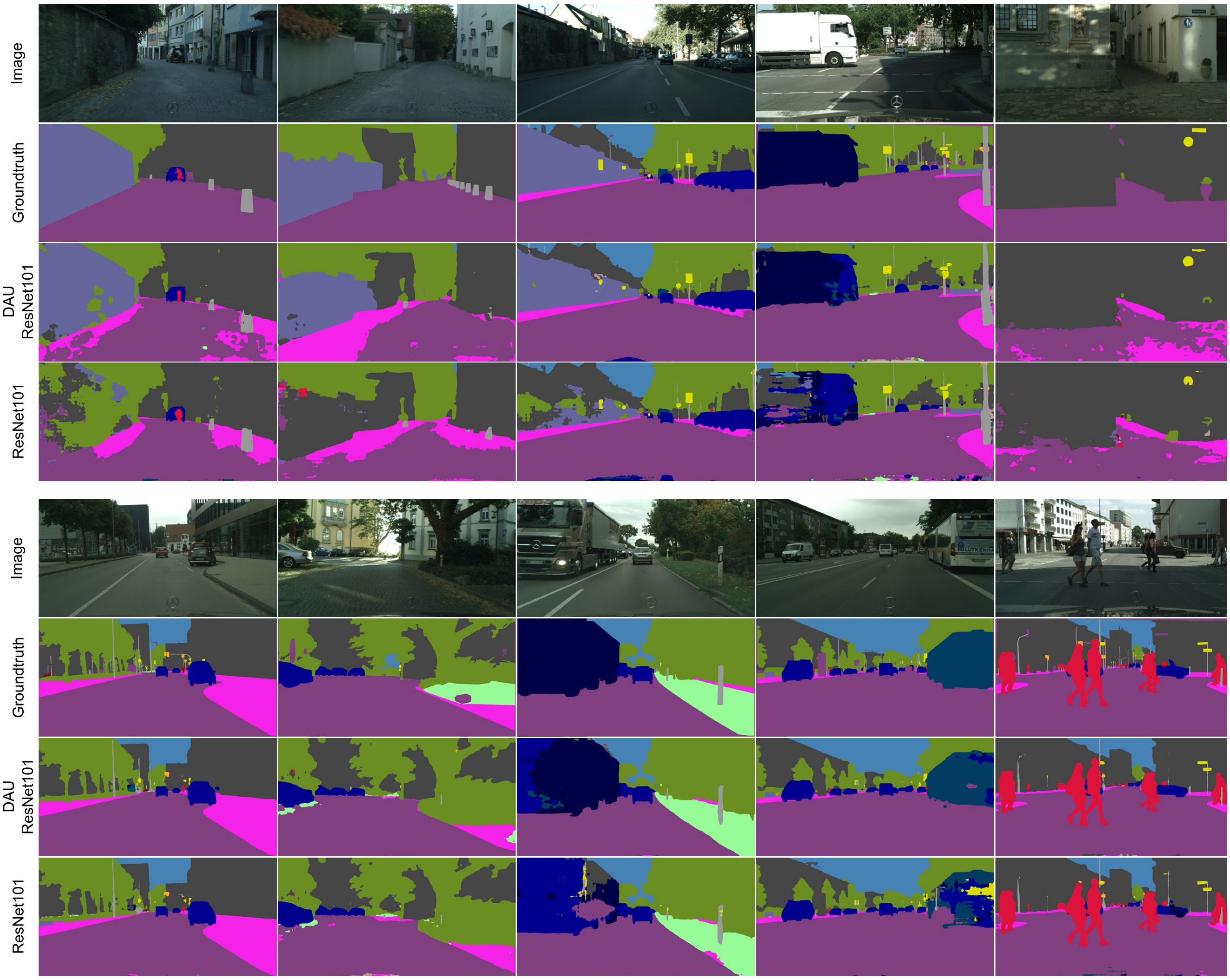


Fig. 9 Examples of semantic segmentation on Cityscape dataset with DAU-ResNet101 in the third and seventh row and standard ResNet101 fourth and eigth row.

previous section. This included removal of the last max-pooling layer, which resulted in a network with output layer resolution reduced by 16× (as opposed to 32× reduction in the original ResNet101). This matches to having the output stride of 16 as in the DeepLab model (Chen et al. 2016a).

