# Learning Steerable Filters for Rotation Equivariant CNNs
# Supplementary Material

## 1. Equivariance properties

In this section we prove the equivariance of the individual layers of Steerable Filter CNNs under rotations by the sampled orientations in $\Theta$, assuming signals on a continuous domain $\mathbb{R}^2$. Translational equivariance follows directly from either the utilization of spatial convolutions or from the independence of the operation on the spatial position.

### 1.1. Input layer

The first layer maps an image $I : \mathbb{R}^2 \to \mathbb{R}$ to a feature map $\zeta^{(1)} : \mathbb{R}^2 \rtimes \Theta \to \mathbb{R}$ by first convolving it with multiple rotated versions $\rho_\theta \Psi$ of a filter $\Psi : \mathbb{R}^2 \to \mathbb{R}$ and subsequently adding a bias $\beta$ and applying a nonlinearity $\sigma$. Both steps are equivariant under rotations of the image by angles $\alpha \in \Theta$. This means that $\rho_\alpha I(x)$ is mapped to $\mathcal{R}_\alpha \zeta^{(1)}(x,\theta) = \rho_\alpha \zeta^{(1)}(x, \theta - \alpha)$ where $\mathcal{R}_\alpha$ is the group action on functions on the group. To see that the first step performs an equivariant mapping, simply insert a rotated image,

$$\left(\rho_\alpha I * \rho_\theta \Psi\right)(x) \;=\; \int\limits_{\mathbb{R}^2} I(\rho_{-\alpha} u)\, \Psi(\rho_{-\theta}(x-u))\, \mathrm{d}u \;,$$

and substitute $\tilde{u} := \rho_{-\alpha} u$. Since the transformation is orthogonal we have $\left|\det\left(\frac{\partial \tilde{u}}{\partial u}\right)\right| = 1$ and hence:

$$\begin{aligned}\left(\rho_\alpha I * \rho_\theta \Psi\right)(x) \;&=\; \int\limits_{\mathbb{R}^2} I(\tilde{u})\, \Psi(\rho_{-(\theta-\alpha)}(\rho_{-\alpha} x - \tilde{u}))\, \mathrm{d}\tilde{u} \\ &= \rho_\alpha \left(I * \rho_{\theta-\alpha} \Psi\right)(x) \\ &= \rho_\alpha y^{(1)}(x, \theta - \alpha) \\ &= \mathcal{R}_\alpha y^{(1)}(x,\theta) \;.\end{aligned}$$

The mutual transformation behavior is visualized in the following commutative diagram:

$$\begin{tikzcd}
I(x) \arrow[r, "*\rho_\theta \Psi"] \arrow[d, "\rho_\alpha"'] & y^{(1)}(x,\theta) \arrow[d, "\mathcal{R}_\alpha"] \\
\rho_\alpha I(x) \arrow[r, "*\rho_\theta \Psi"'] & \rho_\alpha y^{(1)}(x, \theta - \alpha)
\end{tikzcd}$$

Adding a bias $\beta$ to each feature map channel and applying a nonlinearity $\sigma$ does not interfere with translational- or rotational equivariance since both operations do neither depend on the spatial position nor orientation channel:

$$\begin{tikzcd}
y^{(l)}(x,\theta) \arrow[r, "\sigma(\,\cdot\, + \beta)"] \arrow[d, "\mathcal{R}_\alpha"'] & \zeta^{(l)}(x,\theta) \arrow[d, "\mathcal{R}_\alpha"] \\
\rho_\alpha y^{(l)}(x, \theta - \alpha) \arrow[r, "\sigma(\,\cdot\, + \beta)"'] & \rho_\alpha \zeta^{(l)}(x, \theta - \alpha)
\end{tikzcd}$$

