Spatially-Adaptive Filter Units for Deep Neural Networks

Domen Tabernik[1], Matej Kristan[1] and Aleš Leonardis[1,2]
[1]Faculty of Computer and Information Science, University of Ljubljana, Ljubljana, Slovenia
[2]CN-CR Centre, School of Computer Science, University of Birmingham, Birmingham, UK
{domen.tabernik,matej.kristan}@fri.uni-lj.si
a.leonardis@cs.bham.ac.uk

Abstract

Classical deep convolutional networks increase receptive field size by either gradual resolution reduction or application of hand-crafted dilated convolutions to prevent increase in the number of parameters. In this paper we propose a novel displaced aggregation unit (DAU) that does not require hand-crafting. In contrast to classical filters with units (pixels) placed on a fixed regular grid, the displacement of the DAUs are learned, which enables filters to spatially-adapt their receptive field to a given problem. We extensively demonstrate the strength of DAUs on a classification and semantic segmentation tasks. Compared to ConvNets with regular filter, ConvNets with DAUs achieve comparable performance at faster convergence and up to 3-times reduction in parameters. Furthermore, DAUs allow us to study deep networks from novel perspectives. We study spatial distributions of DAU filters and analyze the number of parameters allocated for spatial coverage in a filter.

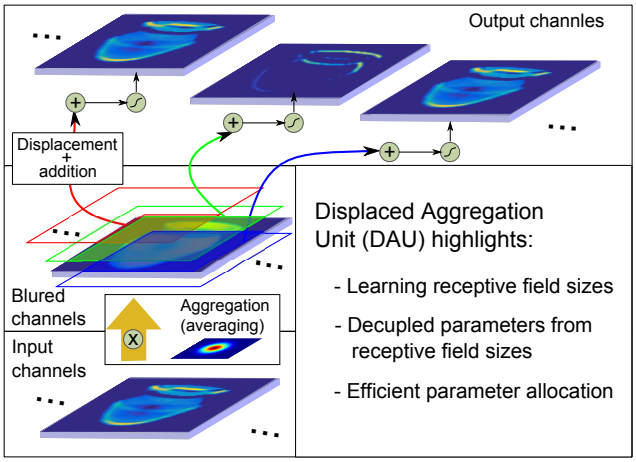


Figure 1: The displaced aggregation units (DAUs) afford efficient implementation. Convolution of a feature channel with a filter composed of several DAUs is implemented as blurring by a single Gaussian and subsampling at learned displacements.

1. Introduction

Deep convolutional neural networks (ConvNet) [10, 25, 21, 14] have become prevalent in visual feature learning. The integral part of these approaches are convolutional filters. In combination with other layers, the definition of the filter directly influences the kind of features a network can capture. Current state-of-the-art ConvNets define filters as rectangular windows of weights where each learnable unit is a single pixel value in the filter.

An important hyperparameter of the filters is their size, which is directly related to the number of free parameters in ConvNets. Large filters are avoided in the interest of keeping this number low and reducing overfitting. On the other hand, feature expressiveness improves with increased receptive fields [2]. Classification networks thus apply small filters and implicitly increase the receptive field size by gradually reducing resolution via pooling layers and increased depth [23]. But in dense prediction problems like segmentation [18, 3], sufficient resolution is required for accurate localization of segmentation boundaries. Thus large receptive fields have to explicitly be accounted for without resolution loss [4, 26, 27].

Increasing a receptive field without sacrificing resolution is addressed by dilated (atrous) convolution [26, 27]. This approach increases the kernel receptive field by spreading out (dilating) the positions of the kernel sampling units (i.e., pixels). Large dilations significantly violate Nyquist theorem [1], resulting in griding artifacts [27]. Mitigation of these requires additional convolutional layers with progressively smaller dilations. The dilation factors are another hyperparameter that is manually tuned. To alleviate manual specification to some extent, [3, 4] propose to use several pre-selected dilation factors and achieve excellent results.

We define a convolutional filter as a mixture of several displaced aggregation units which is a generalization of the convolutional layers typically used for classification

and those for segmentation. In contrast to a standard filter, the aggregation unit is not a single pixel, but a locally averaged response. In our implementation, a Gaussian with a fixed variance is used for averaging. In contrast to standard ConvNets, our units are not positioned on a regular grid. Their displacements are adapted during learning, thus the receptive field size of each filter is tuned separately. This allows for large or small receptive fields without changing the number of parameters, facilitating automatic and efficient allocation of parameters.

