Christoph Rieger, Bsc.

Hierarchical architectures for spiking Winner-Take-All networks

Master's Thesis

to achieve the university degree of

Master of Science

Master's degree programme: Biomedical Engineering

submitted to

Graz University of Technology

Supervisor

Univ.-Prof. Dipl.-Ing. Dr.techn. Robert Legenstein

Institute of Theoretical Computer Science
Head: Univ.-Prof. Dipl.-Ing. Dr.techn. Robert Legenstein

Graz, April 2023

Contents

1 Protocol of work so far

1.1 Experiment 1: Rotated Bars

1.1.1 Introduction

The goal of this task was to recreate example 2 from Nessler et al. (2013). In this example they fed a winner-take-all spiking neural network images with bars in different orientations on them. The network then clustered the images into ten groups depending on their orientation.

1.1.2 Methods

Input data The images used in this task were generated with a size of 29 x 29 pixels. Black bars with a width of 7 pixels going through the center of the image were drawn onto a white background. To simulate noise each pixel had a chance of ten percent to have its color flipped. To ensure that all bars in the images have the same length regardless of their orientation a circular mask with a radius of 15 pixels was applied to the images. This recolored all pixels outside of the mask to white. During the training of the network one image per iteration was generated in a uniformly distributed orientation and each image was shown for 200 ms. For the training of the network 4000 of these images were shown. The randomly chosen orientation could lie between 0 and 359 degrees. Two examples of such images can be seen in Figure 1.1.

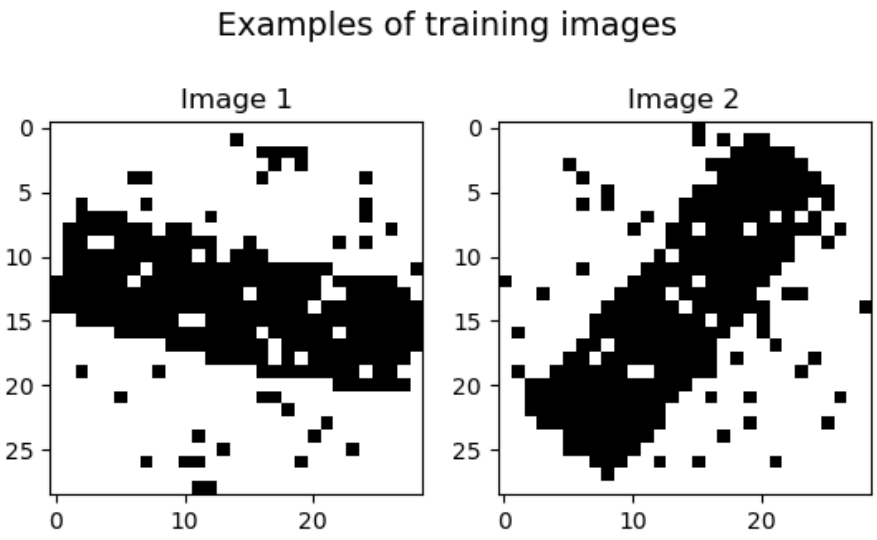

Figure 1.1: Two examples of the generated training data in this experiment.

Neuron model As in Nessler et al. (2013) the input neurons X are firing according to a poisson process with an average firing rate of 20 Hz when active and with 0 Hz when in an inactive state. The excitatory post synaptic potentials (EPSPs) $x_i(t)$ that these neurons produce can be seen in Figure 1.2. A double exponential kernel was used to generate the EPSP. The time constant for the rise of the signal τ_{rise} was 1 ms and the time constant for the decay of the signal τ_{decay} was 15 ms. The addition of the time step size δt was necessary to get the time t at the end of the current simulation step. t_f is the time at which the spike of x_i occurred

$$x_i(t) = e^{-(t+\delta t-t_f)/\tau_{decay}} - e^{-(t+\delta t-t_f)/\tau_{rise}}. \tag{1.1}$$

The firing rate of an output neuron y_k depends exponentially on its membrane potential u_k and the inhibitory signal I(t) it receives

$$r_k(t) = e^{u_k(t)-I(t)}. \tag{1.2}$$

The probability of an individual output neuron to fire within a time step δt is given by

$$r_k(t) \cdot \delta t. \tag{1.3}$$

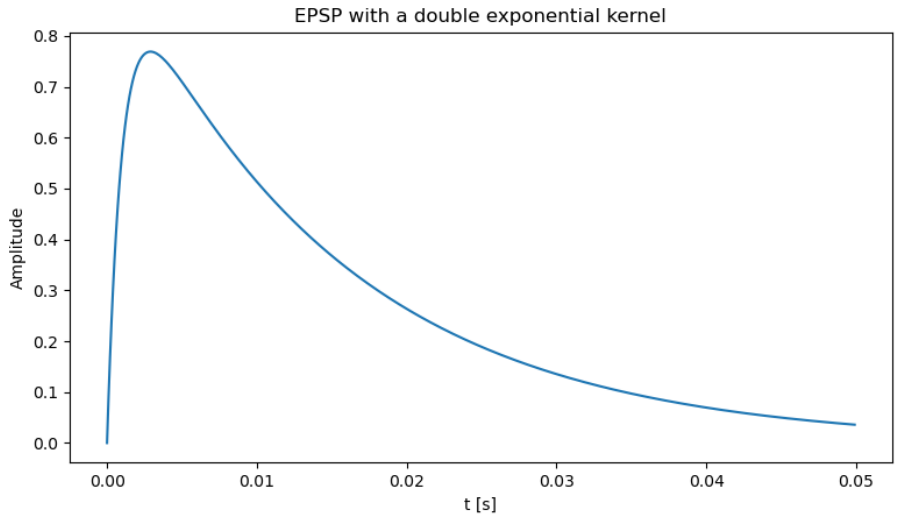

Figure 1.2: Form of an excitatory post synaptic potential generated by an input neuron over time. A double exponential kernel was used to generate this signal. These signals are fed to the next layer of the network.

Network architecture Each pixel of an input image was connected to two neurons. The first of these neurons is in an active state when the pixel is black and in an inactive state otherwise. The second neuron expresses the opposite behaviour. As a consequence the network needs 1682 ($29 \cdot 29 \cdot 2$) excitatory input neurons $x_1, ..., x_n$. These input neurons are fully connected to ten excitatory output neurons $y_1, ..., y_k$. This means that that every input neuron x_i is connected to each output neuron y_k. The membrane potential u_k of each output neuron is calculated by multiplying the EPSP of each input neuron times the weight of the connection between the input and output neuron

$$u_k(t) = \sum_{i=1}^{n} w_{ki} \cdot x_i(t). \tag{1.4}$$

In Nessler et al. (2013) each output neuron y_k also had an intrinsic excitability w_{k0} which was learned for each neuron. For this experiment however it was omitted, as each orientation of input images was equally likely, thus the intrinsic excitability of each output neuron would end up being the same.

The output neurons are modelled in a winner-takes-all (WTA) circuit. This means that whenever one output neuron spikes, a lateral inhibitory signal is fed to all output neurons, thus preventing the further activation of them for $\sigma_{inh} = 5ms$. After completion of the training of the network each output neuron should be active for bars in a coherent area. Each of those areas should ideally be of equal size. As the bars can be oriented between 0 and 359 degrees, but each 180 degree rotation of an image results in an equivalent image, the desired size of the active area of an output neuron should be 18 degrees.

Inhibition The inhibition signal was chosen to depend on the current membrane potential of the output neurons. According to Nessler et al. (2013) the total firing rate of the output neurons is

$$R(t) = \sum_{k=1}^{K} e^{u_k(t) - I(t)}. \tag{1.5}$$

Solving this equation for I(t) yields

$$R(t) = \frac{\sum_{k=1}^{K} e^{u_k(t)}}{e^{I(t)}} \tag{1.6}$$

$$e^{I(t)} = \frac{\sum_{k=1}^{K} e^{u_k(t)}}{R(t)} \tag{1.7}$$

$$I(t) = \ln \frac{\sum_{k=1}^{K} e^{u_k(t)}}{R(t)} \tag{1.8}$$

$$I(t) = -\ln R(t) + \ln \sum_{k=1}^{K} e^{u_k(t)}. \tag{1.9}$$

When implementing the inhibition the $-\ln R(t)$ term of Equation 1.9 was overlooked, that means it was assumed to be zero. Because of that R(t) equals 1 when the inhibition is active. This error was not detected at first, as the chance that a Y neuron fires within a time step of 1 ms with active inhibition is 1/1000 due to that oversight.

Whenever an output neuron produces a spike the inhibition signal I(t) is subtracted from the membrane potential Uk(t) of every output neuron. This happens for the duration of $\sigma_{inh} = 5ms$. Thus follows

$$I(t) = \begin{cases} \ln(\sum_{i=1}^{k} e^{u_k}) & \text{if any } y_k \text{ fired in } [t^f, t^f + \sigma_{inh}] \\ 0 & \text{if any } y_k \text{ did not fire in } [t^f, t^f + \sigma_{inh}]. \end{cases} \tag{1.10}$$

Spike timing dependent plasticity The weights w_{ki} between neurons x_i and y_k are updated whenever an output neuron fires. The time window σ was set to 10 ms according to Nessler et al. (2013). If y_k produces a spike all its weights are updated as

$$\Delta w_{ki} = \begin{cases} \lambda \cdot (ce^{-w_{ki}} - 1) & \text{if } x_i \text{ fired in } [t^f - \sigma, t^f] \\ \lambda \cdot (-1) & \text{if } x_i \text{ did not fire in } [t^f - \sigma, t^f], \end{cases} \tag{1.11}$$

where λ is the learning rate, the parameter c shifts the weight values, t^f is the time when y_k spiked and σ is the time window in which input spikes are considered as "before" an output spike. As the membrane potentials u_k of the output neurons result from the addition of the EPSPs of the 1682 input neurons times the corresponding weight, a way to control the average size of u is needed. If u is to small the output neurons will fire too sparsely and if u is too big it will impair the learning process. So to limit u, the size of the weights is controlled via the parameter c. The learning rate λ is needed to control the size of each weight update. If it is too big few output neurons will take over more than the expected 18° areas and others will only respond for smaller areas or not at all. On the other hand if λ is too small the network will learn very slowly and may never converge. Due to these two parameters being unwieldy to determine analytically they were chosen via grid search.