Cityscape dataset. The Cityscape dataset (Cordts et al. 2016) was used for evaluation. The dataset contains high-resolution images of city driving and the task requires pixel-wise segmentation of the image into 19 classes. Only fine-grained annotations were used (i.e., 2,975 training images) and the networks were evaluated on 500 test images from the validation set.

Optimization. The standard ResNet and DAU-ResNet were first pre-trained on ImageNet (Russakovsky et al. 2015). Both models were then trained for segmentation using a mini-batch stochastic gradient descent with a batch size of 8 for 50,000 iterations (134 epochs). A learning rate of 0.01 was used, with momentum of 0.9 and a weight decay of 10^{-4}. A polynomial decay of the learning rate with a factor of 0.9 was applied. Data augmentation was used with the following operations: images were resized by a factor randomly selected from a uniform distribution in a range of $[0.5, 2.0]$ and high-resolution images were randomly cropped into 769×769 large patches, and left-to-right mirroring was applied with a probability of 0.5. Testing was performed on a single-scale without multi-scale testing.

Results. Results are reported in the first row in Table 7. DAU-ResNet101 achieves 72.8% mIoU and outperforms the standard ResNet101 (68.6% mIoU). A similar difference is observed in the mean accuracy – standard ResNet101 obtains 77.2% mean accuracy, while the DAU version obtains a 82.2% mean accuracy. Improvement of around 4% clearly demonstrates the benefits of having learnable unit displace-

Table 7 Results on Cityscape validation set using deep residual network architecture and DeepLabv3+ improvements. We report mean intersection-over-union (mIoU). DAU-6U is a single displaced aggregation layer with 6 units per channel which replaces ASPP.

Standard ResNet101 backbone						*ResNet101 with DAUs (our) backbone*				
Output Stride	ASPP	Image-pool	Decoder	mIoU	+/-	mIoU	Output Stride	DAU-6U	Image-pool	DAU-Decoder
16				68.6	*+4.2*	**72.8**	16			
16		✓		72.7	*+0.1*	**72.8**	16			
16	✓	✓		**75.6**	*-0.1*	75.5	16	✓	✓	
16	✓	✓	✓	75.8	*+0.3*	**76.1**	16	✓	✓	✓

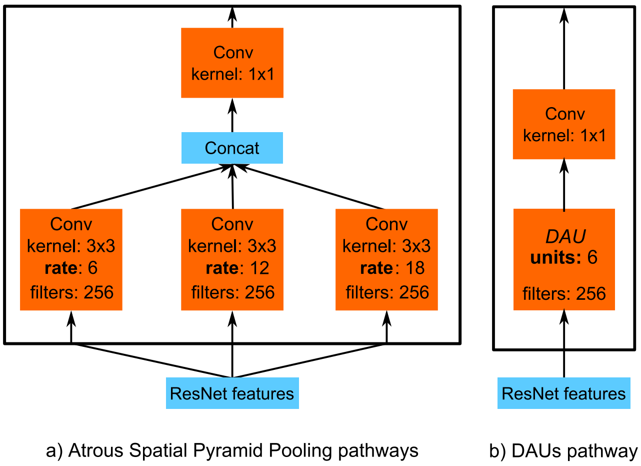
ments, which allow the network to focus on spatial features required for segmentation without requiring manual specification. Note that a 4% increase in the performance was achieved with 15% less parameters in the network. Several examples of semantic segmentation on both networks are depicted in Figure 9. Several top examples well demonstrate gridding artifacts in ResNet101 while DAUs avoid this issue.