Our major contribution is an efficient formulation of the *displaced aggregation units* (DAU) filter with sub-pixel displacements, which allows practical use in deep architectures (see Fig. 1). The DAUs remove the requirement for hand-crafted dilation without modification of other layers, decouple the parameter count from the receptive field size and do not suffer from gridding effects. Backpropagation is derived for all parameters and the new layers are implemented in standard ConvNet package with low-level CUDA procedures [13]. Our secondary contributions are analyses that have not been possible with the existing networks. We demonstrate that a *single* type of DAU-based filters achieve comparable performance to standard ConvNets on classification as well as dilated ConvNets on a segmentation task. We perform analysis of the dilation patterns required for accurate segmentation by recording the distributions of the *learned* displacements in DAUs. Our parameter study demonstrates that using only few DAUs per filter already results in excellent performance. Our tests also show that DAUs allow comparable performance to classical ConvNets at almost 3-fold reduction of the learned parameters in convolutional layers.

The remainder of this paper is structured as follows: in Sec. 2 we review most closely related works, we describe DAU in Sec. 3 and evaluate our model in Sec. 4. In Sec. 5 we present a comprehensive study of DAU filter displacements and conclude with a discussion in Sec. 6.

2. Related work

Receptive field has been considered as an important factor for deep networks in several related works [20, 4]. Luo et al. [20] measured an effective receptive field in convolutional neural networks and observed it increases as the network learns. They suggest an architectural change that forego a rectangular windows of weights for a 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. [5] and Jeon et al. [12]. Dai et al. [5] implemented spatial deformation of features with deformable convolutional networks. They explicitly learn feature displacement but learn them on a per-pixel location basis for input activation map and share them between all channels and features. Our model instead learns different displacements for different channels and features, and shares them over all pixel locations in the input activation map. This makes our model complementary to deformable convolutions. Jeon et al. [12], on the other hand, apply deformation on filter units similarly as we do. They use bilinear interpolation similar to ours to get displacements at a sub-pixel accuracy but they apply them to 3×3 filters. However, they do not learn different offsets for each channel but apply the same offset across all channels and features. This prevents them from decreeing their parameter count as they still use 9 units per filter. We show that significantly less units are needed.

Works by Luan et al. [19] and Jacobsen et al. [11] changed 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 provide displacements of filter units to arbitrary values. Their models have a fixed receptive fields defined as a hyperparameter 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.

Our model also uses concepts for filter parametrization similar to Tabernik et al. [24] but differs significantly in their design. The model by Tabernik et al. [24] is limited to only small scale networks and implements only a shallow network with two convolutional layers due to inefficient parametrization design. Our proposed model enjoys an efficient parametrization and we apply it to larger problems using deeper networks.

3. Displaced aggregation units (DAU)

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

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

where 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 [17].

We define the filters W_s as mixtures of localized aggregated feature responses from the input feature map. We choose Gaussians as an analytic form of aggregation units and compactly write filter as $W_s = \sum_k w_k G(\boldsymbol{\mu}_k; \sigma)$, where the unit displacement and aggregation range are specified by the mean $\boldsymbol{\mu}_k$ and variance σ^2, respectively, and w_k is the input amplification factor. With the exception of variance σ^2, the parameters $\boldsymbol{\mu}_k$ and w_k are unique for each output feature i and channel s, however, we omit this in notation

for clarity. Note that mixtures of Gaussians have recently been explored as potential filters in [24]. But due to the computational complexity of adapting all parameters, the approach was not feasible beyond a two-layer architecture.

In our preliminary study we noticed that while the unit locations play a crucial role in the shallow network performance, the variances do not. We thus make all variances in the Gaussians equal and fixed to a selected value, making the unit aggregation perimeter a single hyperparameter.

3.1. Inference with DAU

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

$$f * G(\boldsymbol{\mu}_k; \sigma) = f * \mathcal{T}_{\boldsymbol{\mu}_k}[G(\sigma)] \quad (2)$$
$$= \mathcal{T}_{\boldsymbol{\mu}_k}[f * G(\sigma)], \quad (3)$$

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

$$Y_i = f\left(\sum_s \sum_k w_k \mathcal{T}_{\boldsymbol{\mu}_k}(G(\sigma) * X_s) + b_s\right). \quad (4)$$

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.

Note that due to discretization, Eq. (4) is accurate only for discrete displacements $\boldsymbol{\mu}_k$. We address this by redefining the translation function in Eq. (4) as a bilinear interpolation

$$\mathcal{T}_x(g, y) = \sum_i \sum_j a_{i,j} \cdot g(y - \lfloor x \rfloor + [i, j]), \quad (5)$$

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

3.2. Learning DAU filter

The DAU contains two learnable parameters: the input amplification w_k and the spatial displacement μ_k. In principle, the shared aggregation perimeter σ could be learned as well, but we found that fixing this value was sufficient in our experiments. Thus the hyperparameters in DAU filters are the aggregation perimeter and the number of DAUs per filter.