1.1.3 Results

Parameter search The two parameters c, which controls the size of the weights, and the learning rate λ were fitted to the network via grid search.

The tested parameters were as follows:

- $c = 1$ ($\lambda = 10^{-2}, 10^{-3}, 10^{-4}$)
- $c = 10$ ($\lambda = 10^{-2}, 10^{-3}, 10^{-4}$)
- $c = 20$ ($\lambda = 10^{-2}, 10^{-2.5}, 10^{-3}, 10^{-3.5}, 10^{-4}, 10^{-4.5}, 10^{-5}$)
- $c = 30$ ($\lambda = 10^{-2}, 10^{-2.5}, 10^{-3}, 10^{-3.5}$)

The simulation was conducted by simulating small discrete time steps and calculating the changes of the network in each step. The step size δt was chosen as 1 ms.

$c = 1$ The best results for $c = 1$ were achieved with $\lambda = 10^{-2}$. But overall this value for c did not work, even tough the network did learn to cluster images into eight groups depending on their orientation. In Figure 1.3 one can see which output neuron was the most active during the training process for each angle in 1° steps.

In Figure 1.4 the training progress of the network can be seen. The figure shows the number of distinct output neurons active during the presentation of each training image shown. Due to the large, compared to other values of c, learning rate the network learned quickly. However after iteration 70 there were images shown for which not a single output neuron spiked. This dying out of the networks activity is due to the parameter c being too small, which leads to too low membrane potentials.

$c = 10$ This value of $c = 10$ had the same problem of the network activity dying out as $c = 1$ although at a later point in time, as can be seen in Figure 1.5. Also in Figure 1.6 the last 50 output spikes of the training process can be seen. As indicated by Figure 1.5 the activity is sparse.

$c = 20$ $c = 20$ was the first value for which the network activity did not die out after some time. For $\lambda = 10^{-2}$ one output neuron learned to spike first for every possible input image orientation, see Figure 1.7.

With $c = 20$ and $\lambda = 10^{-3}$ the first combination that yielded a stable network was found. In Figure 1.8 the amount of distinct output neurons

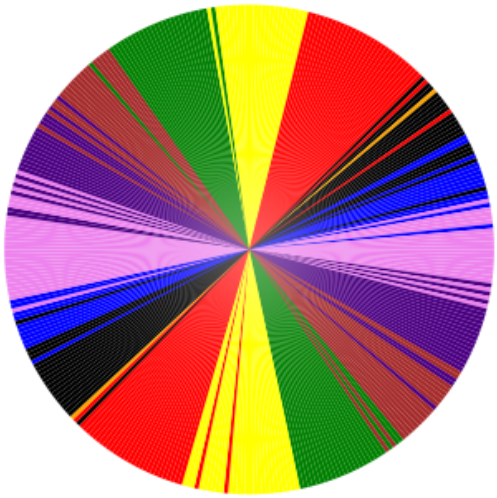

Figure 1.3: Most active output neuron depending on orientation of the training image during the training process. $c = 1, \lambda = 10^{-2}$

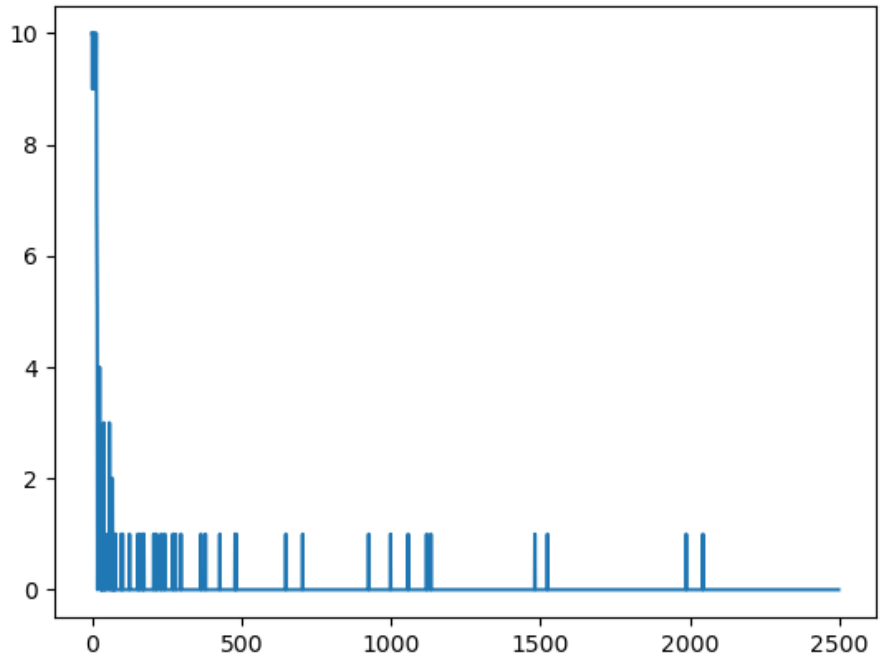

Figure 1.4: Number of distinct output neurons active during the presentation duration of each training image. x-axis: image shown, y-axis: distinct output neurons active, $c = 1, \lambda = 10^{-2}$

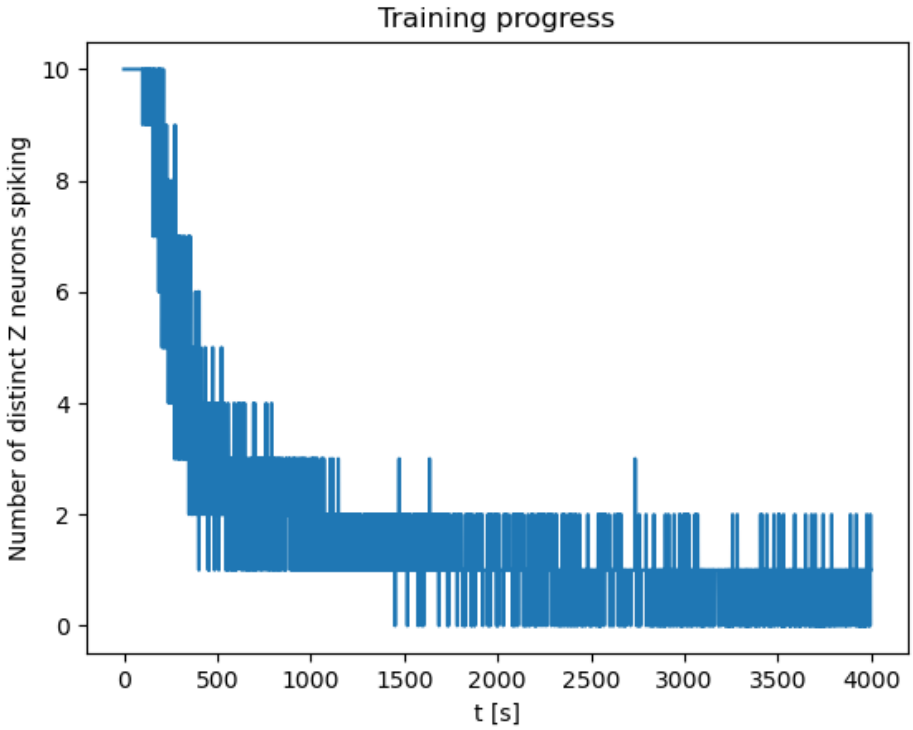

Figure 1.5: Number of distinct output neurons active during the presentation duration of each training image. x-axis: image shown, y-axis: distinct output neurons active, $c = 10, \lambda = 10^{-3}$

