Exploring the Importance of Pretrained Feature Extractors for Unsupervised Anomaly Detection and Localization Supplementary Material

Lars Heckler[1,2] Rebecca König[1] Paul Bergmann[1]

[1]MVTec Software GmbH, [2]Technical University of Munich

{lars.heckler, rebecca.koenig, paul.bergmann}@mvtec.com

1. Receptive Field Estimation

In order to estimate the receptive field of a feature layer, we first compute the gradient norm of the pixels of an empty input image of size 1024×1024 pixels with respect to the center pixel of the respective feature map. We then clip gradients with a norm of less than a threshold of 10^{-2} to 0. This is necessary for networks that contain adaptive average pooling layers in the early parts of the network, such as EfficientNet-B5, which propagates very small gradients to every input pixel. In order to exclude padding artifacts we additionally ignore gradients within a small area around the image border. We then compute the bounding box of pixels that contain non-zero gradients and select the longer side length as the estimated receptive field. Figure 1 visualizes the estimated receptive field for the four evaluated layers of Wide ResNet-50. Our implementation is based on a publicly available code base[1].

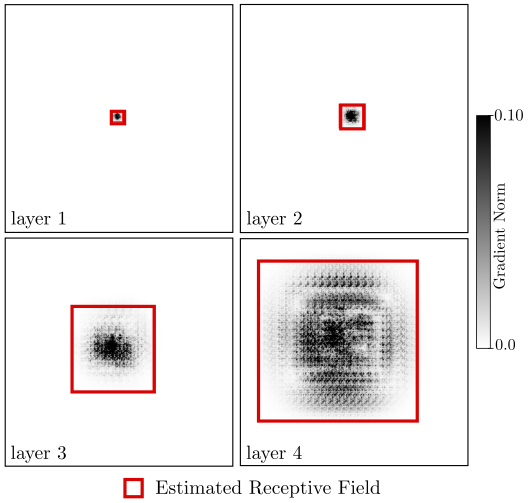

Figure 1. Estimated receptive field for the four evaluated layers of Wide ResNet-50. The absolute gradient activations are visualized in shades of black.

2. Variations over Distinct Object Categories

Figure 2 shows the number of objects for which a particular layer yields the best performance for Asymmetric Student Teacher (AsymST) and FastFlow. For both anomaly classification and localization, the best-performing layer depends on the inspected object.

3. Influence of Image Size

Figure 3 shows the influence of the image size on the final AD performance when using Wide ResNet-50 and DenseNet-201 as feature extractors for FastFlow and PatchCore, respectively.

4. Influence of Different Pretraining Strategies

For investigating the effect of different network initializations resulting from distinct pretraining strategies, we use publicly available model checkpoints for ResNet-50. Table 1 specifies these initializations and the checkpoint source. Experimental results for AsymST and PatchCore are shown in Figure 4 and Figure 5, respectively.

5. Network Architectures and Layers

Table 2 provides an overview over the examined network architectures and layers. For extracting features from pretrained networks, we utilized the PyTorch feature extraction module[2].