### 1.2. Group-convolutional layers

Given feature maps $\zeta^{(l)}(x,\theta)$, the group-convolutional layers perform an equivariant mapping of $\mathcal{R}_\alpha \zeta^{(l)}(x,\theta)$ to $\mathcal{R}_\alpha \zeta^{(l+1)}(x,\theta)$ under the group action $\mathcal{R}$. The step of adding the bias and applying the activation function is equivariant by the same argument as in the first layer. What is left to show is the equivariance $\left(\mathcal{R}_\alpha \zeta^{(l)} \circledast \Psi\right)(x,\theta) = \mathcal{R}_\alpha \left(\zeta^{(l)} \circledast \Psi\right)(x,\theta) = \mathcal{R}_\alpha y^{(l)}(x,\theta)$ of the group convolution. Inserting a transformed feature map and writing the group convolution out explicitly yields:

$$\begin{aligned}&\left(\mathcal{R}_\alpha \zeta^{(l)} \circledast \Psi\right)(x,\theta) \\ &= \int\limits_{\mathbb{R}^2} \sum_{\phi \in \Theta} \zeta^{(l)}(\rho_{-\alpha} u, \phi - \alpha)\, \Psi(\rho_{-\phi}(x-u), \theta - \phi)\, \mathrm{d}u \,.\end{aligned}$$

Again, we substitute $\tilde{u} := \rho_{-\alpha} u$ with $\left|\det\left(\frac{\partial \tilde{u}}{\partial u}\right)\right| = 1$. Furthermore, we let $\tilde{\phi} := \phi - \alpha$ under which the sum is invariant thanks to the cyclic structure of the subgroup $\Theta$, and we obtain

$$\begin{aligned}
&\left(\mathcal{R}_\alpha \zeta^{(l)} \circledast \Psi\right)(x,\theta) \\
=&\int\limits_{\mathbb{R}^2} \sum_{\tilde{\phi}\in\Theta} \zeta^{(l)}(\tilde{u},\tilde{\phi})\, \Psi(\rho_{-\tilde{\phi}}(\rho_{-\alpha}x-\tilde{u}),(\theta-\alpha)-\tilde{\phi})\,\mathrm{d}\tilde{u} \\
=&\left(\zeta^{(l)} \circledast \Psi\right)(\rho_{-\alpha}x,\theta-\alpha) \\
=&\rho_\alpha y^{(l+1)}(x,\theta-\alpha) \\
=&\mathcal{R}_\alpha y^{(l+1)}(x,\theta)\,.
\end{aligned}$$

This proves the equivariance of the intermediate layers. Again, the relations are illustrated in a commutative diagram:

$$\begin{tikzcd}
\zeta^{(l)}(x,\theta) \arrow[r, "\circledast\Psi"] \arrow[d, "\mathcal{R}_\alpha"'] & y^{(l+1)}(x,\theta) \arrow[d, "\mathcal{R}_\alpha"] \\
\rho_\alpha\zeta^{(l)}(x,\theta-\alpha) \arrow[r, "\circledast\Psi"'] & \rho_\alpha y^{(l+1)}(x,\theta-\alpha)
\end{tikzcd}$$

### 1.3. Orientation max-pooling layer

For rotation-invariant segmentation or classification we max-pool over orientations after the last group-convolutional layer. The pooling step is itself equivariant and results in a rotated version of its output:

$$\begin{aligned}
\max_\theta \mathcal{R}_\alpha \zeta^{(l)}(x,\theta) &= \max_\theta \rho_\alpha \zeta^{(l)}(x,\theta-\alpha) \\
&= \rho_\alpha \left(\max_\theta \zeta^{(l)}(x,\theta-\alpha)\right) \\
&= \rho_\alpha \left(\max_\theta \zeta^{(l)}(x,\theta)\right).
\end{aligned}$$

The rotation operator commutes with the maximum over orientation channels because it acts on spatial coordinates only. We again visualize the transformation behavior by a commutative diagram:

$$\begin{tikzcd}
\zeta^{(l)}(x,\theta) \arrow[r, "\max\limits_\theta"] \arrow[d, "\mathcal{R}_\alpha"'] & \max_\theta \zeta^{(l)}(x,\theta) \arrow[d, "\rho_\alpha"] \\
\rho_\alpha\zeta^{(l)}(x,\theta-\alpha) \arrow[r, "\max\limits_\theta"'] & \rho_\alpha \max_\theta \zeta^{(l)}(x,\theta)
\end{tikzcd}$$

In the case of classification the remaining spatial structure is pooled out such that the output is invariant under transformations of the input.

Instead of the maximum pooling which we applied in our experiments, one could also utilize average pooling layers. The equivariance of average pooling can be derived in analogy to the derivation for maximum pooling.

