MorphPool: Efficient Non-linear Unpooling in Convolutional Neural Networks Supplementary Material

Rick Groenendijk
r.w.groenendijk@uva.nl
Leo Dorst
l.dorst@uva.nl
Theo Gevers
th.gevers@uva.nl

Computer Vision Group
University of Amsterdam
Amsterdam, the Netherlands

Supplementary Material A: Implementation Details

Network Architecture

Using the DownUpNet architecture, the quality of semantic predictions as a result of different sampling schemes is tested. A schematic overview of the network architecture is shown in Figure 2. There are three choices for down-sampling operations:

- Convolution with stride 2 and kernel size 3. This operations introduces $C^2 \times K^2$ additional parameters per down-sampling layer, where C is the number of channels of the layer and K the kernel size.
- Max Pool with a non-overlapping kernel of size 2 at stride 2.
- Generalised Morphological Pooling, with variable kernel size. The pooling operation can be flat (similar to max pooling) or be parameterized with any structuring element. In the case of parabolic structuring elements, this layer introduces C parameters. And in the freely parameterized case, this introduced $C \times K^2$ parameters.

There are three choices for up-sampling operations:

- Sparse up-sampling at rate 2, followed by bilinear interpolation and convolution. This operation introduces $C^2 \times K^2$ additional parameters per up-sampling layer.
- Unpooling followed by deconvolution as introduced in [3] and used in [1]. This operation introduces $C^2 \times K^2$ additional parameters per up-sampling layer.
- Morphological unpooling as described in Section 3.2, which unpools by provenance followed by a morphological dilation. In case of a flat structuring element, this is a parameterless layer. In case of parabolic parameterisation, this introduces C parameters. And in the case of a freely parameterized dilation, this layer introduces $C \times K^2$ additional parameters.

© 2022. The copyright of this document resides with its authors.
It may be distributed unchanged freely in print or electronic forms.

Note especially that morphological pooling and unpooling (*i.e.* MorphPool) introduce hardly any additional parameters. Even the general (*i.e.* freely) parameterised dilations introduce a factor C (*i.e.* the number of channels) fewer parameters than full convolutions in the other sampling operations do.

In the network, all convolutions have a kernel size of 3, and are padded to retain resolution. When training on 2D-3D-S, there are two convolution blocks (including batch normalisation and ReLU non-linearity) before and after each sampling operation. This is because the 2D-3D-S is much larger than NYUv2 and SUN-RGBD and initial experimentation showed significantly improved performance of this architecture using a larger network for all methods.

Training details

All methods are implemented in PyTorch, with the exception of the morphological operations which are implemented as C++/CUDA extensions. The extensions are compiled using g++ and CUDA11. All networks are trained using SGD with Nesterov Momentum [2], with a learning rate that decays exponentially by a scalar γ. Based on initial experimentation, for consistent performance this γ can be set such that the initial learning rate λ decays to at 2% of the initial λ, according to $\gamma = \sqrt[\text{epochs}]{\frac{2}{100}}$ with *epochs* the number of training epochs, and $\textit{epochs} \geq 1$.

RGB images are normalised with training set statistics to follow a zero-mean Gaussian, as is common for neural networks. Depth images however are not normalised, and are unscaled depth values in meters. Depth data in general has more sensor noise and infilling artefacts. In combination with the unscaled data, this led to less stable learning at similar λ to RGB images. Therefore, networks trained with depth input start with a learning rat $\lambda = 5\text{e}{-4}$, whereas RGB input starts with $\lambda = 5\text{e}{-3}$.

During training, random crops of size 384×384 are used for NYUv2 and SUN-RGBD, crops of 512×512 are used for 2D-3D-S. During testing, centre crops are used as close to full resolution as possible, while retaining a resolution that is divisible by 2^5. Networks are trained for 100 epochs on NYUv2 and SUN-RGBD with a batch size of 8; on 2D-3D-S networks are trained for 40 epochs with a batch size of 16. The experiments can be run on a NVIDIA GTX1080Ti, except the experiments on 2D-3D-S which is trained on a NVIDIA A6000.