Since DAUs are analytic functions, the filter parameters are fully differentiable and conform with the standard ConvNet gradient-descent learning techniques with backpropagation. The required partial derivatives are

$$\frac{\partial l}{\partial w_k} = \sum_{n,m} \frac{\partial l}{\partial z} \cdot \sum_{\boldsymbol{x}} \mathcal{T}_{\boldsymbol{\mu}_k}(X_s * G(\sigma)), \quad (6)$$

$$\frac{\partial l}{\partial \mu_k} = \sum_{n,m} \frac{\partial l}{\partial z} \cdot \sum_{\boldsymbol{x}} w_k \cdot \mathcal{T}_{\boldsymbol{\mu}_k}(X_s * \frac{\partial G(\sigma)}{\partial \mu}), \quad (7)$$

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

Similarly to inference in Sec. 3.1, the gradient can efficiently be computed using convolution with zero-mean Gaussian (or derivatives) and sampling the response at displacement specified by the mean values in the DAUs. This significantly reduces the computational cost compared to the explicit mixture model filters [24].

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. (4) can be applied for computing the back-propagated error as well, yielding efficient and fast computation.

4. Performance evaluation

This section reports results of the experimental evaluation of DAUs. We first analyze the influence of the hyperparameters on DAU filters and then evaluate our approach by replacing the standard filters in ConvNets with DAUs filters for a classification task (Sec. 4.2) and a segmentation task (Sec.4.3). Sec. 5 reports analysis of the learned filter receptive fields and how the number of DAUs per filter impacts the ConvNet performance.

4.1. Hyperparameter analysis in DAU-ConvNets

We analyze the influence of two hyperparameters on our network: (a) variance σ^2 used in aggregation and (b) the number of DAUs per filter. We analyzed both on classification problem using CIFAR10 [15] dataset.

For the purpose of this evaluation we used a shallow network with only three convolutional layers with DAU filters and three max-pooling layers. To classify the whole image we appended fully-connected layer. Batch normalization [9] was applied to convolutional layers and weights were initialized using [7]. We trained the network with 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. We used momentum of 0.9 as well.

Variance: When evaluating variance we fixed the number of DAUs per filter to four and varied the variance σ^2 from 0.3^2 to 0.8^2. Results are reported in Tab. 1. They indicate that the variances have negligible effect on classification performance with changes between different variances

Table 1: Variance σ^2 hyperparameter evaluation on CIFAR10 classification task using a shallow DAU-ConvNet. Variance has minor effect on classification performance.

Variance σ^2	0.3^2	0.4^2	0.5^2	0.6^2	0.7^2	0.8^2
DAU-ConvNet CIFAR10	82.9	83.4	**83.8**	83.6	82.9	82.8

at only around 1%. We used variance of $\sigma^2 = 0.5^2$ for all remaining experiments in this paper.

Number of DAUs per filter: When evaluating the number of units we used a variance $\sigma^2 = 0.5^2$ and varied the number of units on the second and the third layer using 1, 2, 4 or 6 units. We fixed DAUs on the first layer to four units to capture initial edges and corners. Results are reported in Tab. 2. They indicate only a slight increase of performance when additional units are added. Difference between using a single unit or using six units is only 1%. We used two and four units in remaining experiments as a trade-off between performance and parameter count. Additional extensive evaluation of parameter count was perform in Sec. 5.2.

4.2. Classification performance

Performance of DAUs on the classification task was tested on the ILSVRC 2012 dataset [22]. A standard testing protocol was used. The network was trained on 1.2 million images and tested on the validation set with 50,000 images. All images were cropped and resized to 227 pixels. To keep the experiments as clean as possible, we did not apply any advanced augmentation techniques apart from mirroring during the training with probability 0.5.

As our baseline ConvNet architecture, we chose the AlexNet model [16], which is composed of 7 layers: 5 convolutional and 2 fully connected. We retained the local normalization layers, max-pooling and dropout on fully-connected layers of the original AlexNet [16], but we did not split channels into two streams as was done in the original work [16]. We also used weight initialization technique by Glorot and Bengio [7].

The baseline ConvNet was modified into a DAU-ConvNet as follows. The filters in the convolutional layers from layer 2 to 5 were replaced by our DAU filters from Sec. 3. Four DAUs per filter were used in the second layer and two DAUs per per filter in the remaining three layers. This follows approximate coverage of filter sizes from classic ConvNet with 5×5 filter sizes for the second layer and 3×3 filter sizes for the remaining layers. First layer and fully connected layers remained unchanged using classic convolutional layer. This is partially due to technical limitation of our current implementation. Our recent work on alleviating this issues indicates that even fully connected layers with 36 units (6×6 filter sizes) can be replaced with only 6 DAUs (with comperable perfomance).

