Poolability and Transferability in CNN. A Thrifty Approach

Anonymous for review

Editors: Under Review for MIDL 2020

Abstract

The current trend in deep learning models for semantic segmentation are ever increasing model sizes. These large models need huge data-sets to be trained properly. However medical applications often offer only small data-sets available and require smaller models. A large part of these models' parameters is due to their multi-resolution approach for increasing the receptive field, i.e. alternating convolution and pooling layers for feature extraction. In this work an alternative parameter free approach is proposed to increase the receptive field. This significantly reduces the number of parameters needed in semantic segmentation models and allows them to be trained on smaller data-sets.

1. Introduction

In recent years convolutional neural networks (CNN) have become the number one tool for pattern recognition. One big part of their success lies in the translation invariance. The other part, which we are going to elaborate on, is the fact that through a clever choice of architecture the network is able to make decisions considering the whole image.

The convolution operation is the core of CNN architectures. The number of parameters for a convolution grow with the square of the convolution kernel size, thus most architectures use 3×3 kernels with 9 parameters per feature map. But how to achieve a sensible prediction for an entire image, if a single convolution "sees" only a 3×3 neighborhood? The solution is stacking of convolutional layers. With two layers following each other, the last one can "see" a 4×4 neighborhood. This notion of which part of the input image the output convolution "sees" is called the **receptive field**. By stacking 3×3 layers, it increases by one per convolution. Which means a lot of convolutions must be stacked to have a receptive field as large as a reasonable input image. The increase in receptive field per convolution can be drastically higher when the image is sub-sampled to a lower resolution between two convolution operations. A popular choice for the sub-sampling operation is the maximum pooling (Ranzato et al., 2007) with a stride of 2. This stride effectively halves the image resolution. But any convolution or pooling operation with a stride larger than one can be used for sub-sampling (Springenberg et al., 2014). Figure 1 illustrates the effect of alternating convolution and strided pooling layers on the receptive field. It creates a pyramid of sequentially lower resolutions until only one output neuron remains. With such a pyramidal structure of sufficient depth the last convolution will have access to a down-sampled version of the entire image. For image segmentation tasks this down-sampled image is once again up-sampled to the original size in an inverse pyramid pattern. This concept has been known for a long time (LeCun et al., 1989) and is widely applied in today's CNNs (Lin et al., 2016).

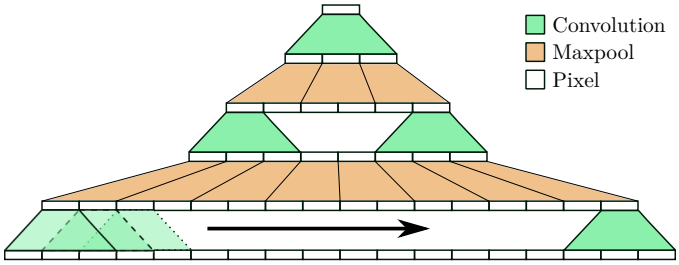


Figure 1: The combination of convolution and pooling layers allows for a receptive field which is exponentially growing with the depth of the network. This example illustrates why such multi-resolution structures are called pyramidal structures: all 18 pixels of the input on the bottom are in the receptive field of the neuron on top.

Feature extraction pyramids are the common way to increase the receptive field. Fully connected layers could also do this but have too many parameters and are not translation invariant (Kauderer-Abrams, 2017). The à trous, or dilated, convolution (Starck et al., 1998) can also increase the receptive field while maintaining a low number of parameters (Chen et al., 2017). Deformable convolutions (Dai et al., 2017) are similar but learn the position of the holes instead of using a static pattern. Both convolution variants allow to reduce the number of layers but still require a pyramidal structure. A breakout from that structure can be achieved with parallel convolutions like in the inception module (Szegedy et al., 2015) or à trous spatial pyramid pooling (ASPP) (Chen et al., 2018) which were combined later by (Liu et al., 2018). While this idea is worthwhile, it does not offer a lower parameter number compared to pyramidal structures

Due to the usual method for increasing the receptive field contemporary CNN models have several ten million parameters. We propose a method for increasing the receptive field to the whole image in a parameter free manner in only one layer. In the following our method is first described and then validated on a synthetic and a real data-set (BraTS 2017 (Menze et al., 2015)) .