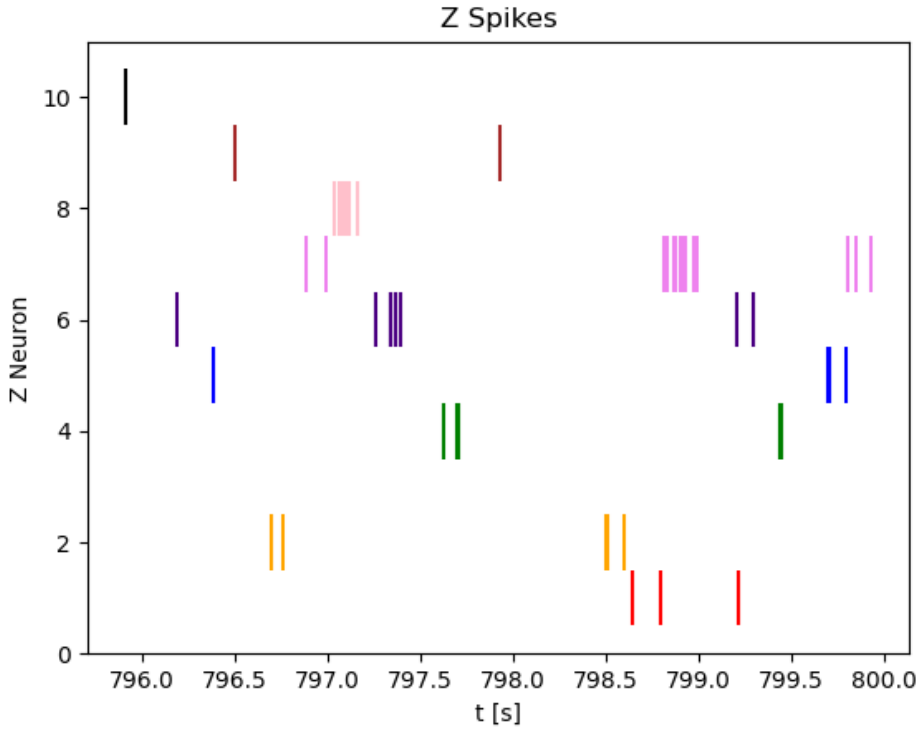

Figure 1.6: Last 50 output neuron spikes, $c = 10, \lambda = 10^{-3}$

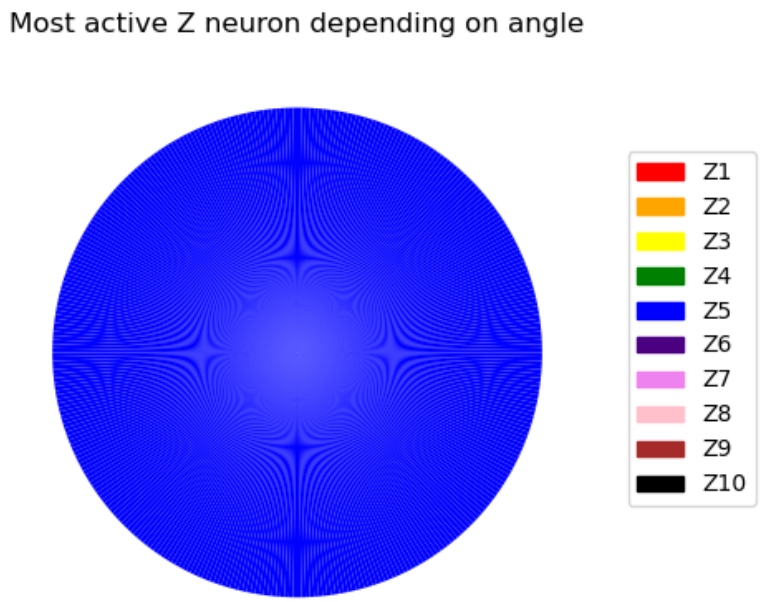

Figure 1.7: Most active output neuron depending on orientation of the training image during the training process. $c = 20, \lambda = 10^{-2}$

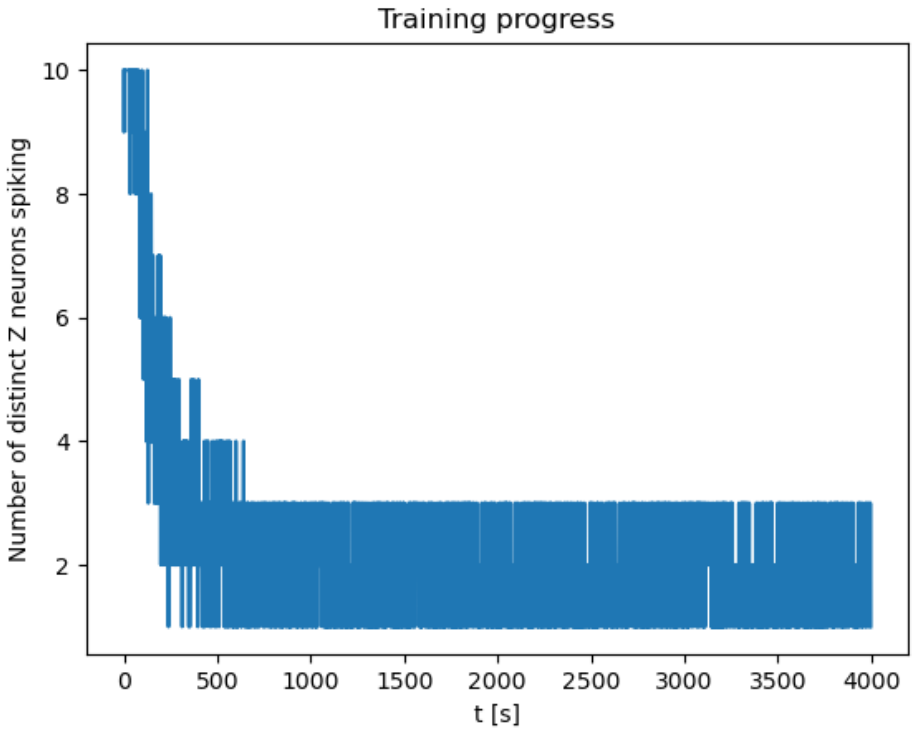

Figure 1.8: Number of distinct output neurons active during the presentation duration of each training image. x-axis: image shown, y-axis: distinct output neuron active, $c = 20, \lambda = 10^{-3}$

firing during the presentation of each training image can be seen. Figure 1.9 shows the proportion of the most active output neuron to all other output neurons active for each training image. Both of these figures can be used to measure the training progress of the network. Figure 1.9 however shows the additional information how certain the network is that a training image belongs to a specific group.

As this network did not die out after some time the trained network was analysed further. 180 images were generated in 1° steps and each was shown to the network for 200 ms. During each image presentation duration the most active output neuron was recorded. This yielded Figure 1.10.

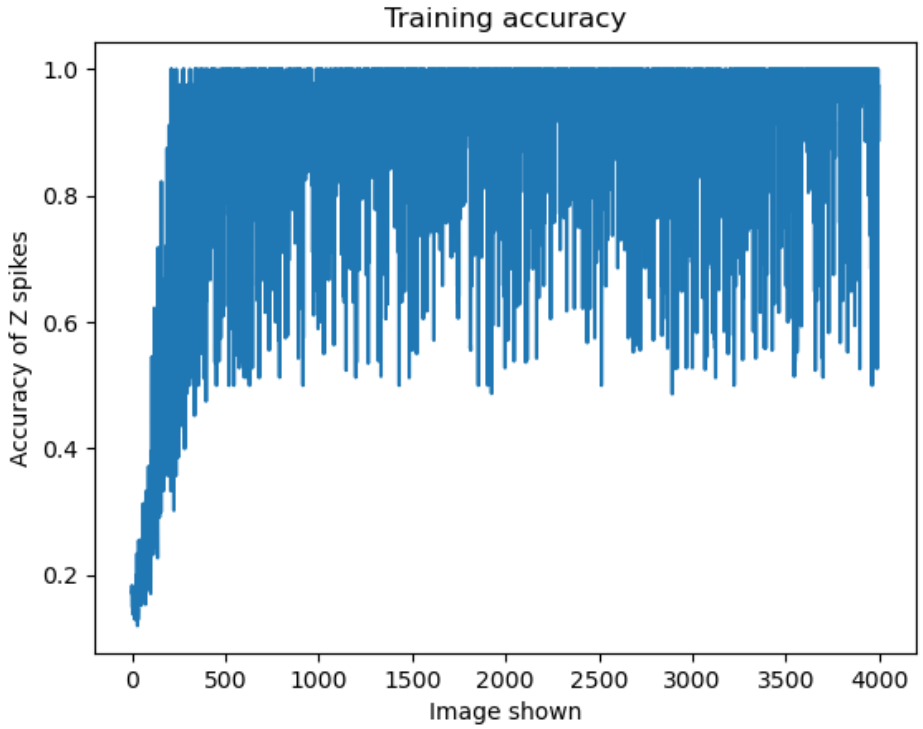

Figure 1.9: Proportion (accuracy) of most active output neuron to activity of all other output neurons during the presentation duration of each training image. $c = 20, \lambda = 10^{-3}$

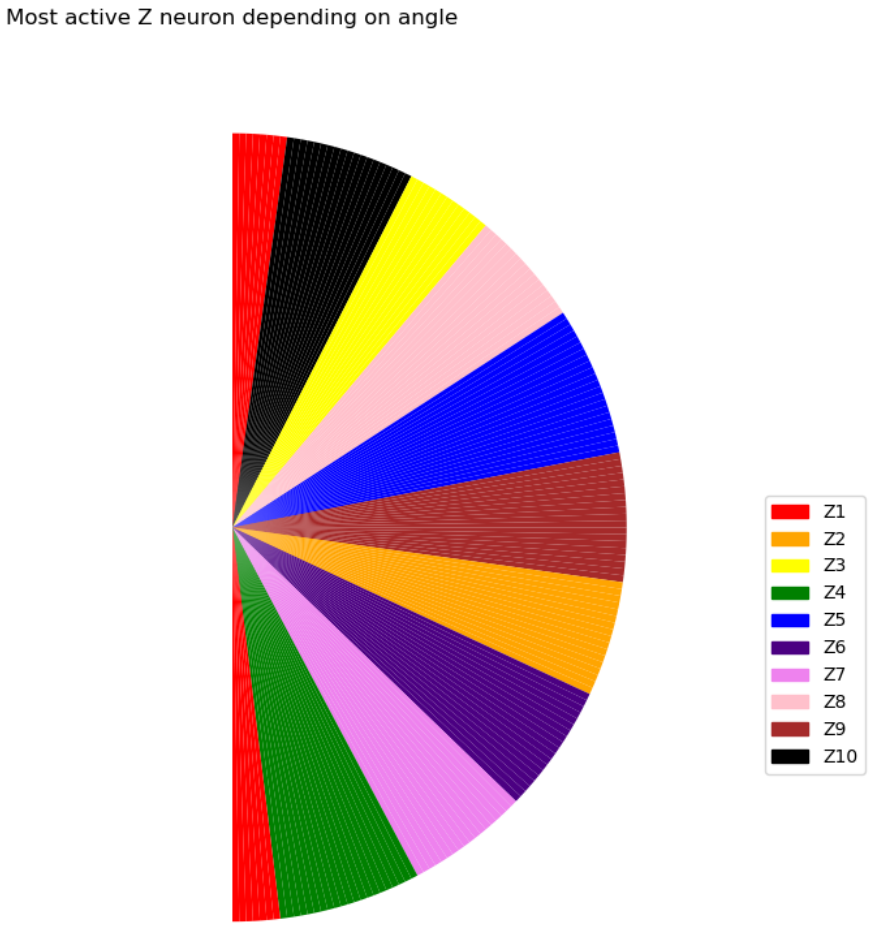

Figure 1.10: Most active output neuron depending on orientation of the training image during the training process. $c = 20, \lambda = 10^{-3}$

Also the number of distinct output neurons firing during each image presentation was recorded and is shown in 1.11. In this figure it seems that the points in which there are three distinct output neurons active are periodic in nature. This can be explained by Figure 1.12. There it can be seen that whenever the image orientation nears the border between two competing output neurons both start to be active. This explains why for large parts of Figure 1.11 2 neurons are active, as large portions of the bar in the image is overlapping the areas of the two nearest output neurons, thus producing high membrane potentials for both. The third distinct output neuron that is occasionally active seems to be of stochastic nature as much of the bars in the images overlaps areas of all other output neurons, thus generating a non zero membrane potential for each output neuron. However the occurrence of 3 distinct active output neurons seems to mostly occur at or close to the border between two competing output neurons, as there are already 2 distinct neurons firing by design.