Table 2: Number of units per filter hyperparameter evaluation on CIFAR10 classification task using a shallow DAU-ConvNet. Larger number of units increase classification performance only slightly.

Number of units per filter	1	2	4	6
DAU-ConvNet CIFAR10	82.9	83.3	83.8	**84.1**

4.2.1 Optimization

We trained both, ConvNet as well as DAU-ConvNet, with stochastic gradient descent using batch size of 128. Both models were trained for 800,000 iterations, or 80 epochs, with initial learning rate of 0.01, which is reduced by a factor of 10 every 200,000th iteration. We used momentum with a factor of 0.9 and a weight decay factor of 0.0005. In our layers with DAUs a decay factor could be applied to weights and offsets as well, although applying to offsets has a different effect than decay on regular weights as it would prevent them from moving further from the center. We used decay only on weights but not on the offsets.

4.2.2 Classification results

The results are reported in Tab. 3 with performance monitoring during the training reported in Fig. 2. After 600,000 iterations, the DAU-ConvNet and the baseline ConvNet converge to a comparable performance. Namely, after 80 epochs, both models achieved accuracy of slightly below 57% (see Tab. 3), however, DAU-ConvNet was converging much faster, resulting in higher performance jumps before the learning rate reduction steps. Tab. 3 shows the number of free parameters in the convolutional layers. Note that DAU-ConvNet requires 30% less parameters than the baseline classic ConvNet and our analysis in Sec. 5.2 shows this can be improved even further.

Even though the final classification performance converges to the same result, our model exhibits good performance even on higher learning rates. This indicates that the DAUs modify landscape of the loss function so that it can be traversed faster with higher learning rates in DAU-ConvNets. This improvement may also be contributed to reduction of the number of parameters in the DAUs and supports the hypothesis that DAUs do not lose expressive power on the account of their simple functional form.

4.3. Semantic segmentation

We analyze the performance of DAUs on a dense prediction problem where large receptive fields and fine resolution are particularly important. In this experiment, we start from

the baseline ConvNet and DAU-ConvNet trained in Sec. 4.2 and fine-tune them for a segmentation task. A standard technique is used to modify the classification networks into segmentation nets. Specifically, the last fully-connected classification layer is replaced by the expansion and classification layer from Long et al. [18] that entails a 1×1 classification layer and up-sampling using a deconvolution layer to obtain pixel-wise loss. To keep the experiments clean we have not added advanced network adaptations that have emerged over recent years, like feature combination across layers, etc., although our approach is general enough to allow such upgrades. The object boundaries are maintained sharp by further increasing higher layer resolution.

4.3.1 Increasing resolution at higher layers

Our classifier from Sec. 4.2 follows the AlexNet architecture and reduces the resolution by 32-fold. For the purpose of segmentation we increase the resolution at higher layers and remove the last two max-pooling layers thus reducing resolution only by 8-fold for segmentation. With this modification, the network retains finer details.

Increasing the resolution on a pre-trained model causes a misalignment of already learned filter weights and their positions w.r.t. the expected resolution. We compensate for that by modifying the parameters of the affected layers. In particular, for the layers with DAU filters we increase displacement of a unit with the appropriate factor, while in classic ConvNet layer we use dilated convolution with the same factor. The layers after the first-removed-max-pooling use a factor of two (layers 3–5) and the layers after the second-removed-max-pooling use a factor of four (layer 6).

4.3.2 Dataset

We evaluate our segmentation DAU-ConvNet and the baseline ConvNet with dilation on PASCAL VOC 2011 segmentation dataset. For the training we use 1,112 training images from PASCAL VOC 2011 segmentation combined with 7,386 images collected by Hariharan et al. [8]. We report results on PASCAL VOC 2011 validation set excluding the images from [8] that were also used for training.

4.3.3 Optimization

We trained the models with mini-batch stochastic gradient descent and a batch size of 20 images for 65,000 iterations, or 150 epoch. We used a fixed learning rate of 0.0002, weight decay of 0.0005 and momentum of 0.9. The added classification layer was initialized with zeros, similar to [18] and we used a normalized per-pixel softmax loss function applied only to pixels with a valid annotation.