6.3 Improving DeepLab with DAUs

Since DAUs inherently provide adjustable receptive field sizes, it becomes a natural fit for a popular semantic segmentation model, DeepLab (Chen et al. 2016b, 2017), where large receptive fields are achieved with hand-tuned dilation. For this experiment, we used the latest version of DeepLab v3+ (Chen et al. 2018) that incorporates the following improvements for semantic segmentation: (a) output stride of 16, (b) atrous spatial pyramid pooling (ASPP) layer, (c) global image-pooling features, and (d) an output decoder layer. As a backbone network, we used ResNet101 that was modified for a semantic segmentation problem from the previous subsection.

The DeepLab architecture was modified to include DAUs as follows. First, the convolution filters in ResNet were replaced by DAUs in the same manner as for the DAU-ResNet from the previous subsection. Next, the atrous spatial pyramid pooling (ASPP) (Chen et al. 2016b) with three parallel convolutions, each with a different dilation rate, was replaced by a single DAU layer with six units per kernel (termed as DAU-6U) as depicted in Figure 10. We used more units than on other layers to provide enough coverage for larger area. Receptive field size is thus adjusted dynamically during training, provided that large enough displacement of a unit is allowed. Lastly, the output decoder was implemented with DAUs (two units per convolution filter) instead of using 3 × 3 convolutions.

Dataset and Optimization. Optimization and evaluation of DeepLab was performed on the Cityscape dataset using the same hyper-parameters as in the previous subsection. The same process of augmentation was used with the input scaling, cropping and flipping. Testing was performed on *val* set and a single-scale without multi-scale testing.



a) Atrous Spatial Pyramid Pooling pathways

b) DAUs pathway

Fig. 10 a) Atrous Spatial Pyramid Pooling (ASPP) block processes the input features along several parallel pathways, each containing 256 convolution filters with a fixed dilation rate – the rates differ between the pathways. b) A single DAU pathway containing 256 DAU filters with 6 DAUs per channel outperforms ASPP and eliminates hand-tuning of the dilation rates using less parameters.

Results. Results are reported in Table 7. Results for several DeepLab versions are reported to quantify contributions of different improvements. Notice that standard ResNet101 becomes competitive with the DAU version only when image-pooling features are included. Since image-pooling features capture global information (i.e., context), this indicates that DAU convolution filters already capture most of the global information through their sparse and adjustable displacements. Results also show that a single DAU-6U layer provides a comparable performance boost to the ASPP with three hand-crafted parallel convolutions. Finally, implementing the decoder with DAUs also improves DeepLab slightly more than using a standard decoder and ASPP. In this case, the DAU version achieves a mIoU of 76.1%, while standard DeepLab achieves a mIoU of 75.8%.

Note that DAU-6U is a single layer with 6 units (18 parameters) while ASPP applies at least three parallel convolutions each with 9 parameters, resulting in at least 27 parameters. While ASPP was hand-crafted and would require separately testing various variations of the receptive field combinations to fine-tune its architecture to a given dataset, the DAU-6U learns them directly from the dataset, thus signif-