Also it was possible to project the learned weights w_{ki} into the 2-D space to observe what they represent. This projection can be seen in Figure 1.13 for the weights of y_{10}.

The results for smaller λ were also valid, but they did not seem to yield superior results to the parameters $c = 20$ and $\lambda = 10^{-3}$. As with smaller learning rates the training simply took longer and the performance of the network did not improve they were discarded and the learning rate $\lambda = 10^{-3}$ was declared winner for $c = 20$. To save space the figures of smaller learning rates will not be shown here.

$c = 30$ also yielded a functioning network, but it did perform analogous to $c = 20$. It did perform in the same way, as with $c = 20$ the output neurons already fire every 5 ms, slowed down by the inhibition. By increasing c further the output neurons did not increase their activity.

1.1.4 Conclusion

The parameters $c = 20$ and $\lambda = 10^{-3}$ were finally chosen as the network performed the best and trained the quickest with these parameters, without

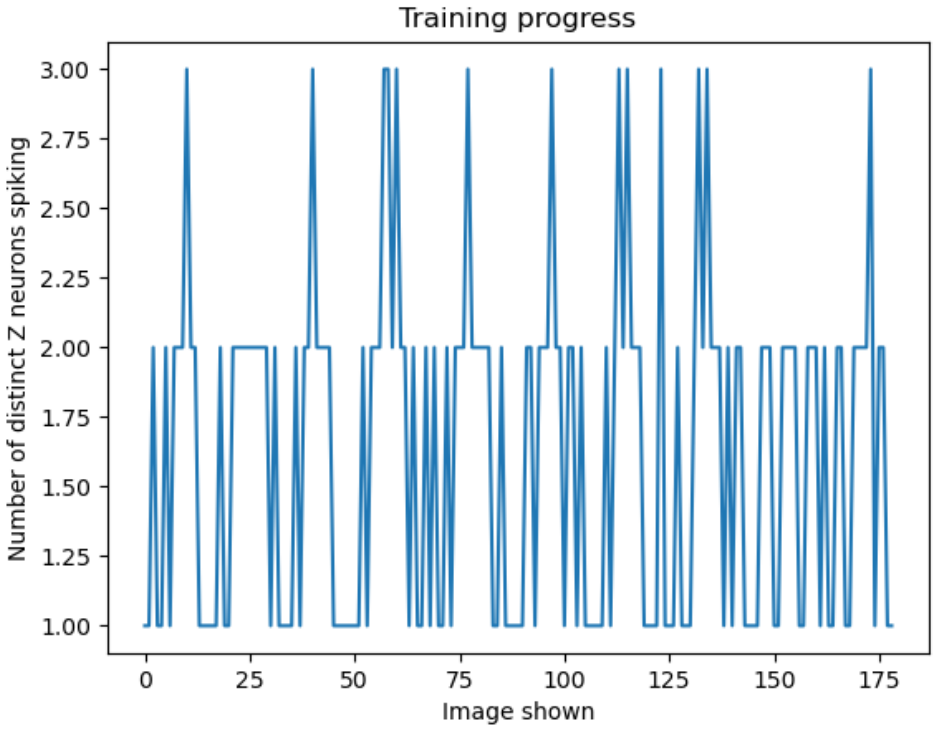

Figure 1.11: Number of distinct output neurons active during the presentation duration of each training image. x-axis: image shown, y-axis: distinct output neuron active, $c = 20, \lambda = 10^{-3}$

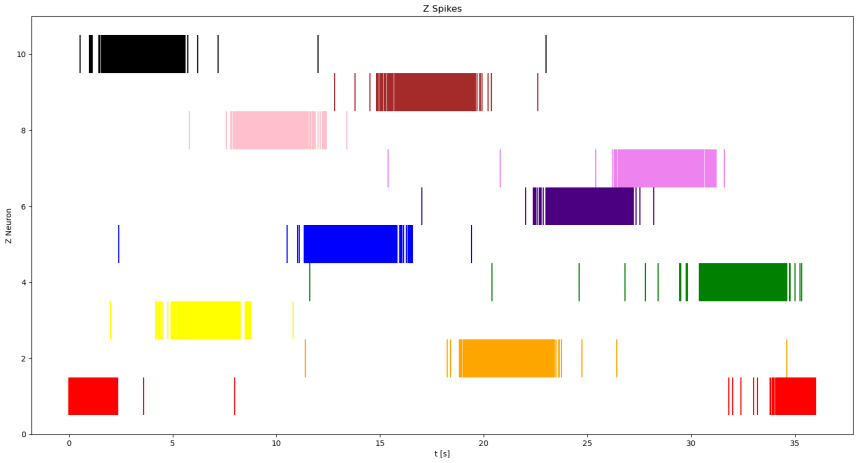

Figure 1.12: Output spikes over the presentation of input images from 0 to 179°, $c = 20, \lambda = 10^{-3}$

raising the membrane potential needlessly.

1.2 Experiment 2: Horizontal and vertical bars

1.2.1 Introduction

For this experiment the impact of a neuron layer that encodes a-priori information should be analysed.

1.2.2 Methods

Input data 29 x 29 black and white images with either horizontal or vertical oriented bars on them were used, as it was more straightforward to express a-priori information. The orientation of the training images was chosen randomly via a uniform distribution. Also the positions of the bars

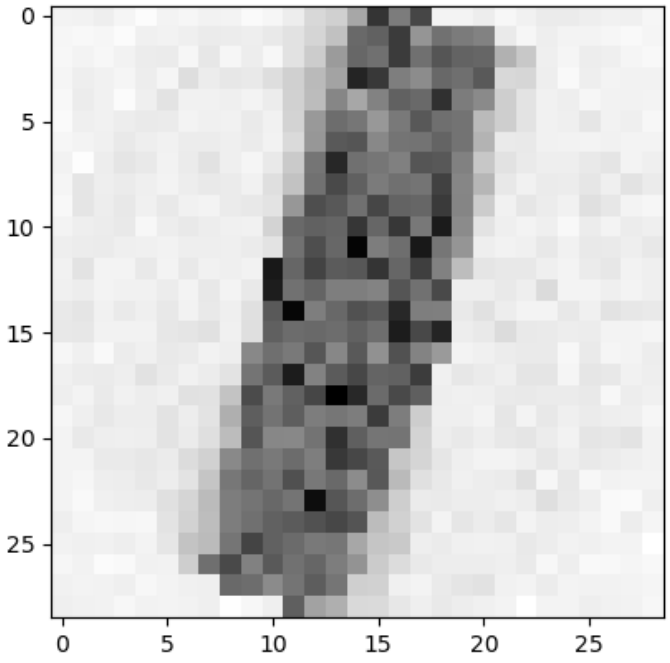

Figure 1.13: Visualization of the learned weights of y_{10}.

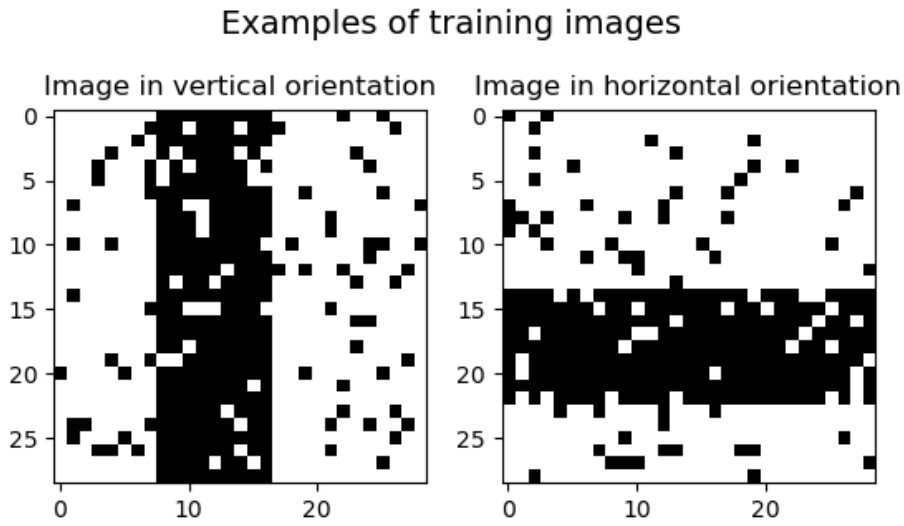

Figure 1.14: Training images generated for experiment 2. One image of each possible orientation at a random position.

in the images were uniformly distributed. The rest of the image generation process is analogous to experiment 1, except no circular mask was used. Examples of the input data can be seen in Figure 1.14. To show the value of the a-priori information validation images with two bars forming a cross were also generated, seen in Figure 1.15. When shown to the network in the validation process the prior neurons were given the information that a cross is either in horizontal or vertical orientation.

Network architecture This experiment used a expanded version of the network used in experiment 1. An additional layer of prior neurons z_1, z_2 was added. Whenever an image was oriented vertically z_1 was active and z_2 was inactive. For horizontal orientations z_2 was active and z_1 was inactive.

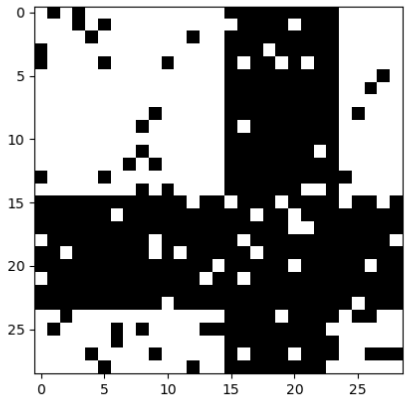

Figure 1.15: Generated cross image which can represent either horizontal or vertical orientation.