2. Extending the Receptive Field

The expansion of the receptive field is made possible thanks to the introduction of an original module called the **transfer block** which uses much less parameters than the usual pyramidal feature extractors and does not modify the resolution of the previous feature map.

The transfer block (figure 2) consists of three layers. The core part is the transfer layer which is enclosed by two convolution layers. This transfer layer builds upon the idea that information needs to be transferred across the feature map in order to have a maximum receptive field for the final neuron. This is achieved through parallel maximum pooling operations with different window sizes.

At first the input is passed through a initial $w \times w$ convolution layer with zero padding which yields n feature maps of the same size as the input. Then each feature map is passed

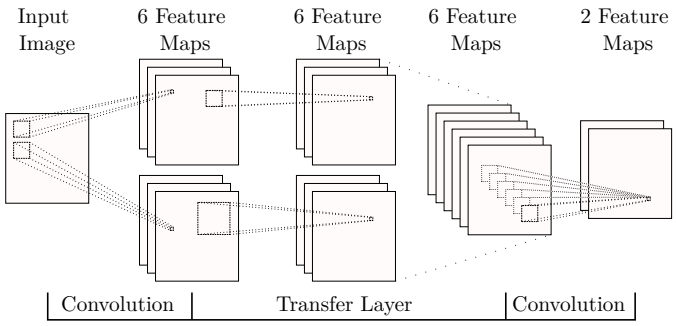


Figure 2: An example of the whole transfer block where $m = 3$ and with only two window sizes. The input image is passed through a convolution layer which produces 6 feature maps. Each of the 6 feature maps is then subject to a maximum pooling operation with stride one, where 3 are processed with the first window size and the other 3 are processed with the second window size. After concatenation the final convolution layer produces the output feature maps

through a maximum pooling with stride one (the transfer layer). What makes this pooling step special is that k different window sizes $l_1 \times l_1, l_2 \times l_2, \ldots, l_k \times l_k$ are used. Each window size is used on m feature maps, with $km = n$. The resulting feature maps are then input to a final convolution layer with n_2 feature maps as output. The feature maps all have the same size as the input image, i.e. no down-sampling occurs.

The transfer layer is responsible for the increase of the receptive field. The largest window size l_k determines the size of the receptive field. For an input of size $s \times s$ it is necessary that $l_k = 2s$ to ensure the receptive field is as large as the input image. The other window sizes determine the scales for which information is collected. The initial convolution ensures that the features are relevant for each scale. The multiplicity m allows to collect multiple features per scale. Maximum pooling looses the spatial orientation of the features. The final convolution layer recovers this information from the neighboring pixels.

For our experiments we chose $m = 10$ and $l_1 = 1$, $l_i = 2^{i-1} + 1$ with $k = \lfloor \log_2(s) + 2 \rfloor$. This is a good trade off between the network size and performance.

The transfer block may seem similar to the ASPP (Chen et al., 2018) at first glance but there are fundamental differences. ASPP uses à trous convolutions to increase the receptive field. This always considers only 9 pixels in the convolution kernel, which looses too much information for large dilation rates. The transfer block does not have blind spots and the full image information is always available. Also the convolutions cost parameters while the maximum pooling is parameter free. Then the number of parameters in the transfer block can be arbitrarily tuned with m and n_2 whereas ASPP does not have this option.

These properties make the transfer block a lightweight replacement for feature extraction pyramids.