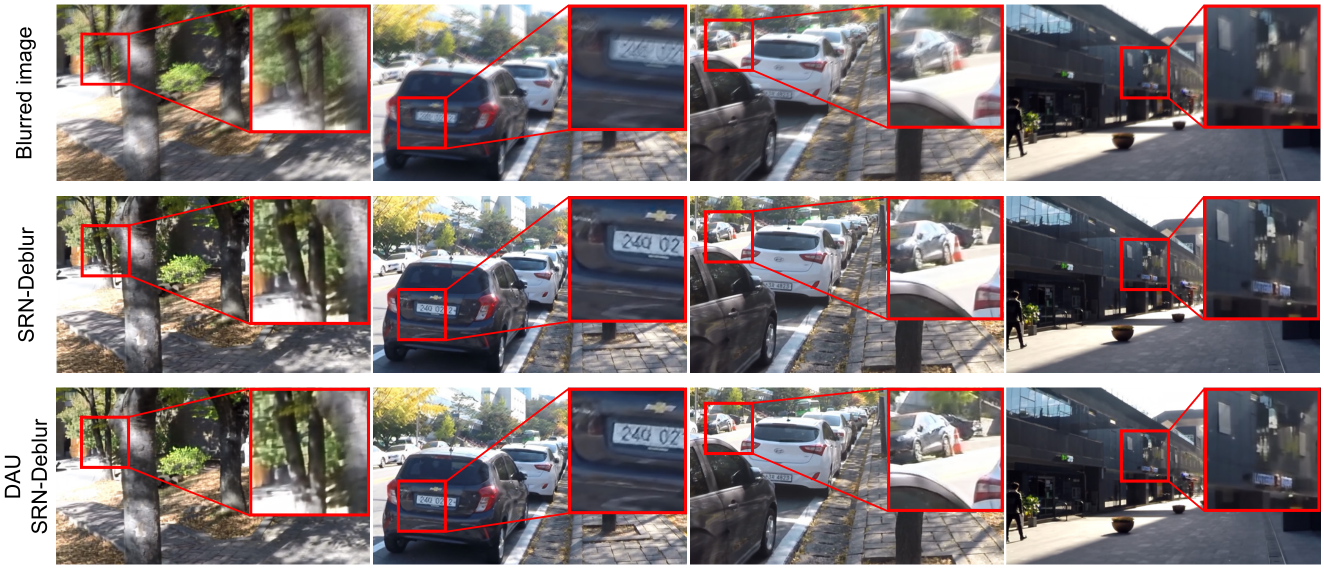


Fig. 11 Examples of de-blurring on GOPRO dataset with SRN-DeblurNet in the second row and DAU variant in the third row.

icantly reducing the complexity of designing high-performance networks.

7 Application to Blind Image De-blurring

As the last application example, we demonstrate the performance of DAUs on the task of blind image de-blurring where large receptive fields have proven to play an important role.

7.1 A Scale-Recurrent Network with DAUs

A scale-recurrent network by Tao et al. (2018), termed SRN-DeblurNet, is a state-of-the-art method for blind image debluring task. SRN-DeblurNet employs a 43-layer U-Net architecture in a scale-recurrent approach to perform a dense regression of each output pixel value. SRN-DeblurNet attains large receptive field sizes by down-sampling and 5×5 convolution kernels. The network obtains top performance on de-blurring benchmarks (Tao et al. 2018), but at a cost of inefficient use of parameters for the spatial coverage.

We propose a DAU-SRN-DeblurNet where 5×5 convolutions are replaced with two displaced aggregation units per convolution filter. The replacements are made in all but four layers: we retain two de-convolution layers and the first and the last layers as classical convolutions[5]. This results in a much more efficient network with 4× fewer parameters than SRN-DeblurNet, and in per-filter adapted receptive field sizes. A central requirement of the convolution filters in SRN-DeblurNet is to enable modeling the identity function which allows the network to pass through pixels that are not blurred. Thus the standard deviations in DAU weights were reduced to 0.35 to reduce their aggregation effect. According to further evaluation by Tao et al. (2018) after the paper acceptance[6], they removed the color ringing artifacts to further improve the performance by applying SRN-DeblurNet without LSTM to the RGB data. We used the same approach in our experiments.

The GOPRO Dataset. The GOPRO dataset (Nah et al. 2017) was used for training and testing. The dataset contains 2,103 pairs of training images and 1,111 pairs of testing images. Each pair consists of two colored images: a blurred image (input) and a sharp image (groundtruth), both in 1280×720 resolution.

Optimization. The training protocol of Tao et al. (2018) was followed. We trained with a mini-batch stochastic gradient descent using the Adam solver (Kingma and Ba 2015) for 2000 epochs with a batch size of 16 images. For SRN-DeblurNet, we used the best hyper-parameters provided by Tao et al. (2018), and a learning rate of 10^{-4} with a polynomial decay using a power factor of 0.3. The learning rate was increased for DAU-SRN-DeblurNet to $5 \cdot 10^{-4}$ to compensate for the smaller weights in DAUs due to normalized Gaussian blurring. Furthermore, since the unit displacement values are several orders of magnitude larger than the weights, we also increased the learning rate for displacement values μ to 10^{-3} and applied a linear decay, i.e., a polynomial decay with a power factor of 1.0. The trainable variables were initialized with the Glorot and Bengio (2010) method. Displacement values of DAUs were initialized randomly with a

[5] Current implementation of DAUs requires an even number of channels.