Prior neurons in an active state had a firing frequency of 50 Hz and fired with 0 Hz when inactive.

Neuron model The input and output neurons functioned the same way as in the previous experiment. The prior neurons fire according to a poisson process with their firing rate. Each prior neuron Z_l is connected to every output neuron Y_k and thus has weights w_{kl} that were learned by the network.

Parameters As the amount of input neurons and the average number of black pixels in an input image stayed roughly the same in this experiment, the same parameters $c = 20$ and $\lambda = 10^{-3}$ could be used. However as there are only two prior neurons, a way to amplify their produced signals was needed, otherwise their impact on the membrane potential of the output neurons would not be distinctive enough. So the EPSPs $z_l(t)$ were multiplied by the factor Z_{factor}. This factor was determined via grid search. The additional prior layer resulted in an expanded version of the membrane potential $u_k(t)$

$$u_k(t) = \sum_{i=1}^{n} w_{ki} \cdot x_i(t) + w_{kl} \cdot Z_{factor} \cdot z_l(t). \tag{1.12}$$

1.2.3 Results

The following values for Z_{factor} were tried with $c = 20$ and $\lambda = 10^{-3}$:

- $Z_{factor} = 3$
- $Z_{factor} = 5$
- $Z_{factor} = 7$
- $Z_{factor} = 10$
- $Z_{factor} = 20$

For $Z_{factor} = 10$ and 20 the prior neurons impacted the learning progress negatively and let single prior neurons respond to too much area. An example of this can be seen in Figure 1.16

The best results were achieved with $Z_{factor} = 5$. When looking at the training progress in Figure 1.17 it can be seen that the training accuracy is higher compared to experiment 1. This is due to the added a-priori information and the fact that less of each bar in an image is overlapping with multiple areas of output neurons. The network activity at the end of the training process can be seen in Figure 1.18. In Figures 1.19 and 1.20 the most active output neuron of the trained network is plotted for horizontal bars in every position and in the second plot for vertical bars in every position. Every one of the ten output neurons responds primarily to one coherent area in one orientation. The values of the learned prior weights w_{kl} were plotted in Figures 1.21 and 1.22. In these figures, in combination with Figures 1.19 and 1.20, can be seen that each prior neuron specialized on one orientation.

During the validation of all possible horizontal bar images the output spike activity was recorded and how many distinct output neurons were spiking during each image presentation period. For the parts where the 7 pixels high bar in the image did not overlap the areas of two output neurons only one output neuron was active. Only in the border areas there were two output neurons active. This can be seen in Figures 1.23 and 1.24. Compared to experiment 1 the activity is more homogeneous as each bar can only be

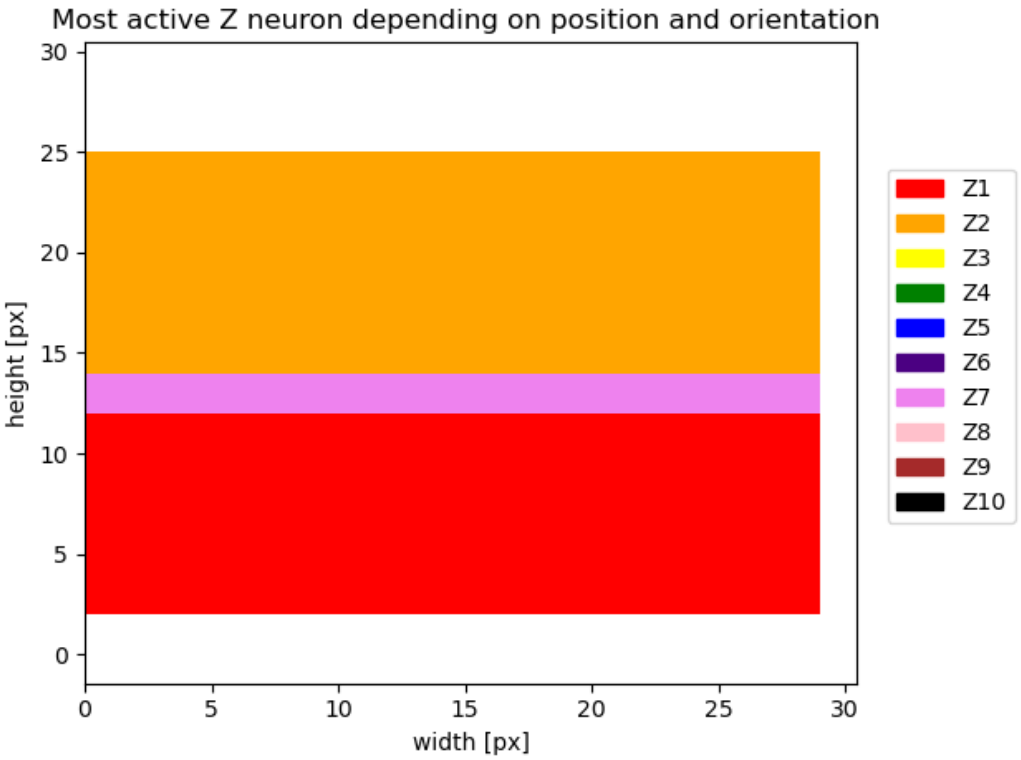

Figure 1.16: Most active output neuron for horizontal orientation and position on the y-axis of the training image during the training process. $c = 20, \lambda = 10^{-3}, Z_{factor} = 20$

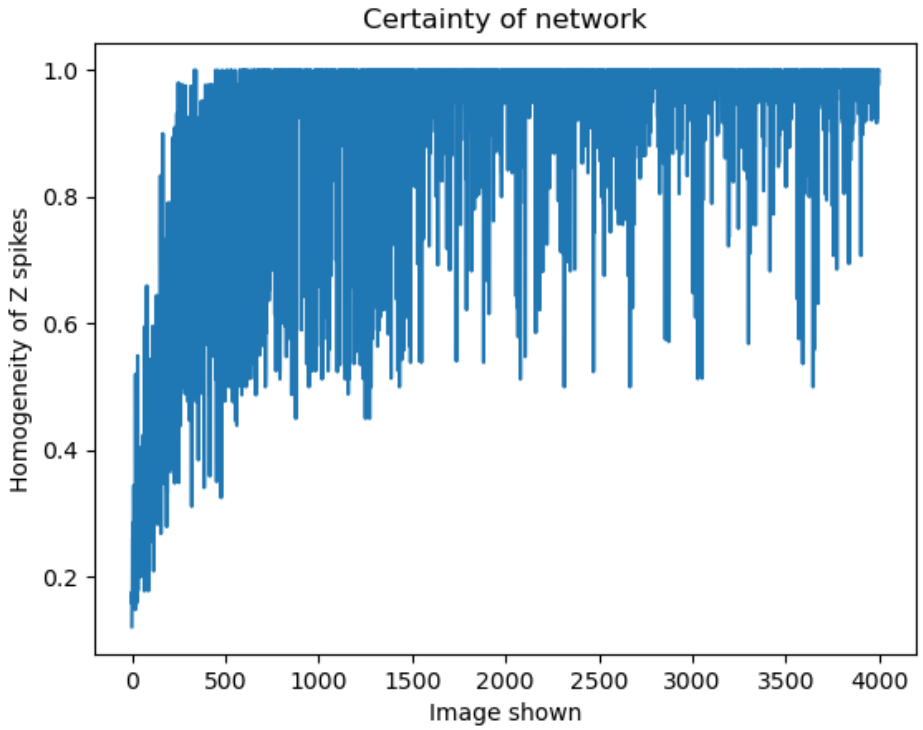

Figure 1.17: Proportion (accuracy) of most active output neuron to activity of all other output neurons during the presentation duration of each training image. $c = 20, \lambda = 10^{-3}, Z_{factor} = 5$

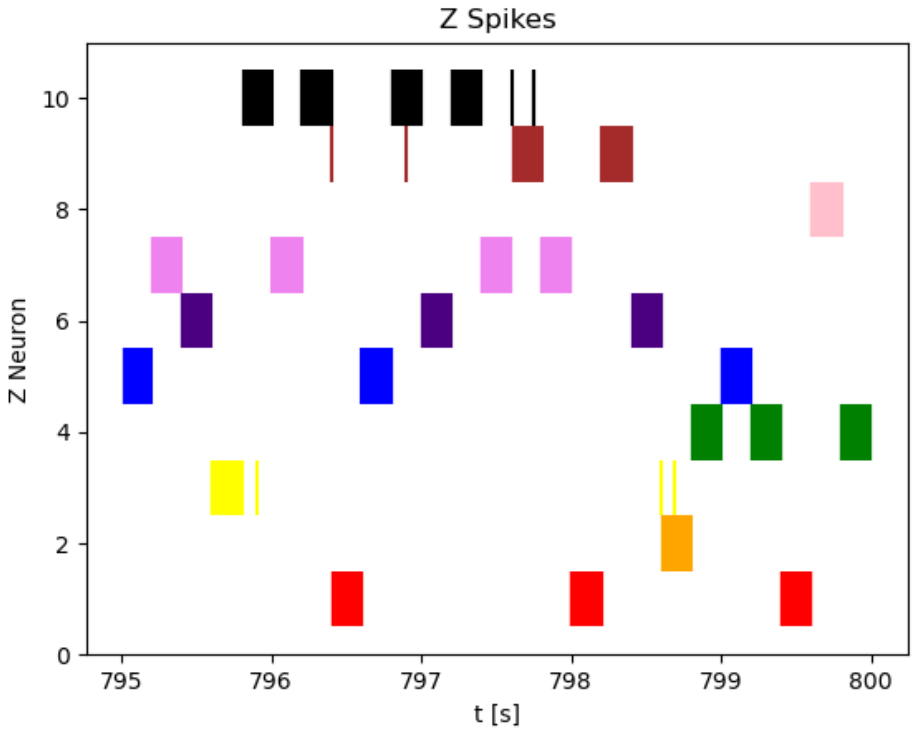

Figure 1.18: Last 1000 output neuron spikes, $c = 20, \lambda = 10^{-3}$, $Z_{factor} = 5$