[1] https://github.com/shelfwise/receptivefield

[2] pytorch.org/vision/stable/feature_extraction.html

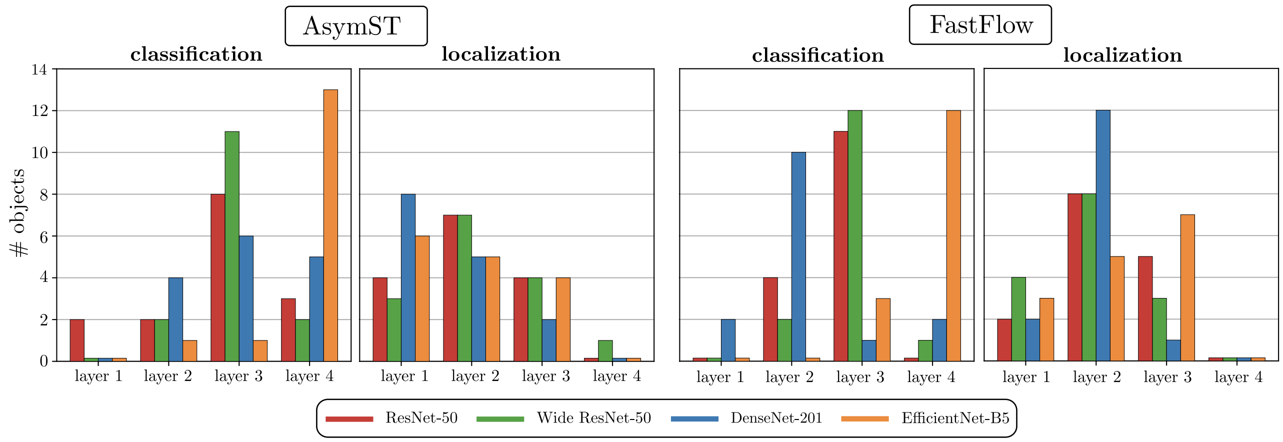

Figure 2. Number of object categories from MVTec AD for which an intermediate layer yields the best performance for AsymST and FastFlow. For each feature extractor, the layers are ordered by their relative receptive field (RRF) from small to large.

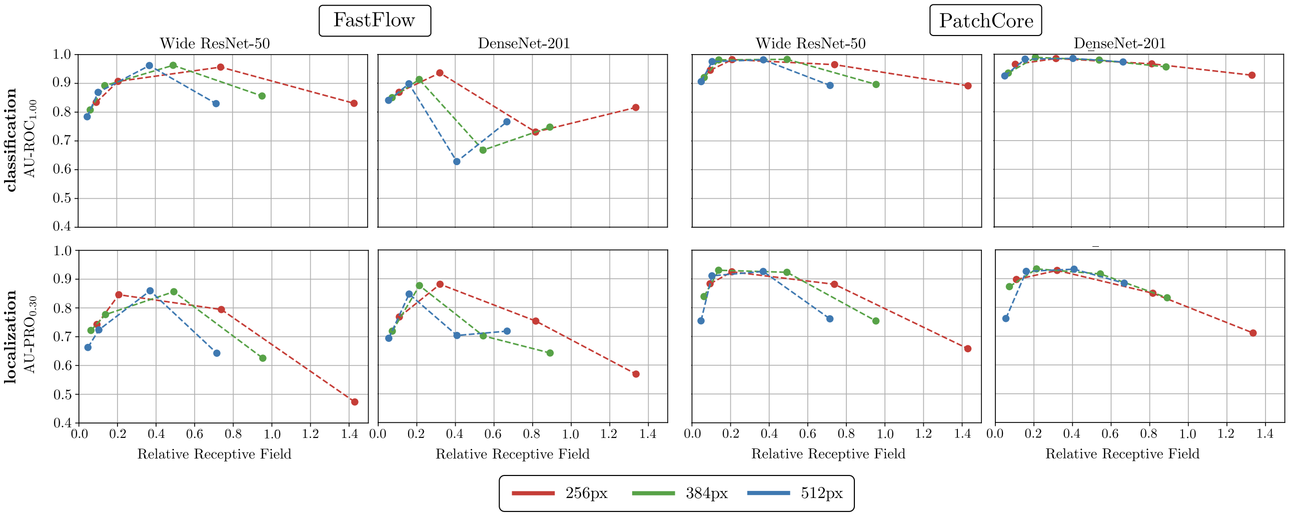

Figure 3. Varying the input image size for FastFlow and PatchCore when using Wide ResNet-50 and DenseNet-201 as feature extractors. Increasing the input dimension reduces the RRF and also affects the performance of the individual feature layers.