Supplementary Material B: Supporting Results

In this section, all supporting results for the main experiments section are given. All results are reported over a variety of tables. For semantic segmentation on Depth, see Table 1 for 2D-3D-S, Table 2 for SUN-RGBD, and Table 3 for NYUv2. For semantic segmentation on RGB, see Table 4 for 2D-3D-S, Table 5 for SUN-RGBD, and Table 6 for NYUv2. And finally, for depth auto-encoding, see Table 7.

Additionally, depth-wise convolutions can be used to equalise the number of parameters available to networks making use of morphological and linear sampling. On top of the results on depth data for SUN-RGBD in the main text, results are listed for depth input on NYU in Table 8, and for RGB input in Table 9 and Table 10. In general, for networks with

Table 1: **2D-3D-S Segmentation on Depth Results.** The first column denotes the sampling method, the second through fourth column denotes the kernel size with which the Pooling, Unpooling, and Convolution scheme worked. For each method of post-processing (None, Convolution, Deconvolution) the up-sampled feature volume, the number of parameters and performance is given. The high-lighted gray cells denote the standard procedures for up and down-sampling: Unpooling followed by (de)convolution, bilinear interpolation followed by convolution. In addition, General MorphPool denotes general parameterised morphological structuring elements. The bold-faced results indicate best performance per column.

				None				Conv				Deconv			
	P	U	C	#params	mIoU	acc	bf	#params	mIoU	acc	bf	#params	mIoU	acc	bf
Linear	3	-	3	12.6M	0.403	0.693	0.426	25.1M	0.413	0.699	0.434	25.1M	0.410	0.692	0.430
Standard Pool	2	3	3	0	0.352	0.657	0.375	12.6M	0.377	0.675	0.393	12.6M	0.397	0.693	0.407
MorphPool	2	3	3	0	0.385	0.682	0.441	12.6M	0.399	0.688	0.449	12.6M	0.410	0.694	0.454
MorphPool	3	5	3	0	0.425	0.707	**0.476**	12.6M	0.428	0.714	0.479	12.6M	0.442	0.720	0.487
Para. MorphPool	3	5	3	4.0K	**0.429**	**0.711**	0.472	12.6M	0.445	0.719	0.487	12.6M	0.433	0.710	0.484
General MorphPool	3	5	3	67.5K	0.425	0.710	0.473	12.6M	**0.445**	**0.722**	**0.488**	12.6M	**0.450**	**0.724**	**0.492**

Table 2: **SUN-RGBD Segmentation on Depth Results.** Results indicate morphological pooling and unpooling have improved performance over standard pooling and unpooling by a large margin. In addition, because of the non-linear nature of depth, morphological operations for down and up-sampling also beat the linear baseline with only half of the parameters.

				None				Conv				Deconv			
	P	U	C	#params	mIoU	acc	bf	#params	mIoU	acc	bf	#params	mIoU	acc	bf
Linear	3	-	3	12.6M	0.349	0.711	0.308	25.1M	0.398	0.741	0.339	25.1M	0.399	0.742	0.334
Standard Pool	2	3	3	0	0.208	0.643	0.255	12.6M	0.297	0.687	0.294	12.6M	0.288	0.690	0.294
MorphPool	2	3	3	0	0.345	0.712	0.328	12.6M	0.375	0.735	0.345	12.6M	0.375	0.732	0.347
MorphPool	3	5	3	0	0.382	0.735	0.346	12.6M	0.412	0.748	0.364	12.6M	0.407	0.747	0.361
Para. MorphPool	3	5	3	4.0K	0.396	0.741	0.351	12.6M	0.412	**0.751**	0.362	12.6M	0.414	**0.750**	**0.366**
General MorphPool	3	5	3	67.5K	**0.398**	**0.745**	**0.352**	12.6M	**0.416**	0.748	**0.366**	12.6M	**0.416**	0.748	0.365

Table 3: **NYUv2 Segmentation on Depth Results.** Results on 2D-3D-S and SUN-RGBD are confirmed by the results on NYUv2. This is unsurprising: NYUv2 is a subset of the larger SUN-RGBD dataset. The quality of inpainting the NYUv2 depth maps is much better than other images in the SUN-RGBD dataset, since other subsets had much sparser depth data.