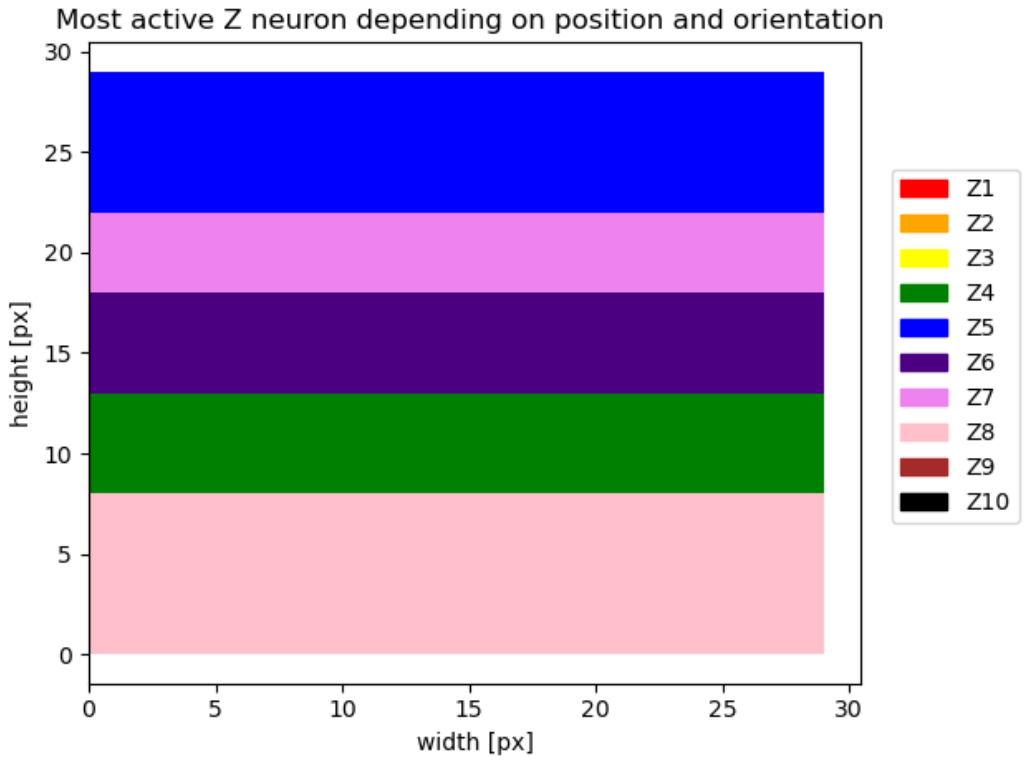

Figure 1.19: Most active output neuron for images of horizontal orientation and position on the x-axis of during the validation process. $c = 20, \lambda = 10^{-3}, Z_{factor} = 5$

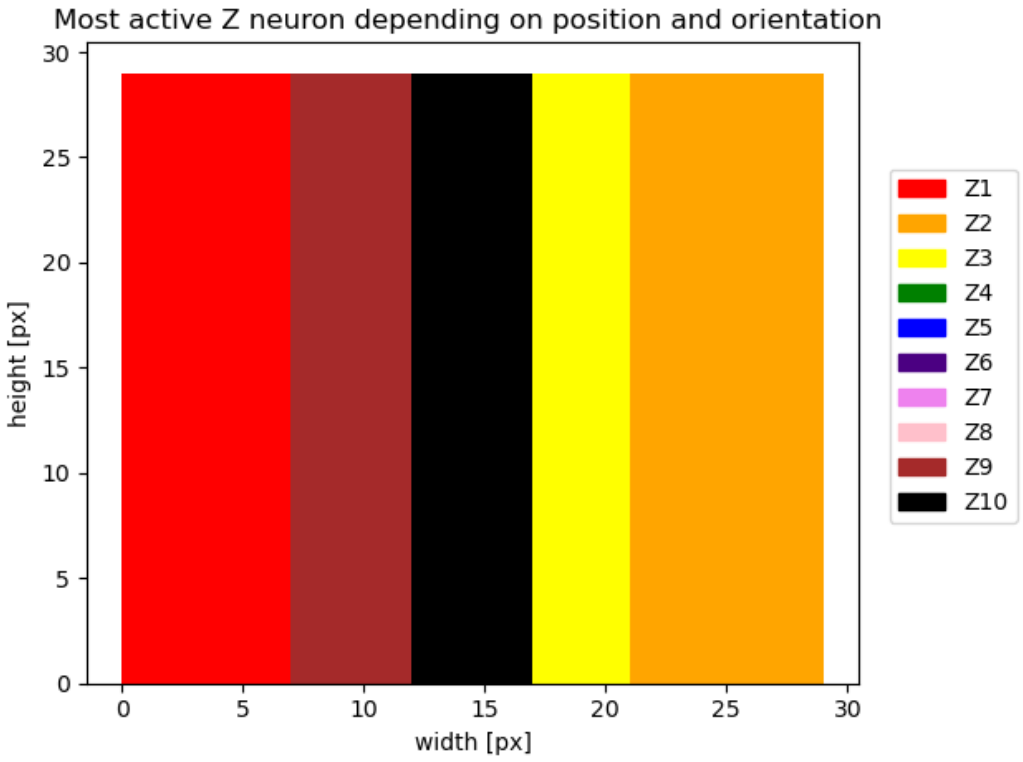

Figure 1.20: Most active output neuron for images of vertical orientation and position on the x-axis of during the validation process. $c = 20, \lambda = 10^{-3}, Z_{factor} = 5$

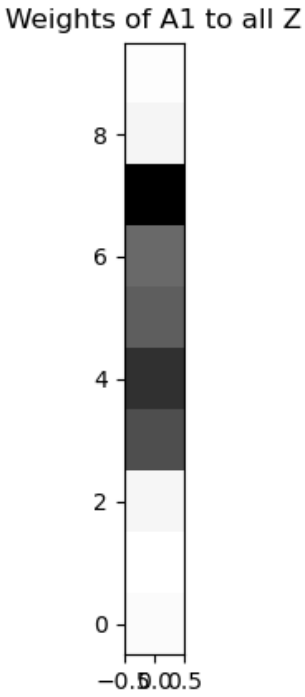

Figure 1.21: Values of w_k1, darker color means higher value. $c = 20, \lambda = 10^{-3}, Z_{factor} = 5$

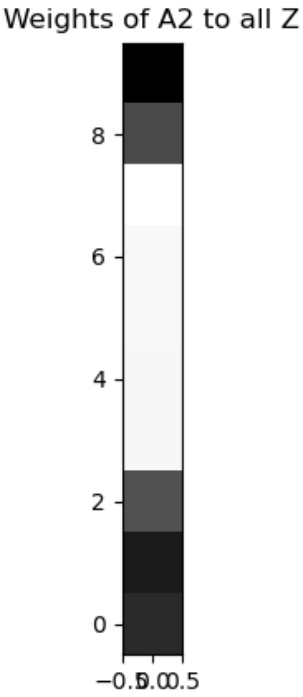

Figure 1.22: Values of $w_k 2$, darker color means higher value. $c = 20, \lambda = 10^{-3}, Z_{factor} = 5$

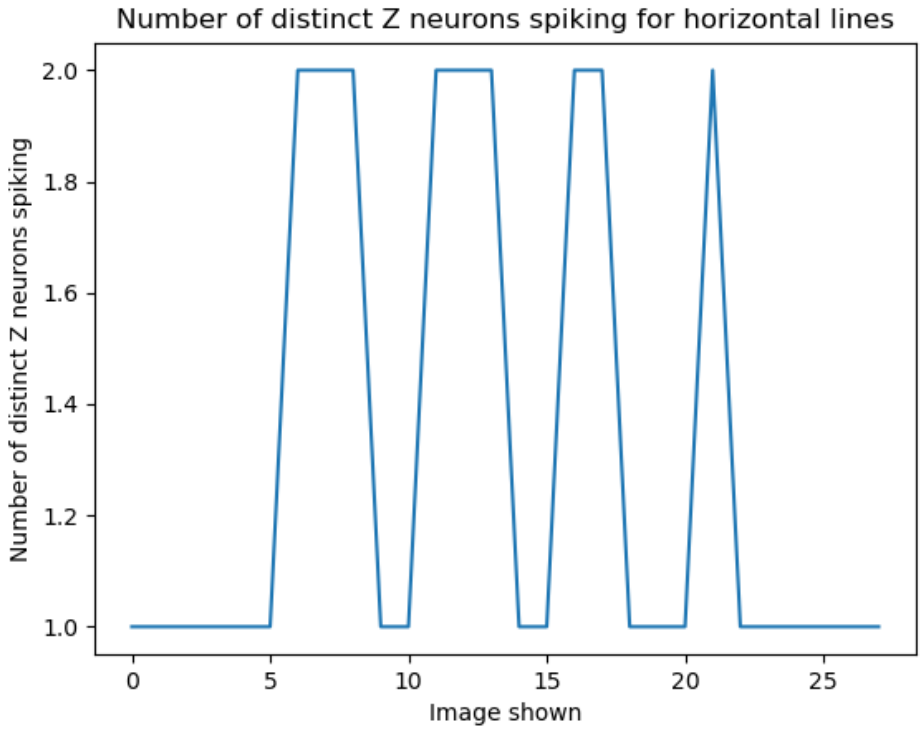

Figure 1.23: Distinct output neurons spiking during the presentation of all possible horizontal oriented validation images, $c = 20, \lambda = 10^{-3}$, $Z_{factor} = 5$

in at most the area of four output neurons, when two of these areas are reinforced by the prior neurons.

To show the impact of the prior neurons an image with two bars on it forming a cross (Figure 1.25) was generated and z_2 was set active indicating that the orientation is supposed to be horizontal. This resulted in the spiking pattern seen in Figure 1.26. In it Y_1 is more active than Y_9 due to the influence of the prior neuron Z_2.