[6] https://github.com/jiangsutx/SRN-Deblur

Table 8 Results on the GOPRO dataset using the reference and DAU-based SRN-DeblurNet architecture with reported peak-signal-to-noise-ratio (PSNR) and the number of trainable parameters (in million).

Network architecture	PSNR (dB)	Number of params.	Number of units
SRN-DeblurNet	**30.07**	6.878 M	6.878 M
DAU-SRN-DeblurNet	30.02	**1.781** M	**0.708** M

zero-mean normal distribution with $\sigma = 0.5$. Data augmentation was not used in training but images were randomly cropped into 256×256 patches to fit them into the memory. This followed the learning protocol of Tao et al. (2018).

Results. Results are reported in Table 8. Both methods achieved a peak-signal-to-noise-ratio (PSNR) of slightly above 30 dB. Note that SRN-DeblurNet required 6.8 million parameters, while DAU-SRN-DeblurNet required only 25% of parameters (1.7 million) for the same performance. The difference is even more substantial when considering the number of units required for spatial coverage (see Table 8) – in this case DAU-SRN-DeblurNet requires only 10% of units compared to SRN-DeblurNet. Examples of de-blurred images with both methods are shown in Figure 11.

We have also observed that a larger aggregation perimeter (i.e., larger standard deviation of DAUs) did not significantly affect the performance. DAUs with $\sigma = 0.5$ achieved PSNR of 29.84 dB, while DAUs with $\sigma = 0.35$ resulted in PSNR of 30.02 dB. Considering that the increased aggregation perimeter introduces significant feature blurring, a larger performance difference might be expected. The small difference points to an effective and robust DAU structure that is able to compensate for the added blurring effect, which is particularly important in the deblurring task.

8 Discussion and Conclusion

We proposed DAU convolution filters to replace fixed grid-based filters in classical convolutional networks. The DAUs modify only the convolutional layer in standard ConvNets, can be seamlessly integrated into existing architectures, and afford several advantages. In addition to the filter unit weights, they allow learning the receptive field size. Since the number of parameters is decoupled from the receptive field size, they efficiently allocate the free parameters, resulting in compact models and efficient learning. In addition, DAUs eliminate the need for hand-crafted convolution filter patterns (e.g., dilated convolutions) and allow automatic adaptation for a broad spectrum of computer vision tasks.

The parameter reduction capability was demonstrated on classification, semantic segmentation and blind image de-blurring. In particular, experiments with the AlexNet (Krizhevsky et al. 2012) architecture on a classification task have shown that DAUs achieved a similar performance to the standard network with only 30% of the parameters. Similar improvement has been demonstrated on a state-of-the-art de-blurring method, SRN-DeblurNet (Tao et al. 2018), where the same performance has been achieved with only 25% of the parameters. With only three free parameters per unit, this shows that networks can potentially allocate an order of magnitude fewer units for providing sufficient spatial coverage and that existing deep learning methods are inefficiently using their parameters.

The experiments on semantic segmentation have further shown that DAUs trained for one task enable a straightforward adaptation to another task by using the same architectural model and only learning new parameters for the new task. We have demonstrated this by adapting a DAU-based residual network (He et al. 2016a) with the architecture for the classification to the semantic segmentation. The experiment on Cityscape dataset has shown improvement in the performance of the DAU-ResNet model by 4% compared to standard ResNet without significant modifications to the network. This was achieved while using 15% fewer parameters. The classic ResNet has become competitive to DAUs only after global image-pooling features have been added. This means that DAUs already capture the contextual information through position adaptation, which has to be added manually by architectural change in the standard network.