				None				Conv				Deconv			
	P	U	C	#params	mIoU	acc	bf	#params	mIoU	acc	bf	#params	mIoU	acc	bf
Linear	3	-	3	12.6M	0.305	0.597	0.175	25.1M	0.320	0.609	0.176	25.1M	0.312	0.603	0.176
Standard Pool	2	3	3	0	0.121	0.453	0.177	12.6M	0.159	0.495	0.173	12.6M	0.151	0.487	0.176
MorphPool	2	3	3	0	0.290	0.592	0.201	12.6M	0.316	0.608	0.211	12.6M	0.318	0.613	0.207
MorphPool	3	5	3	0	0.323	0.607	0.200	12.6M	0.357	0.627	0.208	12.6M	0.360	0.630	0.212
Para. MorphPool	3	5	3	4.0K	0.348	0.627	0.205	12.6M	**0.360**	0.630	**0.212**	12.6M	0.367	0.630	0.215
General MorphPool	3	5	3	67.5K	**0.353**	**0.632**	**0.207**	12.6M	0.357	**0.630**	0.207	12.6M	**0.377**	**0.643**	**0.219**

Table 4: **2D-3D-S Segmentation on RGB Results.** Similar to the segmentation results on depth, regular pooling and unpooling, even followed by (de)convolution, yields inferior performance to this paper's generalised morphological pooling. Linear interpolation, however, performs on par worse than morphological operations. This could be due to the RGB images, which cannot necessarily be expected to suit a set of morphological operations. Again, morphological operations introduce no or little additional parameters, whereas linear sampling plus convolution does. Finally, morphological pooling & unpooling performs best at semantic boundaries as measured by the Boundary F1-score.

				None				Conv				Deconv			
	P	U	C	#params	mIoU	acc	bf	#params	mIoU	acc	bf	#params	mIoU	acc	bf
Linear	3	-	3	12.6M	0.355	0.637	0.289	25.1M	**0.370**	0.649	0.301	25.1M	**0.372**	**0.650**	0.302
Standard Pool	2	3	3	0	0.325	0.607	0.257	12.6M	0.338	0.623	0.266	12.6M	0.345	0.625	0.271
MorphPool	2	3	3	0	0.344	0.632	0.299	12.6M	0.351	0.631	0.296	12.6M	0.362	0.638	0.305
MorphPool	3	5	3	0	**0.365**	**0.648**	0.316	12.6M	0.361	0.648	0.314	12.6M	0.356	0.642	**0.313**
Para. MorphPool	3	5	3	4.0K	0.362	0.642	**0.317**	12.6M	0.354	0.642	0.308	12.6M	0.365	0.649	**0.313**
General MorphPool	3	5	3	67.5K	0.351	0.638	0.310	12.6M	0.364	**0.650**	**0.315**	12.6M	0.364	0.646	0.312

Table 5: **SUN-RGBD Segmentation on RGB Results.** SUN-RGBD shows similar results to 2D-3D-S on RGB input. In this table, parameterising the structuring elements by either parabolic or general structuring elements is also shown.