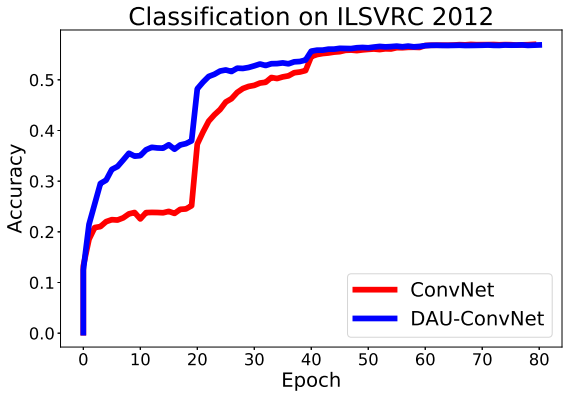


Figure 2: Classification top-1 accuracy on ILSVRC 2012 validation set using AlexNet architecture. Our DAU-ConvNet converges faster with larger learning rates than standard ConvNet.

Table 3: Results on ILSVRC 2012 validation set using AlexNet architecture and corresponding number of parameters on convolutional layers. We report top-1 accuracy.

	Top-1 accuracy (%)	Number of parameters on conv. layers
DAU-ConvNet	56.89	2.3 mio
ConvNet [16]	56.99	3.7 mio

4.3.4 Segmentation results

The performance of DAU-ConvNet compared to the baseline ConvNet with dilation is shown in Fig. 3. The DAU-ConvNet shows faster convergence in testing loss. In addition, DAU-ConvNet shows consistently better segmentation performance than the baseline ConvNet-dilation across all measures. The mean IU and per-pixel accuracy are improved by approximately 2%. Looking at the per-class mean IU in Tab. 4, we observe improved performance across all categories, with the exception of "dog", "sheep" and "train".

5. Analysis of displaced aggregation units

In this section we conducted two experiments to gain further insights into DAUs. The first experiment analyzed the spatial distribution of the DAUs in the learned filters (Sec. 5.1). The second experiment explored the relation between the number of DAUs per filter and the network performance (Sec. 5.2).

5.1. Spatial adaptation of filter units

We investigate spatial distribution of DAUs in our network by observing distributions of the learned displacements in the segmentation DAU-ConvNet in Sec. 4.3. The aim of the experiment was to expose two aspects: (i) the dis-

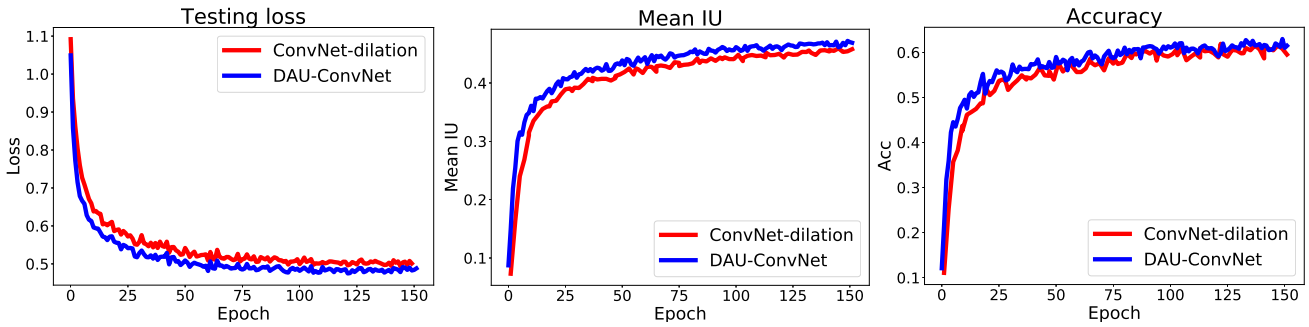


Figure 3: Performance monitoring during fine-tuning on segmentation task. Results are reported on PASCL VOC 2011 segmentation validation set. We report testing loss value and averaged mean-iu and accuracy.

Table 4: Results on segmentation task using a PASCAL VOC 2011 validation set. We report per-class mean-IU and averaged mean-IU over all classes.

	background	aeroplane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow	diningtable	dog	horse	motorbike	person	potted plant	sheep	sofa	train	tv/monitor	mean IU
DAU-ConvNet	**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**
ConvNet-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

tribution of the learned displacements, which indicates displacement locations favored for a given task, and (ii) overall spatial distribution, which indicates the preferred receptive field size.

Such an experiment is very difficult to perform with classical ConvNets and requires a combinatorial sweep over alternative architectures with various manually-defined filter designs. For example, dilated convolutions can alter unit positions, but this must be done with a specific pre-defined dilation factor. In contrast, with displaced aggregated units in our filters we can analyze their displacements that adjust during the learning on the segmentation problem with a sub-pixel accuracy and not being confined to the same pattern across all filters. Such an analysis is not possible with the existing ConvNet architectures.