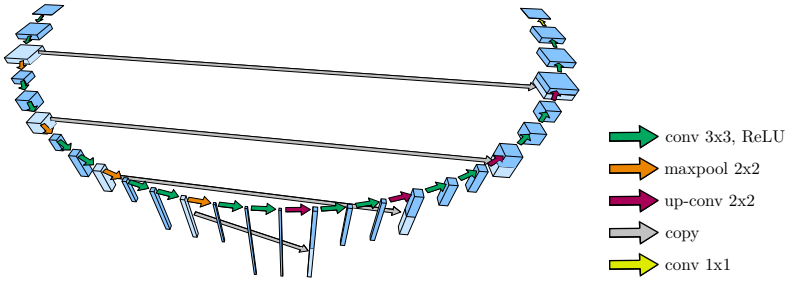
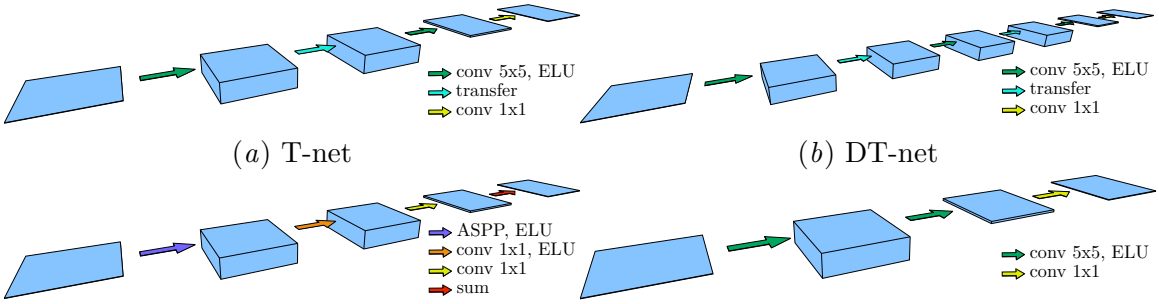


Figure 3: The U-Net architecture with its encoder and decoder structure. Arrows represent operations and cubes represent feature maps where the height of the cube stands for the number of feature maps and the width and depth of the cubes for the size of the feature maps. The original U-Net uses 5 different resolutions, thus we call it U-Net 5. A variant with 4 resolutions (U-Net 4) is also included in our experiments.



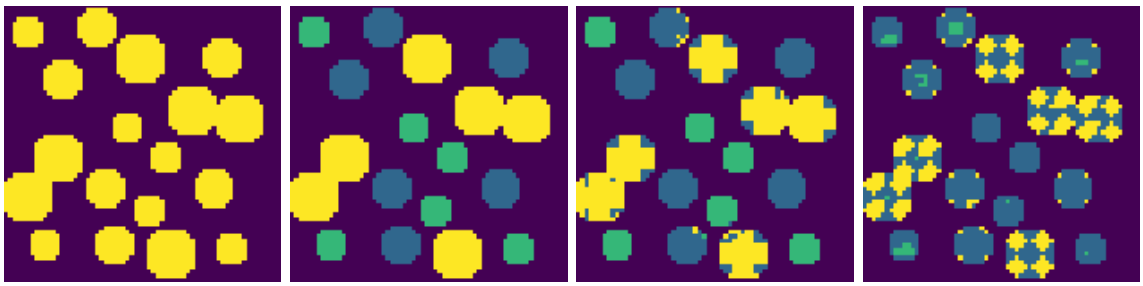
(a) T-net (b) DT-net

(c) ASPP module (d) CNN reference (R-Net)

Figure 4: (a) Transfer-Net (T-Net): transfer block with $n_2 = k$ plus a 1×1 convolution layer (b) Double Transfer-Net (DT-Net): chains together transfer blocks. (c) The ASPP module increases the receptive field with dilated convolutions. Each dilated convolution produces 64 feature maps in our configuration. (d) CNN reference: the most basic CNN architecture. It serves as a reference that demonstrate the effects of an insufficient receptive field.

3. Experiments

Two experiments validate that the transfer block is working and compare it to a U-Net model (Ronneberger et al., 2015). The U-Net was chosen because it is the archetype of modern convolutional networks used for bio-medical image segmentation tasks and achieved good performance in many applications. Note that we do not expect to beat the U-Net with a model that has less than 1% of its parameters. We simply want to demonstrate that the transfer block is a working feature extractor and can be a building block to decrease the number of parameters of larger models in the future.



(a) (b) (c) (d)

Figure 5: One example of the circles validation-set. The binary image (a) had to be segmented according to the ground truth (b). The U-Nets and the Transfer-Nets perfectly segmented the image and delivered exactly (b). The result of the R-Net is shown in (c). As the receptive field of the R-Net is not large enough to capture the largest circles entirely it has trouble discerning the corners of the large and middle circles as they are identical. The ASPP module is inapt for the task d).