				None				Conv				Deconv			
	P	U	C	#params	mIoU	acc	bf	#params	mIoU	acc	bf	#params	mIoU	acc	bf
Linear	3	-	3	12.6M	0.381	0.694	0.314	25.1M	**0.396**	**0.707**	0.329	25.1M	**0.398**	**0.709**	0.329
Standard Pool	2	3	3	0	0.298	0.646	0.270	12.6M	0.351	0.682	0.312	12.6M	0.353	0.682	0.310
MorphPool	2	3	3	0	0.348	0.675	0.330	12.6M	0.368	0.697	0.346	12.6M	0.381	0.699	**0.349**
MorphPool	3	5	3	0	0.387	0.695	0.334	12.6M	0.363	0.701	0.342	12.6M	0.379	0.705	0.344
Para. MorphPool	3	5	3	4.0K	0.383	0.695	0.335	12.6M	0.382	0.701	0.346	12.6M	0.387	0.706	**0.349**
General MorphPool	3	5	3	67.5K	**0.388**	**0.698**	**0.337**	12.6M	0.394	0.704	**0.352**	12.6M	0.390	0.707	0.346

Table 6: **NYUv2 Segmentation on RGB Results.** Similar to 2D-3D-S and SUN-RGBD on image input, results show that **(a)** morphological pooling outperforms standard pooling; **(b)** morphological pooling performs on par or slightly better than linear sampling, although at much reduced parameter count; and **(c)** morphological operations perform better at semantic boundaries.

				None				Conv				Deconv			
	P	U	C	#params	mIoU	acc	bf	#params	mIoU	acc	bf	#params	mIoU	acc	bf
Linear	3	-	3	12.6M	0.321	0.578	0.191	25.1M	**0.339**	0.595	0.194	25.1M	**0.338**	**0.597**	0.194
Standard Pool	2	3	3	0	0.193	0.514	0.198	12.6M	0.281	0.555	**0.234**	12.6M	0.271	0.556	**0.229**
MorphPool	2	3	3	0	0.295	0.550	0.219	12.6M	0.316	0.574	0.227	12.6M	0.314	0.577	0.223
MorphPool	3	5	3	0	0.312	0.584	0.203	12.6M	0.335	**0.599**	0.213	12.6M	0.330	0.591	0.210
Para. MorphPool	3	5	3	4.0K	0.309	0.571	0.193	12.6M	0.328	0.588	0.209	12.6M	0.325	0.582	0.206
General MorphPool	3	5	3	67.5K	**0.327**	**0.586**	**0.211**	12.6M	0.326	0.593	0.208	12.6M	0.322	0.587	0.210

Table 7: **NYUv2 Depth Auto-encoding Results.** Similar to semantic segmentation on depth, morphological operations are well suited to sampling features from a Depth modality. Morphological pooling & unpooling outperform linear sampling and standard pooling on this task. Parameterising the morphological operations with a free kernel yields best performance.

				None			Conv			Deconv		
	P	U	C	ARD ↓	RMS ↓	acc < 1.25 ↑	ABS ↓	RMS ↓	acc<1.25 ↑	ARD ↓	RMS ↓	acc<1.25 ↑
Linear	3	-	3	0.193	0.563	0.715	0.190	0.566	0.716	0.199	0.594	0.703
Standard Pool	2	3	3	0.239	0.751	0.605	0.224	0.687	0.634	0.225	0.695	0.633
MorphPool	2	3	3	0.180	0.526	0.733	0.166	0.496	**0.774**	0.180	0.523	0.738
MorphPool	3	5	3	0.177	0.534	0.734	0.169	0.495	0.760	0.174	**0.508**	0.749
General MorphPool	3	5	3	**0.171**	**0.505**	**0.747**	**0.161**	**0.462**	**0.774**	**0.171**	0.510	**0.753**

Table 8: **Depth-wise Convolution Results for Depth on NYU**. Similar to in the main text, it is possible to use depth-wise convolutions to equalise the number of parameters available during up-sampling. Gray cells indicate networks that have the same number of available parameters to learn interpolation for up-sampling, either morphologically or linearly. Just like on SUN-RGBD, MorphPool outperforms the linear setting on NYU.