## 2. Derivation of the generalized He weight initialization scheme

In this section we give the derivation of the generalized weight initialization scheme whose results are stated in the main paper. For completeness we recall the assumptions going into the following calculations. We consider the activation of a single neuron in layer $l$,

$$\zeta^{(l)}_{\hat{c}x} = \max(0, y^{(l)}_{\hat{c}x}), \tag{1}$$

where rectified linear units were chosen as nonlinearities. The pre-nonlinearity activations are given by the convolution with filters $\Psi$ and summing over the input channels:

$$\begin{aligned}
y^{(l)}_{\hat{c}x} &= \sum_c \left(\zeta^{(l-1)}_c * \Psi^{(l)}_{\hat{c}c}\right)_x + \beta^{(l)}_{\hat{c}} \\
&= \sum_c \sum_{x'} \zeta^{(l-1)}_{c,x-x'} \Psi^{(l)}_{\hat{c}cx'} + \beta^{(l)}_{\hat{c}}.
\end{aligned} \tag{2}$$

For convenience we shifted the addition of the bias to the pre-nonlinearity activations. The filters are defined by

$$\Psi^{(l)}_{\hat{c}cx} = \sum_{q=1}^{Q} w^{(l)}_{\hat{c}cq} \psi_{qx}\,,$$

that is, they are built from $Q$ real valued atomic filters $\psi_q$. We keep the discussion general by not restricting the atomic filters to be steerable. In analogy to Glorot and Bengio [1] and He et al. [2] we assume the activations and gradients to be i.i.d. and to be independent from the weights. We let the weights themselves be mutually independent and have zero mean but do *not* restrict them to be identically distributed because of the inherent asymmetry coming from the different atomic filters. Furthermore we initialize all biases to be zero.

### 2.1. Backpropagation

In order to prevent vanishing or exploding gradients of the loss $\mathcal{E}$ due to inappropriate initialization we demand their variance $\mathrm{Var}\left[\frac{\partial\mathcal{E}}{\partial\zeta^{(l)}}\right]$ to be constant across all layers. It follows from (1) and (2) that the gradient with respect to the activation $\zeta^{(l)}_{c_0x_0}$ of a particular neuron in layer $l$ is given by

$$\begin{aligned}
\frac{\partial\mathcal{E}}{\partial\zeta^{(l)}_{c_0x_0}} &= \sum_{\hat{c},x} \frac{\partial\mathcal{E}}{\partial y^{(l+1)}_{\hat{c}x}} \frac{\partial y^{(l+1)}_{\hat{c}x}}{\partial\zeta^{(l)}_{c_0x_0}} \\
&= \sum_{\hat{c},x} \frac{\partial\mathcal{E}}{\partial\zeta^{(l+1)}_{\hat{c}x}} \mathbb{I}_{y^{(l+1)}_{\hat{c}x}>0} \sum_q w^{(l+1)}_{\hat{c}c_0q} \psi_{q,x-x_0}\,,
\end{aligned} \tag{3}$$

where the indicator function $\mathbb{I}$ stems from the derivative of the rectified linear unit. Like He et al. [2] we assume the factors occurring in (3) to be statistically independent. Observing that $\mathbb{E}\left[w^{(l)}\right]$ and therefore also $\mathbb{E}\left[\frac{\partial\mathcal{E}}{\partial\zeta^{(l)}}\right]$ vanish, and without loss of generality setting $x_0 = 0$ this leads to

$$\begin{aligned}
&\mathrm{Var}\left[\frac{\partial\mathcal{E}}{\partial\zeta^{(l)}_{c_0x_0}}\right] = \mathbb{E}\left[\left(\frac{\partial\mathcal{E}}{\partial\zeta^{(l)}_{c_0x_0}}\right)^2\right] \\
&= \sum_{\hat{c},\hat{c}'}\sum_{x,x'}\sum_{q,q'}\mathbb{E}\left[\frac{\partial\mathcal{E}}{\partial\zeta^{(l+1)}_{\hat{c}x}}\frac{\partial\mathcal{E}}{\partial\zeta^{(l+1)}_{\hat{c}'x'}}\right]\mathbb{E}\left[\mathbb{I}_{y^{(l+1)}_{\hat{c}x}>0}\mathbb{I}_{y^{(l+1)}_{\hat{c}'x'}>0}\right] \\
&\qquad\cdot\ \mathbb{E}\left[w^{(l+1)}_{\hat{c}c_0q}w^{(l+1)}_{\hat{c}'c_0q'}\right]\psi_{q,x}\psi_{q',x'} \\
&= \sum_{\hat{c}}\sum_{x}\sum_{q}\mathbb{E}\left[\left(\frac{\partial\mathcal{E}}{\partial\zeta^{(l+1)}_{\hat{c}x}}\right)^2\right]\mathbb{E}\left[\mathbb{I}_{y^{(l+1)}_{\hat{c}x}>0}\right] \\
&\qquad\cdot\ \mathbb{E}\left[\left(w^{(l+1)}_{\hat{c}c_0q}\right)^2\right]\psi^2_{q,x} \\
&= \sum_{\hat{c}}\sum_{x}\sum_{q}\frac{1}{2}\mathrm{Var}\left[\frac{\partial\mathcal{E}}{\partial\zeta^{(l+1)}_{\hat{c}x}}\right]\mathrm{Var}\left[w^{(l+1)}_{\hat{c}c_0q}\right]\psi^2_{q,x}.
\end{aligned}$$