Experiments show that DAUs can completely replace atrous spatial pyramid pooling (ASPP) in DeepLab (Cheng et al. 2014). By adjusting displacement, DAUs were able to selectively focus on spatial areas of sub-features that are important for specific tasks. This was demonstrated on semantic segmentation where DeepLab with only a single extra DAU layer was able to fully replace several parallel convolutions in ASPP that use different dilation factors. This was achieved at the same or slightly better performance while using 15% fewer parameters. More importantly, DAUs removed the need for hand-tuning the dilation factors in DeepLab. Thus, they enable learning without repeating extensive experiments to hand-tune dilation for a new domain.

DAUs seamlessly integrate into existing state-of-the-art architectures with plug-and-play capability by simply replacing the standard convolution layers. We have published CUDA implementations for Caffe and TensorFlow frameworks and plan to release all the DAU versions of state-of-the-art architectures reported in this work, making all results in this work fully reproducible.

An active area of exploration in the deep learning community is development of mathematical tools for formal analysis of ConvNet properties. Such analysis is very difficult with the classical ConvNet formulation with discrete convolution filters that can take arbitrary values, and simplifications of the model have to be made. A highly interesting analysis is the work of Bruna and Mallat (2013) who treat ConvNets from a spectral perspective. Note that DAUs also provide a new formal view of the ConvNet pipeline. In their simplest variant with a single unit per convolution filter, DAU-ConvNets can be considered as a sequence of feature low-pass filtering (blurring) and spatial shifting with intermediate nonlinearities. Our results show that even this simplest formulation achieves comparable performance to the classical ConvNets, but is more tractable. We expect that this mathematically simpler formulation will open new venues for further theoretical analysis of deep models.

9 Acknowledgements

The authors would like to thank Hector Basevi for his valuable comments and suggecstion on improving the paper. This work was supported in part by the following research projects and programs: project GOSTOP C3330-16-529000, DIVID J2-9433 and ViAMaRo L2-6765, program P2-0214 financed by Slovenian Research Agency ARRS, and MURI project financed by MoD/Dstl and EPSRC through EP/N019415/1 grant. We thank Vitjan Zavrtanik for his contribution in porting the DAUs to the TensorFlow framework.

References

Amidror I (2013) Mastering the Discrete Fourier Transform in One, Two or Several Dimensions. Springer 2

Bruna J, Mallat S (2013) Invariant Scattering Convolution Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence 35(8):1872–86, DOI 10.1109/TPAMI.2012.230, URL http://www.ncbi.nlm.nih.gov/pubmed/23787341, 1203.1513 16

Chang J, Gu J, Wang L, MENG G, XIANG S, Pan C (2018) Structure-Aware Convolutional Neural Networks. In: Procedings of the Neural Information Processing Systems, pp 1–10 3

Chen LC, Papandreou G, Kokkinos I, Murphy K, Yuille AL (2016a) DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. Pattern Analysis and Machine Intelligence pp 1–14, URL http://arxiv.org/abs/1606.00915, 1606.00915 1, 12, 13

Chen LC, Papandreou G, Kokkinos I, Murphy K, Yuille AL (2016b) DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs pp 1–14, DOI 10.1109/TPAMI.2017.2699184, URL http://arxiv.org/abs/1606.00915, 1606.00915 1, 2, 9, 14

Chen LC, Papandreou G, Schroff F, Adam H (2017) Rethinking Atrous Convolution for Semantic Image Segmentation URL http://arxiv.org/abs/1706.05587, 1706.05587 3, 14

Chen Lc, Zhu Y, Papandreou G, Schroff F (2018) Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. In: European Conference on Machine Learning: Workshop on Music and Machine Learning 14

Cheng MM, Zhang Z, Lin WY, Torr P (2014) BING: Binarized Normed Gradients for Objectness Estimation at 300fps. In: Computer Vision and Pattern Recognition, Ieee, pp 3286–3293, DOI 10.1109/CVPR.2014.414, URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6909816 16

Cordts M, Omran M, Ramos S, Rehfeld T, Enzweiler M, Benenson R, Franke U, Roth S, Schiele B (2016) The Cityscapes Dataset for Semantic Urban Scene Understanding. In: Computer Vision and Pattern Recognition, DOI 10.1109/CVPR.2016.350, URL http://arxiv.org/abs/1604.01685, 1604.01685 13