Up / Down	None				Depth-wise Conv				Conv			
	#params	mIoU	acc	bf	#params	mIoU	acc	bf	#params	mIoU	acc	bf
Conv	12576576	0.305	0.597	0.175	12626176	0.323	0.613	0.177	47494976	0.362	0.631	0.189
Depth-wise Conv	17856	0.283	0.583	0.163	67456	0.300	0.596	0.167	34936256	0.314	0.607	0.176
Standard Pool	0	0.121	0.453	0.177	49600	0.149	0.480	0.173	34918400	0.204	0.543	0.198
MorphPool	0	0.323	0.607	0.200	49600	**0.367**	0.637	**0.211**	34918400	0.361	0.627	0.213
Para. MorphPool	3968	0.348	0.627	0.205	53568	0.366	**0.640**	0.205	34922368	0.379	0.637	0.214
General MorphPool	67456	**0.353**	**0.632**	**0.207**	117056	0.359	0.633	0.205	34985856	**0.380**	**0.643**	**0.217**

the same number of parameters MorphPool outperforms its linear counterpart. Again, the effect is most pronounced for depth data.

Table 9: **Depth-wise Convolution Results for RGB on SUN-RGB**. Similar to the experiments that compare depth and RGB input, the difference in performance between pooling methods is less pronounced. However, MorphPool outperforms the linear network with the same number of parameters.

Up / Down	None				Depth-wise Conv				Conv			
	#params	mIoU	acc	bf	#params	mIoU	acc	bf	#params	mIoU	acc	bf
Linear	12576576	0.381	0.694	0.314	12626176	0.418	**0.726**	0.346	47494976	0.353	**0.717**	0.325
Depth-wise Linear	17856	0.368	0.684	0.301	67456	0.383	0.706	0.326	34936256	0.393	0.705	0.321
Standard Pool	0	0.298	0.646	0.270	49600	0.376	0.696	0.319	34918400	0.382	0.701	0.333
MorphPool	0	0.387	0.695	0.334	49600	0.418	0.720	0.364	34918400	0.395	0.714	**0.354**
Para. MorphPool	3968	0.383	0.695	0.335	53568	0.417	0.722	0.364	34922368	**0.397**	0.715	0.353
General MorphPool	67456	**0.388**	**0.698**	**0.337**	117056	**0.419**	0.723	**0.365**	34985856	0.392	0.713	0.348

Table 10: **Depth-wise Convolution Results for RGB on NYU**. Again, MorphPool outperforms the linear setting at the same number of parameters.

Down \ Up	None				Depth-wise Conv				Conv			
	#params	mIoU	acc	bf	#params	mIoU	acc	bf	#params	mIoU	acc	bf
Conv	12576576	0.321	0.578	0.191	12626176	0.344	0.519	0.201	47494976	**0.351**	**0.607**	0.196
Depth-wise Conv	17856	0.299	0.561	0.180	67456	0.310	0.568	0.189	34936256	0.322	0.579	0.181
Standard Pool	0	0.193	0.514	0.198	49600	0.275	0.565	0.222	34918400	0.310	0.577	**0.236**
MorphPool	0	0.312	0.584	0.203	49600	0.332	0.594	0.227	34918400	0.317	0.583	0.192
Para. MorphPool	3968	0.309	0.571	0.193	53568	0.333	0.594	0.226	34922368	0.331	0.588	0.203
General MorphPool	67456	**0.327**	**0.586**	**0.211**	117056	**0.344**	**0.604**	**0.232**	34985856	0.336	0.594	0.207

References

[1] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE international conference on computer vision*, pages 1520–1528, 2015.

[2] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147. PMLR, 2013.

[3] Matthew D Zeiler, Graham W Taylor, and Rob Fergus. Adaptive deconvolutional networks for mid and high level feature learning. In *2011 international conference on computer vision*, pages 2018–2025. IEEE, 2011.