We investigate two types of distributions: (i) a 1D distance-to-center distribution and (ii) a distribution of displacements in 2D space. We obtain 1D spatial distribution by collecting displacement values of units from all features at a specific layer and compute their distances to the center of the filter. All distances are collected in a histogram with each unit contributing with its corresponding input amplification factor. We obtain the second 2D spatial distribution by plotting all displacements from a specific layer into the same graph.

5.1.1 Results and discussion

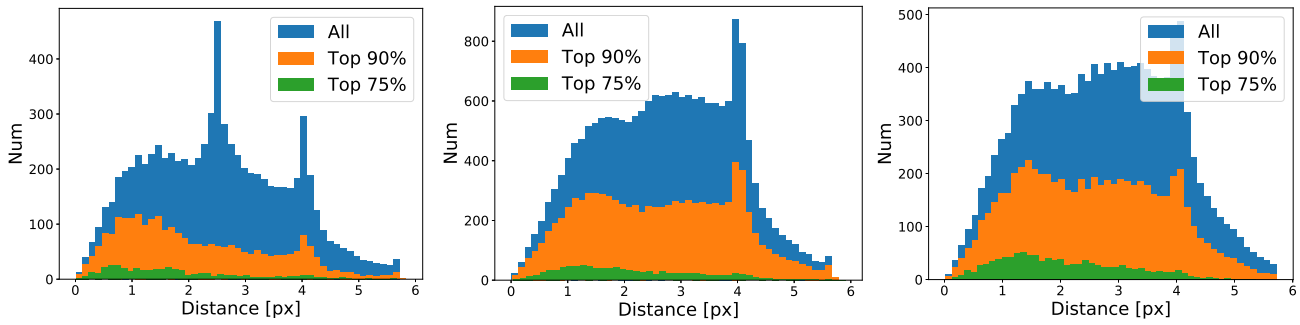
We compute several per-layer distributions from the DAU-ConvNet model trained for semantic segmentation in Sec. 4.3. All 1D distributions are shown in Fig. 4 and all 2D distributions are shown in Fig. 5. The first set of distributions is computed from all absolute amplification values $|w_k|$. The second distribution is obtained by retaining 90% of the largest absolute amplification values and the third distribution by retaining 75% of the largest absolute amplification values.

We observe in Fig. 4 two significant spikes, one at 2.5 pixels and another at 4 pixels away from the center. The spike at 2.5 pixels, that occurs only at the third layer, is artificial due to the fixed initialization points. It indicates that many units did not move from their initialization point during learning. This can be observed in Fig. 5 with high density at initialization center points (red dots). Further inspection shows that those units do not contribute to the filter significantly. In fact, they disappear in the plots when units with lowest amplification value are removed.

The 4th and the 5th layers have similar initialization points but no apparent spikes in their distance-to-center distributions, as shown in (Fig. 4b and Fig. 4c). This indicates a low learning rate for the 3th layer where displacements may not have been able to move quickly enough. As results in Sec. 5.2 suggest, some of these may be removed without performance reduction.

The second spike at 4 pixels away from the center (Fig. 4) is more significant since it does not disappear when removing units with small amplification factors. This spike occurs due to an artificial limit on boundary of the receptive field which in our case is set at four pixels in both spatial dimensions[1]. Still, a significant number of those units have

[1]Our current Caffe/CUDA implementation allows distances only up to



(a) Layer 3 (b) Layer 4 (c) Layer 5

Figure 4: Distance-to-center distributions collected from displacement of DAUs. Distributions reported per-layer (columns) and after elimination of units with smallest amplification factor using different relative thresholds (colors).

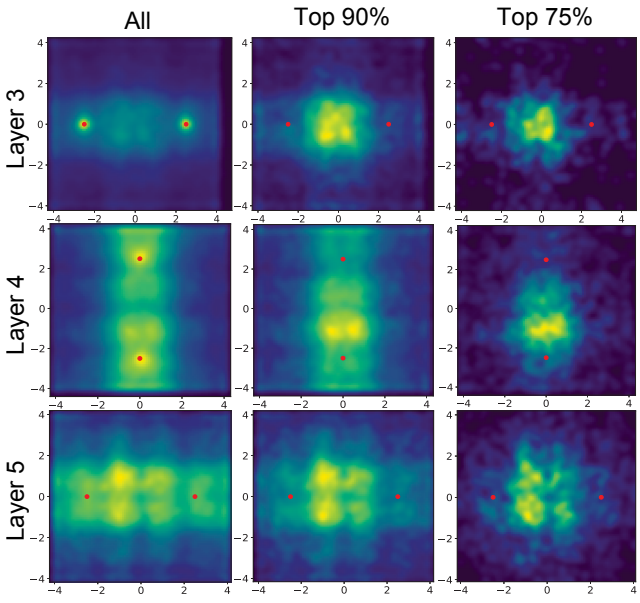


Figure 5: 2D distributions of displacements collected from DAUs. Red dots indicate initialization points. Distributions reported for layer 3, 4 and 5 in top, middle and bottom row, respectively, and each after retaining different number of important units (in columns).

large amplification factors. This points to the need of further increasing the allowed sizes of the receptive fields.