The factor $\frac{1}{2}$ in the last line originates from the symmetric distribution of $y^{(l)}$ in conjunction with the indicator function. Using the fact that the weights' variances are initialized to only depend on $q$ and the assumption of identically distributed gradients, both can be pulled out of the sums:

$$\begin{aligned}
&\mathrm{Var}\left[\frac{\partial\mathcal{E}}{\partial\zeta^{(l)}}\right] \\
&= \mathrm{Var}\left[\frac{\partial\mathcal{E}}{\partial\zeta^{(l+1)}}\right]\frac{\hat{C}}{2}\sum_q \mathrm{Var}\left[w^{(l+1)}_q\right]\left\|\psi_q\right\|_2^2 .
\end{aligned}$$

It seems reasonable to assign the contribution to the overall variance equally to the $Q$ summands. Demanding the gradients' variances to be constant over layers then leads to the initialization condition

$$\mathrm{Var}\left[w_q\right] = \frac{2}{\hat{C}Q\left\|\psi_q\right\|_2^2}.$$

## 2.2. Forward pass

The calculation for the forward pass is similar to the case of backpropagation but considers the variance $\mathrm{Var}\left[y^{(l)}\right]$ of pre-nonlinearity activations instead of gradients. As an exact calculation depends on the expectation value $\mathbb{E}\left[\zeta^{(l-1)}\right]$, which is not known, we approximate the result by exploiting the central limit theorem. To this end, we note that the pre-nonlinearity activations (2) are summed up from $C\sum_q|\operatorname{supp}\psi_q|$ independent terms of finite variance which is a relatively large number in typical networks. This allows to approximate the variance by the asymptotic result implied by the central limit theorem:

$$\begin{aligned}
&\mathrm{Var}\left[y^{(l)}_{\hat{c}x}\right] \\
&= \mathrm{Var}\left[\sum_c\sum_{x'}\sum_q \zeta^{(l-1)}_{c,x-x'}w^{(l)}_{\hat{c}cq}\psi_{qx'}\right] \\
&\overset{\text{(CLT)}}{\approx} \sum_c\sum_{x'}\sum_q \mathrm{Var}\left[\zeta^{(l-1)}_{c,x-x'}w^{(l)}_{\hat{c}cq}\psi_{qx'}\right] \\
&= \sum_c\sum_{x'}\sum_q \mathbb{E}\left[\left(\zeta^{(l-1)}_{c,x-x'}\right)^2\right]\mathbb{E}\left[\left(w^{(l)}_{\hat{c}cq}\right)^2\right]\psi^2_{qx'} .
\end{aligned}$$

In the last step we made use of the independence of the weights from the previous layer's feature maps and $\mathbb{E}[w] = 0$. The symmetric distribution of weights leads to a symmetric distribution of pre-nonlinearity activations which in conjunction with ReLU nonlinearities implies $\mathbb{E}[\zeta^2] = \frac{1}{2}\mathrm{Var}[y]$. To see this, note that the symmetry of the distribution of pre-nonlinearity activation leads on the one hand to

$$\begin{aligned}
\mathrm{Var}[y] &= \mathbb{E}[y^2] \\
&= \int_{\mathbb{R}}\tilde{y}^2\,p_y(\tilde{y})\,\mathrm{d}\tilde{y} \\
&= 2\int_{\mathbb{R}+}\tilde{y}^2\,p_y(\tilde{y})\,\mathrm{d}\tilde{y}
\end{aligned}$$