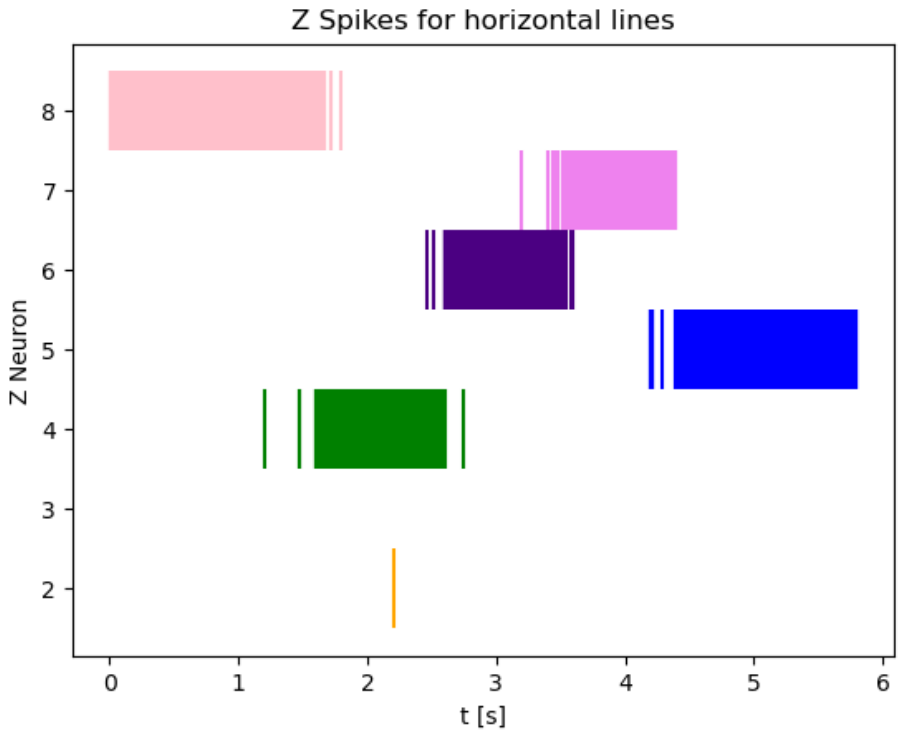

Figure 1.24: Output neuron spikes during the presentation of all possible horizontal oriented validation images, $c = 20, \lambda = 10^{-3}$, $Z_{factor} = 5$

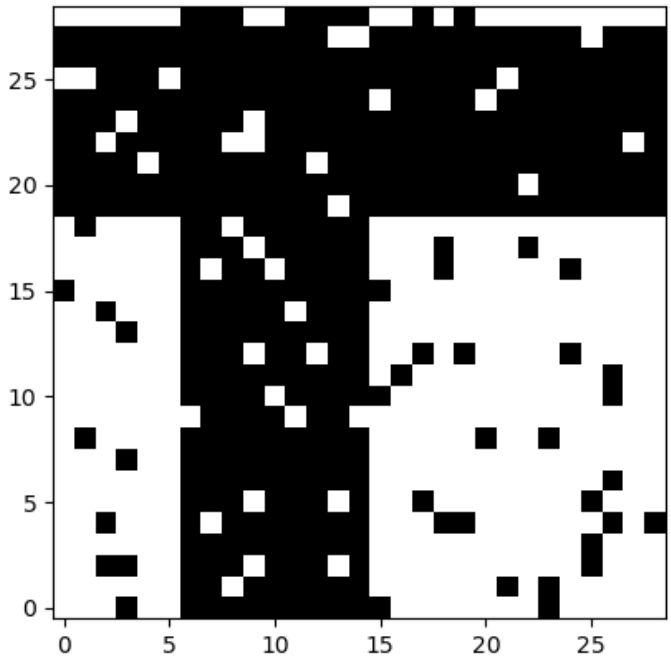

Figure 1.25: Generated validation cross image, with orientation defined as horizontal. $c = 20, \lambda = 10^{-3}, Z_{factor} = 5$

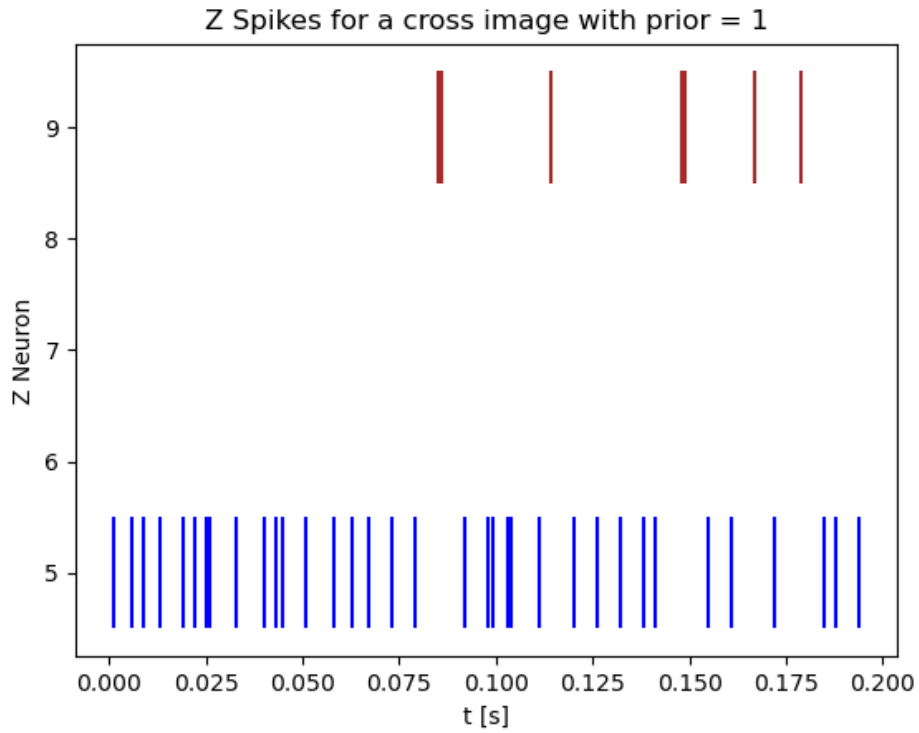

Figure 1.26: Output spikes during the presentation of the validation cross image. $c = 20, \lambda = 10^{-3}, Z_{factor} = 5$

1.3 Experiment 3: Rotated bars with adaptive inhibition

1.3.1 Introduction

This experiment analyses a different approach to the implementation of the inhibition of the output neurons. In Experiment 1.1 and 1.2 the output neurons were stopped from firing for a defined time window after any output neuron fired. The time window during which the inhibition was active was 5 ms, which resulted in an average output firing rate of 200 Hz. In this experiment an adaptive inhibition will be used which will regulate the membrane potentials of the output neurons so that all of them together fire with 200 Hz on average. The distinction to the previous experiments is that there never is a time window in which no output neuron may fire. Also it is assumed that the weight shifting parameter c will not be needed to be fitted to the network, as the adaptive inhibition regulates the output firing rate of the network.

1.3.2 Methods

The adaptive inhibition is given by Equation 1.9 which was already used in Experiments 1.1 and 1.2. However the total output firing rate $R(t)$ was set to 200 Hz in this experiment.

1.3.3 Results

Several different values for the parameter c were tested. As expected the firing rate of the output neurons 200 Hz on average regardless of the value of c. However for values of c bigger than 100 the network did not learn correctly. For those values some output neurons learned to respond to areas of more than 18°, while other neurons did not respond to any areas. For $c = 20$ the network performed equally to Experiment 1.1. The results for that parameter can be seen in Figures 1.27, 1.28 and 1.29.

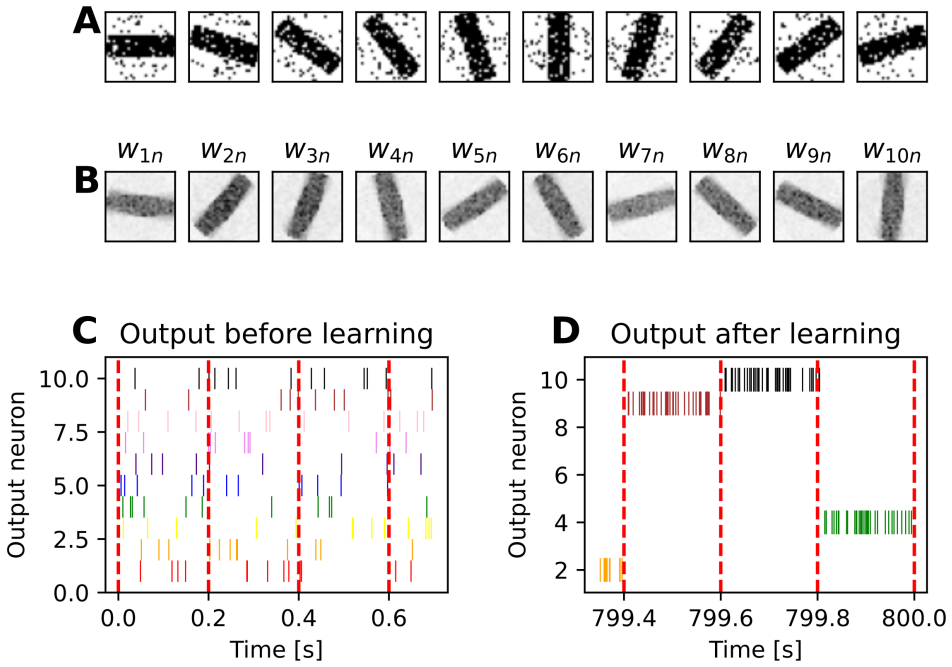

Figure 1.27: **Training. A** Examples of 29 x 29-pixel input images of rotated bars and background noise. **B** Learned weights of the connections between input and output neurons. **C, D** Spike activity expressed by the output neurons before and after the training of the network.