Dai J, Qi H, Xiong Y, Li Y, Zhang G, Hu H, Wei Y (2017) Deformable Convolutional Networks. In: International Conference on Computer Vision, DOI 10.1051/0004-6361/201527329, URL http://arxiv.org/abs/1703.06211, 1703.06211 3

Eigen D, Rolfe J, Fergus R, Lecun Y (2014) Understanding Deep Architectures using a Recursive Convolutional Network pp 1–9, arXiv:1312.1847v2 3, 7

Everingham M, Van Gool L, Williams CKI, Winn J, Zisserman A (2011) The PASCAL Visual Object Classes Challenge 2011 (VOC2011) Results. http://www.pascal-network.org/challenges/VOC/voc2011/workshop/index.html 9

Glorot X, Bengio Y (2010) Understanding the difficulty of training deep feedforward neural networks. Aistats 9:249–256, DOI 10.1.1.207.2059 6, 8, 15

Hariharan B, Arbel P, Bourdev L, Maji S, Malik J, Berkeley UC, Systems A, Ave P, Jose S (2011) Semantic Contours from Inverse Detectors. In: International Conference on Computer Vision 9

He K, Zhang X, Ren S, Sun J (2014) Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. In: European Conference on Computer Vision, pp 346–361, DOI 10.1109/TPAMI.2015.2389824, URL http://arxiv.org/abs/1406.4729v1{%}5Cnpapers3://publication/uuid/09415D06-A785-4329-82C0-B9FF2B1FEAB7, 1406.4729 6

He K, Zhang X, Ren S, Sun J (2016a) Deep Residual Learning for Image Recognition. In: CVPR, pp 171–180, DOI 10.3389/fpsyg.2013.00124, 1512.03385 1, 11, 12, 16

He K, Zhang X, Ren S, Sun J (2016b) Identity mappings in deep residual networks. In: European Conference on Computer Vision, vol LNCS-9908, pp 630–645, DOI 10.1007/978-3-319-46493-0_38, 1603.05027 11

Holschneider M, Kronland-Martinet R, Morlet J, Tchamitchian P (1990) A Real-Time Algorithm for Signal Analysis with the Help of the Wavelet Transform. In: Combes JM, Grossmann A, Tchamitchian P (eds) Wavelets, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 286–297 7

Iandola FN, Han S, Moskewicz MW, Ashraf K, Dally WJ, Keutzer K (2016) SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size pp 1–13, DOI 10.1007/978-3-319-24553-9, URL http://arxiv.org/abs/1602.07360, 1602.07360 3

Jacobsen JH, van Gemert J, Lou Z, Smeulders AWM (2016) Structured Receptive Fields in CNNs. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) pp 2610–2619, DOI 10.1109/CVPR.2016.286, URL http://arxiv.org/abs/1605.02971, 1605.02971 3

Jaderberg M, Vedaldi A, Zisserman A (2014) Speeding up Convolutional Neural Networks with Low Rank Expansions. In: British Machine Vision Conference, p 7, DOI 10.5244/C.28.88, arXiv:1405.3866v1 3

Jeon Y, Kim J (2017) Active Convolution: Learning the Shape of Convolution for Image Classification DOI 10.1109/CVPR.2017.200, URL http://arxiv.org/abs/1703.09076, 1703.09076 2, 3

Kaiming H, Gkioxara G, Dollar P, Girshick R (2017) Mask R-CNN. In: International Conference on Computer Vision, pp 2961–2969, 1703.06870 1, 3

Kingma DP, Ba JL (2015) Adam: a Method for Stochastic Optimization. In: International Conference on Learning Representations, pp 1–13, arXiv:1412.6980v5 15

Krizhevsky A (2009) Learning Multiple Layers of Features from Tiny Images. Science Department, University of Toronto, TechReport pp 1–60 6

Krizhevsky A, Sutskever I, Hinton GE (2012) ImageNet Classification with Deep Convolutional Neural Networks. In: Advances In Neural Information Processing Systems 25, pp 1097–1105, 1102.0183 8, 11, 16