As common in medical segmentation task we use the Dice coefficient (Dice Lee R., 1945) to measure the segmentation quality. The Dice of two sets A and B is

$$Dice = \frac{2A \cap B}{|A| + |B|} \tag{1}$$

3.1. Architectures

The tested network architectures are outlined in figures 3 and 4. They are all followed by a softmax layer and trained with Adam (Kingma and Ba, 2014) using cross entropy as loss function. The networks are trained in parallel, i.e. all networks receive the identical mini-batches in the same order, on 64×64 input batches. Training was continued long after convergence to see if over-fitting would occur.

Table 1: Network configurations for the circles experiment and their final dice score which is identical for training and validation-set.

Model	Kernel Size	Parameters	Learning Rate	m	Dice
U-Net 5	3	31030788	0.001	-	1.0
U-Net 4	3	7696388	0.002	-	1.0
DT-Net	5	136679	0.0005	10	1.0
T-Net	5	14109	0.002	10	1.0
R-Net	5	14109	0.005	10	0.92
ASPP	3	106260	0.005	-	0.24

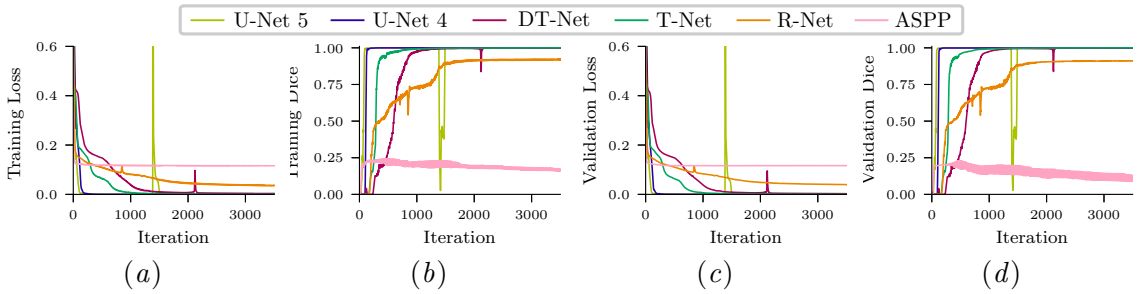


Figure 6: Development of the loss (a), (c) and the dice coefficient (b), (d) on the training and validation set for the circles data-set. The curves have been smoothed with a low pass filter. This smoothing was applied to the learning curves in figure 7 as well.

3.2. Synthetic Data

As a simple example of multi class segmentation binary images of size 64×64 with circles with a radius of 3, 4 and 5 pixels are generated. The circles are placed randomly and do not overlap but may touch each other. Each circle size is a class for segmentation. The training-set contained 20 images and the validation-set 10 images. The small number of samples was chosen to show that the architectures can learn with few examples, as we target bio-medical applications where data-sets tend to be very small. The point is that the circles are larger than the convolution kernels. Thus the models needs to have a working enlargement of the receptive field. The R-Net does not have this and has problems with the larger circles (figure 5, (c)). The à trous convolutions of ASPP prevent it from gathering the necessary local information of the circle and it fails the task (figure 5, (d)). The U-Net learns the task with ease and the T-Net shows that the increase of the receptive field works, albeit with a slightly longer time to convergence (figure 6).

3.3. BraTS

For the second experiment we chose a real data set which is renowned for its difficulty. The brain tumor segmentation (BraTS) (Menze et al., 2015) challenge is a recurring challenge attached to the MICCAI Conference. Each year the segmentation results become better, but the problem is an ongoing research. For this experiment we use the high grade glioma part of the BraTS 2017 data-set[1]. It contains multi-modal MRI of 210 patients which were manually segmented by experts, i.e. a ground truth is available. On these images three different classes have to be segmented from the background: the enhancing tumor, the necrotic and non-enhancing tumor and as a third class the peritumoral edema. This makes it an ideal real data-set for supervised learning of a multi-class segmentation task.