and on the other hand to

$$\begin{aligned}
\mathbb{E}[\zeta^2] &= \int_{\mathbb{R}}\tilde{\zeta}^2\,p_\zeta(\tilde{\zeta})\,\mathrm{d}\tilde{\zeta} \\
&= \int_{\mathbb{R}}\tilde{\zeta}^2\left(\frac{1}{2}\delta(0) + \Theta(\tilde{\zeta})p_y(\tilde{\zeta})\right)\mathrm{d}\tilde{\zeta} \\
&= \int_{\mathbb{R}+}\tilde{\zeta}^2\,p_y(\tilde{\zeta})\,\mathrm{d}\tilde{\zeta}\,,
\end{aligned}$$

where $\delta$ denotes the delta distribution and $\Theta$ is the Heaviside step function. As before, we drop all indices which the random variables are independent from to compute the sums. This leads to

$$\mathrm{Var}\left[y^{(l)}\right] \approx \mathrm{Var}\left[y^{(l-1)}\right]\frac{C}{2}\sum_q \mathrm{Var}\left[w^{(l)}_q\right]\left\|\psi_q\right\|_2^2 ,$$

which in turn suggests a weight initialization according to

$$\mathrm{Var}\left[w_q\right] = \frac{2}{CQ\left\|\psi_q\right\|_2^2}$$

to ensure that the activations' variances are not amplified.

| Operation | Filter Size | Feature Channels |
|---|---|---|
| Steerable input layer | $7 \times 7$ | 16 |
| Steerable group convolution | $5 \times 5$ | 24 |
| Spatial max pooling | $2 \times 2$ | |
| Steerable group convolution | $5 \times 5$ | 32 |
| Steerable group convolution | $5 \times 5$ | 32 |
| Spatial max pooling | $2 \times 2$ | |
| Steerable group convolution | $5 \times 5$ | 48 |
| Steerable group convolution | $5 \times 5$ | 64 |
| Global spatial pooling | | |
| Global orientation pooling | | |
| Fully connected | | 64 |
| Fully connected | | 64 |
| Fully connected + Softmax | | 10 |

Table 1: Architecture of the SFCNN used in the initial experiments on the resolution of sampled orientations and the rotational generalization.

| Operation | Filter Size | Feature Channels |
|---|---|---|
| Steerable input layer | $9 \times 9$ | 24 |
| Steerable group convolution | $7 \times 7$ | 32 |
| Spatial max pooling | $2 \times 2$ | |
| Steerable group convolution | $7 \times 7$ | 36 |
| Steerable group convolution | $7 \times 7$ | 36 |
| Spatial max pooling | $2 \times 2$ | |
| Steerable group convolution | $7 \times 7$ | 64 |
| Steerable group convolution | $5 \times 5$ | 96 |
| Global spatial pooling | | |
| Global orientation pooling | | |
| Fully connected | | 96 |
| Fully connected | | 96 |
| Fully connected + Softmax | | 10 |

Table 2: Architecture of the SFCNN used with $\Lambda = 16$ sampled orientations in the final benchmarking experiments on rotated MNIST.

### 2.3. Normalization of complex atomic filters

The results derived above suggest to initialize the weights of each layer uniformly by

$$\mathrm{Var}\left[w_q\right] = \frac{2}{CQ\left\|\psi_q\right\|_2^2} \quad \text{or} \quad \mathrm{Var}\left[w_q\right] = \frac{2}{\hat{C}Q\left\|\psi_q\right\|_2^2}$$

after normalizing the atomic filters to $\left\|\psi_q\right\|_2 = 1$. An additional complication arises in our network construction where steerability is only preserved when the relative amplitude of the filters' real and imaginary parts is not changed. While for circular harmonics both parts have equal norms in continuous space, this is not necessarily true for their sampled versions which rules out an independent normalization of the real and imaginary parts. As a steerability consistent way of normalizing circular harmonics, we propose to adequately normalize their complex modulus. The proper scale follows from $\|\psi\|_2^2 = \|\mathrm{Re}\,[\psi]\|_2^2 + \|\mathrm{Im}\,[\psi]\|_2^2$ for $\psi \in \mathbb{C}$ to be $\|\psi\|_2 = 1$ for DC filters whose imaginary part vanishes and $\|\psi\|_2 = \sqrt{2}$ for non-DC filters.

## 3. Details on the experimental setup

Here we give further details on the network architectures and the training setup of our experiments.