A consistent shape of distance-to-center distributions throughout the layers (Fig. 4) points to a desired spatial distribution of units for segmentation. It indicates that units must densely cover locations at a distance of 1-2 pixels away from the center Some units with high amplification factor are located far away from the center which indicates a need to covering larger receptive fields albeit with lower density. The same conclusion is drawn from 2D spatial distributions in Fig. 5.

5.2. Parameter-space analysis

We analyzed the impact of the number of DAUs per filter on the network performance to gain additional insights. Several research papers investigate the influence of parameter space in classic ConvNets with respect to the number of layers, number of features or filter sizes [6], but could not report analysis with respect to the filter units. The classic ConvNets are limited by a minimal filter size of 3×3 that already has a minimal spatial coverage. Reducing the parameter count by reducing the filter size would not be feasible. Our redefinition of filter units on the other hand allows us to investigate filters with even smaller number of parameters without affecting spatial coverage and the receptive field sizes.

The number of units per filter was set through a hyperparameter. Thus the number of parameters was kept equal across all filters during training. Then the units with small amplification weights were removed

We perform the experiments on a classification problem with ILSVRC 2012 and AlexNet architecture as presented in Sec. 4.2. We used the same optimization settings for all variants.

Three variations of our network are compared (see Tab. 6): Large, Medium and Small. The Medium DAU-ConvNet is the network from Sec. 4.2. The Small DAU-ConvNet uses as few as two or a single DAU per filter, while the Large DAU-ConvNet uses six to four DAUs. This affects the number of learned parameters as follows. The Small DAU-ConvNet contains 400,000 DAUs, the Medium DAU-ConvNet contains 800,000 DAUs, and the Large DAU-ConvNet contains 1.5 mio DAUs. These values translate to 4.5 mio, 2.3 mio, and 1.2 mio parameters on convolutional layers for Small, Medium and Large DAU-ConvNet, respectively. For the reference, the baseline ConvNet from Sec. 4.2 contained 3.7 mio units on conv. layers.

4 or 8 pixels. This can be overcome with a improved implementation.

Table 5: Analysis of the number of parameters and units per filter with three variants of DAU-ConvNet: 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 percentage of removed units.

Relative	Large DAU-ConvNet			Medium DAU-ConvNet			Small DAU-ConvNet		
threshold	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 6: Per-filter unit and parameter count with three variants of DAU-ConvNet: Large, Medium and Small. Note, a unit in DAU has three parameters and ConvNet has one.

	Per-filter unit count			
	Large	Medium	Small	ConvNet
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

5.2.1 Results and discussion

The results are reported in Tab. 5. We observe that all three networks achieve classification accuracy of approximately 56-57% on ILSVRC 2012. These results indicate that DAU-ConvNets may require only one to two units per filter resulting in 3 to 6 parameters per filter on convolutional layers. This is significantly lower than classic networks that already contain 9 parameters for the smallest filter (i.e., 3×3) and 25 for a moderately large (i.e., 5×5). The low parametrization is possible in DAU-ConvNets since the network learns on its own the receptive field perimeter without the need to increase the parameter space to cover large displacements.

Furthermore, looking at the performance when eliminating units with small amplification factor reveals further improvements. In all three networks we were able to eliminate 7-13% of units without affecting their classification performance at all.

6. Discussion and conclusion

We proposed a displaced aggregation filter units (DAUs) to replace a fixed, grid-based unit in existing convolutional networks. The DAUs modify only the convolutional layer in standard ConvNets, but afford several advancements. The receptive field is now learned. The learning is efficient since DAUs decouple the number of parameters from the receptive field size and efficiently allocate the free parameters. We demonstrated this on the classification and segmentation tasks, and showed faster convergence on the classification task and improved performance on the segmentation task.

The DAUs remove the filter size hyperparameter, but introduce a hyperparameter on the DAU's aggregation perimeter size and the number of DAUs per filter. We experimentally showed that both have minor affect on the classification performance. We can set aggregation perimeter size to a fixed value, while a larger number of units per filter marginally increases performance. With less than 1% drop in performance we can use only one unit per filter. This is a highly interesting result as it suggests that efficient ConvNets can be implemented by replacing general convolution layers by Gaussian filters and a single sub-pixel sampling per filter.