The networks are trained on 64×64 patches using all 4 available MRI modalities, i.e. 4 input channels. Each mini-batch is forced to have 50% of the patches showing a tumour to counter class imbalance. The patches are sampled such that the center is inside the brain. For the training-set 100 patients were randomly chosen and 4 different patients for the

1. www.med.upenn.edu/sbia/brats2017.html

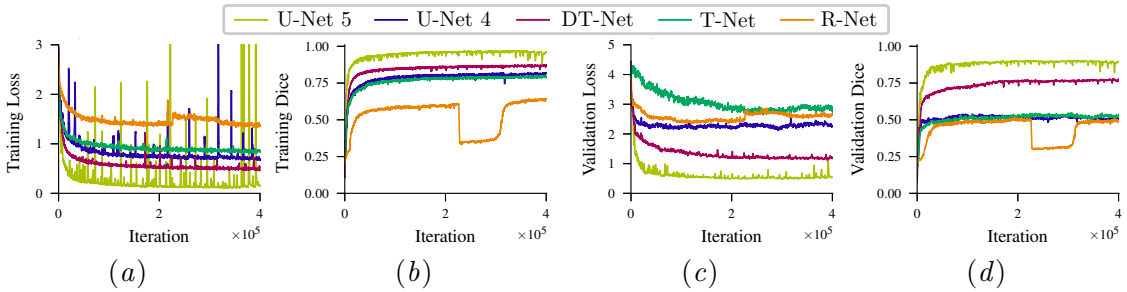


Figure 7: Development of the loss (a), (c) and the dice coefficient (b), (d) on the training and validation set of the BraTS data-set. They represent the best performance for each architecture. Remarkable is that the validation loss does not seem to correspond to the validation dice. The reason is that the Dice is calculated from a thresholded confidentiality map and fluctuations below the threshold have no effect on the Dice.

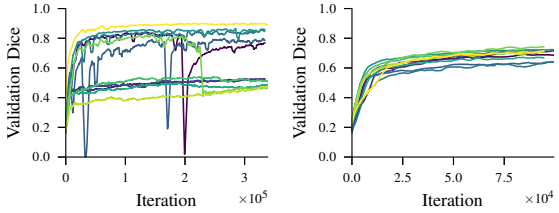


Figure 8: Validation Dice for multiple runs for the U-Net 5 a) and the DT-Net b). The network configurations are identical between runs. The U-Net is unstable, different models trained in the same configuration converge to largely different solutions. With a bit of bad luck we would have never seen a well performing U-Net and the conclusion would be that the U-Net 5 is outperformed by the DT-Net. This actually happened in the first version of this work. The DT-Net however always converges to similar solutions and is therefore more reliable.

Table 2: Network configurations for the BraTS experiment and their final dice score on the training and validation set.

Model	U-Net 5	U-Net 4	DT-Net	T-Net	R-Net
Kernel Size	3	3	5	5	5
Parameters	31032516	7698116	141929	19359	19359
Learning Rate	0.001	0.002	0.0005	0.002	0.002
m	-	-	10	10	-
Training Dice	0.96	0.82	0.87	0.80	0.64
Validation Dice	0.89	0.51	0.77	0.51	0.49

validation-set. As the validation set is not used to stop the training it is truly independent from the training set. Our goal is to observe the networks' training behavior and not the segmentation performance, so the test-set is omitted here.

During the training of these models we noticed that the U-Net would converge to very different solutions in the same training configuration, often not reaching its potential. The convergence of the U-Net and the DT-Net are compared in figure 8. In contrast to the transfer block based networks, the U-Net seems to be inherently unstable. The figure shows only the validation Dice, but the training loss and Dice show the same behavior. For the moment we assume that this is an effect of the network size, but it would be interesting to investigate this further.

This real data-set is way more difficult than segmenting circles, which allows to see a real difference in the capabilities of the different network architectures. Looking at the loss values the networks are well separated (figure 7). But a look at the dice score shows a lower loss value does not necessarily indicate a better segmentation performance. The networks based on the transfer block perform well. Of course we cannot expect to beat the U-Net 5 with a network which has only 0.44% of the parameters. But the DT-Net comes remarkably close in terms of performance and clearly beats the U-Net 4 which still has 56 times more parameters. Even the T-Net beats the U-Net 4 on the validation set while having less than twenty thousand parameters compared to more than seven million of the U-Net 4.

This proves that the transfer block works, and that it delivers performance at an astonishingly low parameter cost.

4. Conclusion

We challenged the concept for feature extraction which has been uncontested for three decades, the feature extraction pyramid. Our method translates the series of resolutions to parallel maximum pooling with different window sizes. The receptive field of our method can be modified freely through the pooling window sizes without affecting the parameter number, whereas traditional feature extraction pyramids have a high parameter cost associated with an increase of the receptive Field.