### 3.1. Rotated MNIST

For our initial experiments on the dependence on sampled orientations and the networks' rotational generalization capabilities we utilize the architecture given in Table 1 as baseline. Based on the results of these experiment we fix the number of sampled orientations to $\Lambda = 16$ and tune the network architecture further. We achieve the best benchmark results using the slightly larger network given in Table 2. In particular, we found that increasing the size of the filter masks improved the results. Both architectures consist of one steerable input layer which maps the input images to the group, five following group convolutional layers and three fully connected layers. After every two steerable filter layers we perform a spatial $2 \times 2$ max-pooling. The orientation dimension and the remaining spatial dimensions are pooled out globally after the last convolutional layer. We normalize the activations by adding batch normalization layers [3] after each convolutional and fully connected layer. The batch normalization on the group does not interfere with the equivariance when the responses are normalized by averaging over both spatial and orientation dimensions.

The number of feature channels stated in the tables refers to the number of learned filters $\hat{C}$ of the corresponding layer. As these filters are themselves applied with respect to $\Lambda$ orientations we end up with $\hat{C}\Lambda$ responses; e.g. $24 \cdot 16 = 384$ effective responses in the first layer of the smaller network. Note that the extraction of this comparatively large number of responses without overfitting is possible because the rotational weight sharing leads to an increased parameter utilization (in the sense of Cohen and Welling [4]) by a factor $\Lambda$.

All networks are trained for 40 epochs using the Adam optimizer [5] with standard parameters. The initial learning rate is set to 0.015 and is decayed exponentially with a rate of 0.8 per epoch starting from epoch 15. We regularize the weights with an elastic net penalty with hyperparameters $\lambda_{L1} = \lambda_{L2}$ which are set to $10^{-7}$ and $10^{-8}$ for the convolutional and fully connected layers respectively. Dropout [6] is used only in the fully connected layers with a dropping probability of $p = 0.3$.