LeCun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. Proceedings of the IEEE 86(11):2278–2323, DOI 10.1109/5.726791 4

Long J, Shelhamer E, Darrell T (2015) Fully Convolutional Networks for Semantic Segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, vol 8828, pp 3431–3440, DOI 10.1109/CVPR.2015.7298965, 1411.4038 9

Luan S, Zhang B, Chen C, Cao X, Ye Q, Han J, Liu J (2017) Gabor Convolutional Networks. British Machine Vision Conference pp 1–12, URL http://arxiv.org/abs/1705.01450, 1705.01450 3

Luo P, Wang G, Lin L, Wang X (2017) Deep Dual Learning for Semantic Image Segmentation. Computer Vision and Pattern Recognition (CVPR) pp 2718–2726, DOI 10.1109/ICCV.2017.296, URL http://personal.ie.cuhk.edu.hk/{~}pluo/pdf/luoWLWiccv17.pdf 10

Luo W, Li Y, Urtasun R, Richard Z (2016) Understanding the Effective Receptive Field in Deep Convolutional Neural Networks. In: NIPS, 1701.04128 1, 3

Nah S, Kim TH, Lee KM (2017) Deep Multi-scale Convolutional Neural Network for Dynamic Scene Deblurring. In: Computer Vision and Pattern Recognition, pp 3883–3891, DOI 10.1109/CVPR.2017.35, 1612.02177 15

Redmon J, Farhadi A (2017) YOLO9000: Better, Faster, Stronger DOI 10.1109/IJCNN.2015.7280696, URL http://arxiv.org/abs/1612.08242, 1612.08242 1

Ronneberger O, Fischer P, Brox T (2015) U-Net: Convolutional Networks for Biomedical Image Segmentation. In: Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015, pp 234–241, DOI 10.1007/978-3-319-24574-4_28, 1505.04597 1, 2

Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, Huang Z, Karpathy A, Khosla A, Bernstein M, Berg AC, Fei-Fei L (2015) ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision 115(3):211–252, DOI 10.1007/s11263-015-0816-y, URL http://dx.doi.org/10.1007/s11263-015-0816-y, 1409.0575 7, 8, 13

Shelhamer E, Long J, Darrell T (2016) Fully Convolutional Networks for Semantic Segmentation. Pattern Analysis and Machine Intelligence pp 1–12, DOI 10.1109/TPAMI.2016.2572683, URL http://arxiv.org/abs/1605.06211, 1605.06211 1

Simonyan K, Zisserman A (2015) Very Deep Convolutional Networks for Large-Scale Image Recoginition. In: International Conference on Learning Representations, pp 1–14, arXiv:1409.1556v6 1

Tabernik D, Kristan M, Wyatt JL, Leonardis A (2016) Towards Deep Compositional Networks. In: International Conference on Pattern Recognition, URL http://arxiv.org/abs/1609.03795, 1609.03795 3, 5, 7

Tabernik D, Kristan M, Leonardis A (2018) Spatially-Adaptive Filter Units for Deep Neural Networks. In: Computer Vision and Pattern Recognition, pp 9388–9396, URL http://arxiv.org/abs/1711.11473, 1711.11473 3

Tao X, Gao H, Wang Y, Shen X, Wang J, Jia J (2018) Scale-recurrent Network for Deep Image Deblurring. In: Computer Vision and Pattern Recognition, pp 8174–8182, DOI 10.1109/CVPR.2018.00853, URL http://arxiv.org/abs/1802.01770, 1802.01770 1, 15, 16

Xie S, Girshick R, Dollár P, Tu Z, He K (2017) Aggregated Residual Transformations for Deep Neural Networks. In: Conference on Computer Vision and Pattern Recognition, DOI 10.1109/CVPR.2017.634, URL http://arxiv.org/abs/1611.05431, 1611.05431 1

Yu F, Koltun V, Funkhouser T (2017) Dilated Residual Networks. In: Computer Vision and Pattern Recognition, 1705.09914 2