Table 1. Pretraining Strategies and Network Initialization

pretraining/ initialization	**checkpoint source**	**info**
MoCo	github.com/facebookresearch/moco	MoCo v2 epoch800
SwAV	github.com/facebookresearch/swav	epoch 800; batchsize 4096
SeLa	github.com/facebookresearch/swav	SeLa-v2; epoch 400; multi-crop 2x160 + 4x96
SimCLR	github.com/google-research/simclr	SimCLRv1; ResNet50 (1x); converted to PyTorch github.com/tonylins/simclr-converter
supervised	PyTorch	weights=ResNet50_Weights.IMAGENET1K_V2
random	PyTorch	weights='DEFAULT'

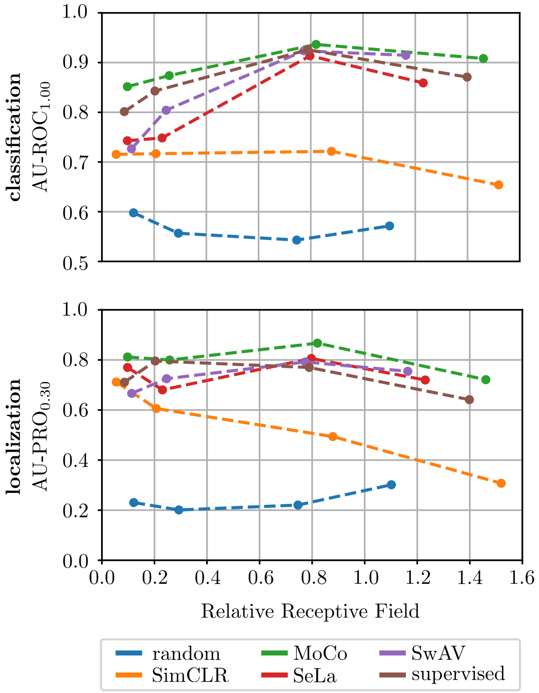

Figure 4. AsymST AD performance of AsymST with ResNet-50 as feature extractor using different pretraining strategies. Weight initializations obtained from self-supervised paradigms are a competitive alternative to supervised ImageNet pretraining.

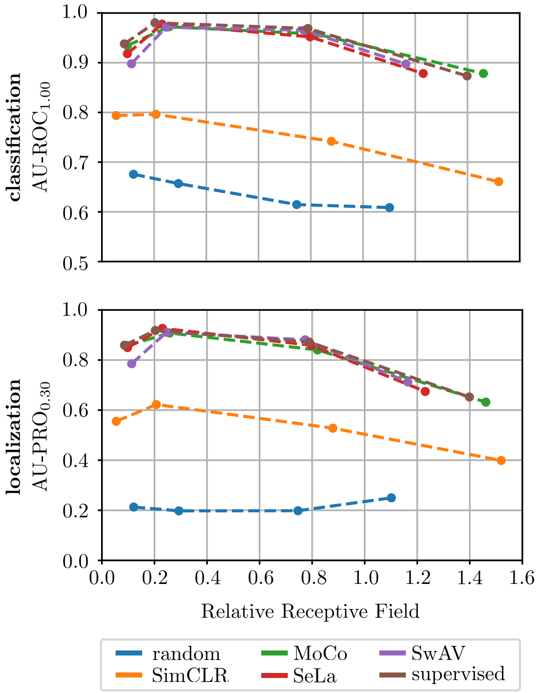

Figure 5. AD performance of PatchCore with ResNet-50 as feature extractor using different pretraining strategies. Weight initializations obtained from self-supervised paradigms are a competitive alternative to supervised ImageNet pretraining.

Table 2. Network Architectures and Layer Specifications

architecture	**layer specification**	**description**
ResNet-50	layer1.2.relu_2	last layers of the four main stages
	layer2.3.relu_2	
	layer3.5.relu_2	
	layer4.2.relu_2	
Wide ResNet-50	layer1.2.relu_2	last layers of the four main stages
	layer2.3.relu_2	
	layer3.5.relu_2	
	layer4.2.relu_2	
DenseNet-201	features.denseblock1.cat	last layers of the four main stages
	features.denseblock2.cat	
	features.denseblock3.cat	
	features.denseblock4.cat	
EfficientNet-B5	features.2.4.add	last layers of block 2,3,5,6
	features.3.4.add	
	features.5.6.add	
	features.6.8.add	