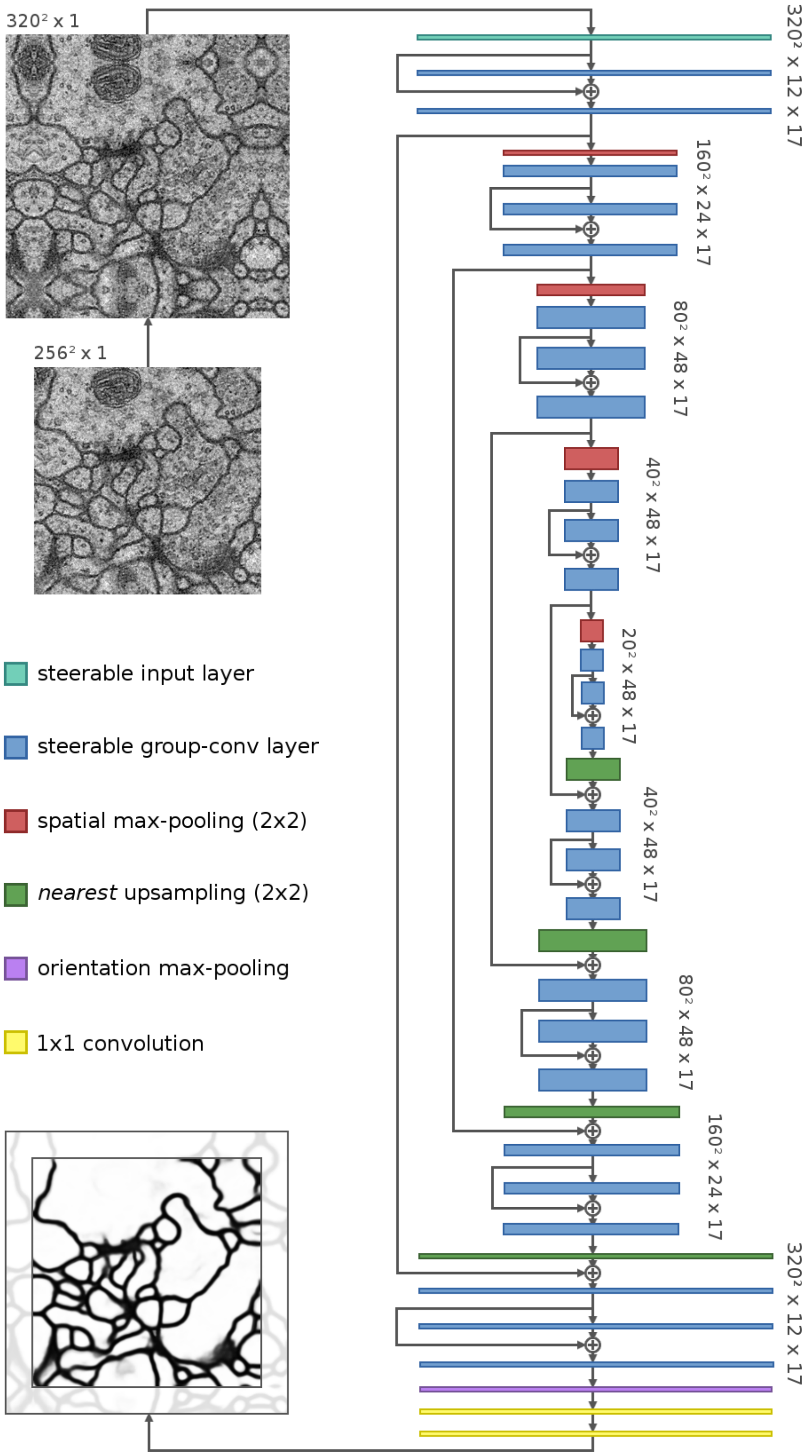

Figure 1: Network architecture used to predict the membrane probability map for the ISBI 2012 EM segmentation challenge. The topology is inspired by the U-Net [7] and FusionNet [8] but uses the proposed steerable group-convolution layers with $\Lambda = 17$ orientations. To mitigate boundary artifacts we feed reflect-padded images into the network.

### 3.2. ISBI 2012 EM segmentation challenge

The network architecture used to segment the membranes from raw EM images of neural tissue for the ISBI EM segmentation challenge is visualized in Figure 1. Inspired by the U-Net [7] it is build as a symmetric encoder-decoder network with additional skip-connections between stages of the same resolution. This allows to extract semantic information from a large field of view while at the same time preserving precise spatial localization. Further, we adopt two modifications from [8]: we do not concatenate the skipped feature maps but add it to the decoder features upsampled from the previous stage, and we use intermediate residual blocks (here of depth 1). On the highest resolution level we learn $\hat{C} = 12$ filters, applied in $\Lambda = 17$ orientations which corresponds to $\hat{C}\Lambda = 204$ effective channels. The number of filters is doubled when going to the second and third level and is afterwards kept constant since we did not observe further gains in performance when adding more channels. All group-convolutional layers utilize kernels of size $7 \times 7$ pixels while the input layer applies $11 \times 11$ pixel kernels.

As input, we feed the network cropped regions of $256 \times 256$ pixels which are padded to $320 \times 320$ pixels by reflecting a region of 32 pixels around the borders to alleviate boundary artifacts. The padded regions are augmented by random elastic deformations, reflections and rotations by multiples of $\frac{\pi}{2}$. After the decoder we max-pool over orientations to obtain locally invariant features and crop out $256 \times 256$ pixels centrally. Two subsequent $1 \times 1$ convolution layers map these features pixel-wise to the desired probability map.

The network is optimized by minimizing the spatially averaged binary cross-entropy loss between predictions and the ground truth segmentation masks using the ADAM optimizer. As on the rotated MNIST dataset we regularize the convolutional weights with an elastic net penalty with hyperparameters $\lambda_{L1} = \lambda_{L2}$ set to $10^{-7}$ and $10^{-8}$ for the steerable and $1 \times 1$ convolution layers respectively. Here we chose a dropout probability of $p = 0.4$ both in the steerable as well as in the $1 \times 1$ convolution layers. The learning rate is decayed exponentially by a factor of $0.85$ per epoch starting from an initial rate of $5 \cdot 10^{-2}$.

## References

[1] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2010. 2

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1026–1034. 2, 3

[3] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning (ICML)*, 2015, pp. 448–456. 4

[4] T. Cohen and M. Welling, "Steerable CNNs," in *International Conference on Learning Representations (ICLR)*, 2017. 4

[5] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, 2015. 4

[6] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014. 4

[7] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2015, pp. 234–241. 5, 6

[8] T. M. Quan, D. G. Hilderbrand, and W.-K. Jeong, "Fusionnet: A deep fully residual convolutional neural network for image segmentation in connectomics," *arXiv preprint arXiv:1612.05360*, 2016. 5, 6