In two experiments it was shown that our transfer block can master the same tasks as a feature extraction pyramids. Not only the transfer block is a drop in replacement for a feature extraction pyramid, i.e. another way to do things, it also has advantages over feature extraction pyramids. It provides comparable performance while using an astonishing less than 1% of the parameters of a proven segmentation method, the U-Net. And as a smaller network it is more robust and converges reliably to the same result from different random initializations.

In the future our method for increasing the receptive field will lead to much smaller and faster networks while maintaining the known CNN performance. This is interesting for applications where computational resources are limited, like autonomous driving or real-time bio-medical applications.

References

Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *CoRR*, June 2017.

Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, April 2018. doi: 10.1109/tpami.2017.2699184.

Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE, oct 2017. doi: 10.1109/iccv.2017.89.

Dice Lee R. Measures of the amount of ecologic association between species. *Ecology*, 26(3):297–302, 1945. ISSN 00129658, 19399170. doi: 10.2307/1932409. URL http://www.jstor.org/stable/1932409.

Eric Kauderer-Abrams. Quantifying translation-invariance in convolutional neural networks. *CoRR*, December 2017.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, December 2014.

Y. LeCun, L. D. Jackel, B. Boser, J. S. Denker, H. P. Graf, I. Guyon, D. Henderson, R. E. Howard, and W. Hubbard. Handwritten digit recognition: Applications of neural net chips and automatic learning. *IEEE Communication*, pages 41–46, November 1989. invited paper.

Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie. Feature pyramid networks for object detection. *CoRR*, abs/1612.03144, 2016. URL http://arxiv.org/abs/1612.03144.

Songtao Liu, Di Huang, and Yunhong Wang. Receptive field block net for accurate and fast object detection. In *Computer Vision – ECCV 2018*, pages 404–419. Springer International Publishing, 2018. doi: 10.1007/978-3-030-01252-6_24.

B. H. Menze, A. Jakab, S. Bauer, J. Kalpathy-Cramer, K. Farahani, J. Kirby, Y. Burren, N. Porz, J. Slotboom, R. Wiest, L. Lanczi, E. Gerstner, M. Weber, T. Arbel, B. B. Avants, N. Ayache, P. Buendia, D. L. Collins, N. Cordier, J. J. Corso, A. Criminisi, T. Das, H. Delingette, Ç. Demiralp, C. R. Durst, M. Dojat, S. Doyle, J. Festa, F. Forbes, E. Geremia, B. Glocker, P. Golland, X. Guo, A. Hamamci, K. M. Iftekharuddin, R. Jena, N. M. John, E. Konukoglu, D. Lashkari, J. A. Mariz, R. Meier, S. Pereira, D. Precup, S. J. Price, T. R. Raviv, S. M. S. Reza, M. Ryan, D. Sarikaya, L. Schwartz, H. Shin, J. Shotton, C. A. Silva, N. Sousa, N. K. Subbanna, G. Szekely, T. J. Taylor, O. M. Thomas, N. J. Tustison, G. Unal, F. Vasseur, M. Wintermark, D. H. Ye, L. Zhao, B. Zhao, D. Zikic, M. Prastawa, M. Reyes, and K. Van Leemput. The multimodal brain tumor image segmentation benchmark (BRATS). *IEEE Transactions on Medical Imaging*, 34 (10):1993–2024, October 2015. ISSN 0278-0062. doi: 10.1109/TMI.2014.2377694.

M. Ranzato, F. J. Huang, Y. Boureau, and Y. LeCun. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2007. doi: 10.1109/CVPR.2007.383157.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015. URL http://arxiv.org/abs/1505.04597.

Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin A. Riedmiller. Striving for simplicity: The all convolutional net. *CoRR*, abs/1412.6806, 2014. URL http://arxiv.org/abs/1412.6806.

Jean-Luc Starck, Fionn D. Murtagh, and Albert Bijaoui. *Image Processing and Data Analysis. The Multiscale Approach*, volume 94. Cambridge University Press, 1998. ISBN 978-0-521-59914-6. doi: 10.1017/cbo9780511564352.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, jun 2015. doi: 10.1109/cvpr.2015.7298594.