The analysis of learned DAU displacements showed that units are concentrated at the filter center, while some are positioned further away. This shows the capacity to learn small as well as large receptive fields within a unified framework. Our distributions directly point to locations that need to be densely sampled in filters. This can be used to adjust the dilation factors from atrous convolutions in classical ConvNets [4] more efficiently.

Lastly, our comprehensive study of per-filter parameter allocation showed an inefficient allocation of parameters in existing ConvNets. DAU-ConvNets achieved comparable performance to classic CovnNets at 3-times less parameters per filter. Analysis shows there is also room for further improvements as elimination of units with lowest amplification factors (even without post-hoc fine-tunning) can save 10% of parameters without sacrificing the performance. Furthermore, our recent preliminary work on applying DAUs to fully connected layers indicates possible savings in parameters for fully connected layers as well.

Acknowledgments. This work was supported in part by the following research projects and programs: project GOSTOP C3330-16-529000 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.

References

[1] I. Amidror. *Mastering the Discrete Fourier Transform in One, Two or Several Dimensions*. 2013. 1

[2] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the Devil in the Details: Delving Deep into Convolutional Nets. In *arXiv preprint arXiv: ...*, pages 1–11, 2014. 1

[3] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *Pattern Analysis and Machine Intelligence*, pages 1–14, 6 2016. 1

[4] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam. Rethinking Atrous Convolution for Semantic Image Segmentation. 2017. 1, 2, 8

[5] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei. Deformable Convolutional Networks. In *International Conference on Computer Vision*, 2017. 2

[6] D. Eigen, J. Rolfe, R. Fergus, and Y. Lecun. Understanding Deep Architectures using a Recursive Convolutional Network. pages 1–9, 2014. 7

[7] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. *Aistats*, 9:249–256, 2010. 3, 4

[8] B. Hariharan, P. Arbel, L. Bourdev, S. Maji, J. Malik, U. C. Berkeley, A. Systems, P. Ave, and S. Jose. Semantic Contours from Inverse Detectors. In *International Conference on Computer Vision*, 2011. 5

[9] K. He, X. Zhang, S. Ren, and J. Sun. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. In *European Conference on Computer Vision*, pages 346–361, 2014. 3

[10] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *Computer Vision and Pattern Recognition*, pages 171–180, 2016. 1

[11] J.-H. Jacobsen, J. van Gemert, Z. Lou, and A. W. M. Smeulders. Structured Receptive Fields in CNNs. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2610–2619, 2016. 2

[12] Y. Jeon and J. Kim. Active Convolution: Learning the Shape of Convolution for Image Classification. 2017. 2

[13] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional Architecture for Fast Feature Embedding. 2014. 2

[14] H. Kaiming, G. Gkioxara, P. Dollar, and R. Girshick. Mask R-CNN. In *International Conference on Computer Vision*, 2017. 1

[15] A. Krizhevsky. Learning Multiple Layers of Features from Tiny Images. *Science Department, University of Toronto, TechReport*, pages 1–60, 2009. 3

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances In Neural Information Processing Systems 25*, pages 1097–1105, 2012. 4, 5

[17] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2323, 1998. 2

[18] J. Long, E. Shelhamer, and T. Darrell. Fully Convolutional Networks for Semantic Segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 8828, pages 3431–3440, 2015. 1, 5

[19] S. Luan, B. Zhang, C. Chen, X. Cao, Q. Ye, J. Han, and J. Liu. Gabor Convolutional Networks. *British Machine Vision Conference*, pages 1–12, 2017. 2

[20] W. Luo. Understanding the Effective Receptive Field in Deep Convolutional Neural Networks. *Nips*, (Nips), 2016. 2

[21] J. Redmon and A. Farhadi. YOLO9000: Better, Faster, Stronger. 2017. 1

[22] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. 4

[23] K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recoginition. In *International Conference on Learning Representations*, pages 1–14, 2015. 1

[24] D. Tabernik, M. Kristan, J. L. Wyatt, and A. Leonardis. Towards Deep Compositional Networks. In *International Conference on Pattern Recognition*, 2016. 2, 3

[25] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated Residual Transformations for Deep Neural Networks. In *Conference on Computer Vision and Pattern Recognition*, 2017. 1

[26] F. Yu and V. Koltun. Multi-Scale Context Aggregation by Dilated Convolutions. In *International Conference on Learning Representations*, 2016. 1

[27] F. Yu, V. Koltun, and T. Funkhouser. Dilated Residual Networks. In *Computer Vision and Pattern Recognition*, 2017. 1