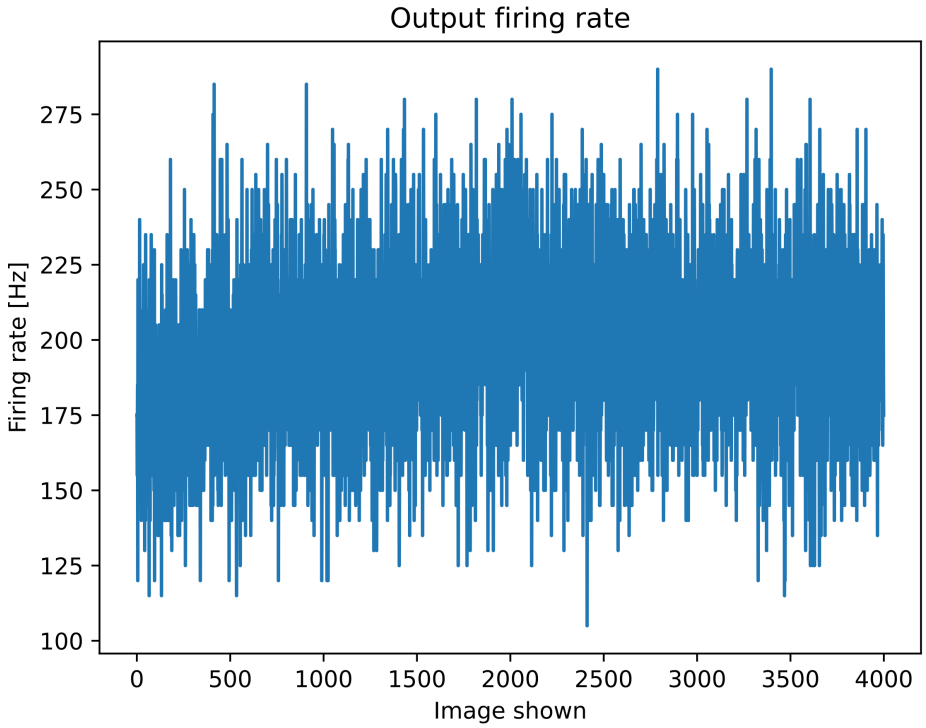

Figure 1.28: Firing rate of all output neurons combined over the training process.

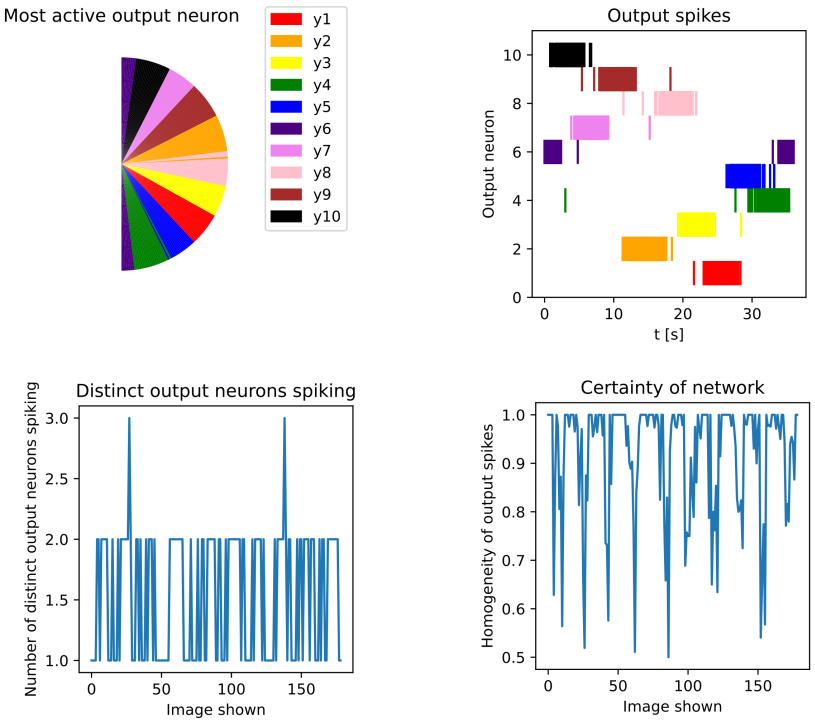

Figure 1.29: **Validation. A** Most active output neuron depending on orientation of the training image. Training images were shown for 200 ms of each possible orientation in 1° steps. **B** Spike activity expressed by the output neurons during the validation process described in (**B**). **C** Number of distinct output neurons active during the presentation of each validation image. **D** Proportion (accuracy) of most active output neuron to activity of all other output neurons during the presentation duration of each training image.

1.4 Experiment 4: Rotated bars with adaptive inhibition

1.4.1 Introduction

This experiment applies the adaptive inhibition already analysed in Experiment 1.3 to Experiment 1.2. In Experiment 1.2 the impact of the prior neurons was too small to influence the spiking activity of the output neurons. The newly implemented adaptive inhibition simplifies the process of changing the prior neurons, as it keeps the output firing frequency constant, thus preventing the need to perform a new parameter search upon each change to the prior neurons.

1.4.2 Methods

As in Experiment 1.3 the firing rate of the output neurons was set to 200 Hz. To increase the influence of the prior neurons the firing rate of the prior neurons was increased from 50 Hz to 200 Hz. This change alone this not increase the impact of the prior neurons enough to help the network correctly detect a "cross-bar image" as either horizontal or vertical. It was also tried to increase the parameter c_{prior} for the learning of the prior weights to shift the weight values towards bigger values. This did not increase the values of the weights, they stayed at a maximum of four. Thus the number of prior neurons had to be increased. Via grid search different numbers of prior neurons were tried.

1.4.3 Results

The tried parameters were as follows:

- $c = c_{prior} = 20$
- $\lambda = 10^{-3}$
- number of prior neurons $= 10, 20, 50, 100, 200$

For 50, 100 and 200 prior neurons the training process was impaired by the activity of the prior neurons. This arose as some output neurons responding to too large areas, while other prior neurons not responding to any specific areas. For 50 prior neurons four output neurons responded to horizontal bars, while six output neurons responded to vertical bars. This was unexpected and might be due to the stochastic nature of the training image generation. This was not yet analysed further and the validation of the network was performed with 20 prior neurons, as it was the largest amount of prior neurons that resulted in a properly trained network. The results of the training process can be seen in Figures 1.30 and 1.31.

Next the impact of the prior was checked. To do this an image with one horizontal (at pixel 12) and one vertical bar (at pixel 5) on it was generated. Then the prior was set once to 0 (vertical) and once to 1 (horizontal). The image and the prior was then fed to the network. The cross image can be seen in Figure 1.32. The spiking activity depending on the prior can be seen in Figures 1.33 and 1.34.

To better show the impact of the prior it was gradually changed from horizontal to vertical. The starting firing frequency of the horizontal prior neurons was 200 Hz and 0 Hz for the vertical prior neurons. A cross image was shown to the network for 200 ms. After each image presentation duration the firing frequency of the horizontal prior neurons was decreased by 1 Hz and increased by 1 Hz for the vertical prior neurons. The used cross image can be seen in Figure 1.35. The firing frequency of the 2 most active output neurons depending on the firing frequency of the vertical prior neurons can be seen in Figure 1.36. As expected the correct horizontal output neuron is active in the beginning with about 200 Hz and the correct vertical output neuron is almost inactive. With rising firing frequency of the vertical prior neurons the activity of the horizontal output neurons decreases and the activity of the vertical output neurons increases. The membrane potentials of all output neurons for vertical prior neurons spiking with 200 Hz and horizontal prior neurons spiking with 0 Hz can be seen in Figure 1.37.

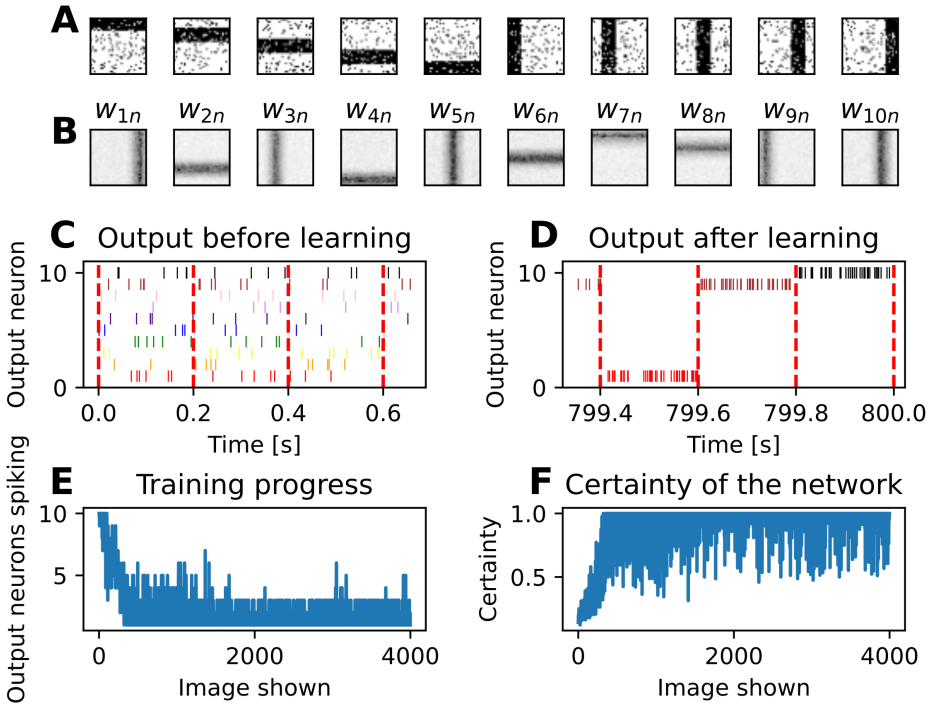

Figure 1.30: **Training with 20 prior neurons. A** Examples of 35 x 35-pixel input images of horizontal and vertical bars with background noise. **B** Learned weights of the connections between input and output neurons. **C, D** Spike activity expressed by the output neurons before and after the training of the network. **E** Number of distinct output neurons active during the presentation duration of each training image. **F** Proportion of most active output neuron to activity of all other output neurons during the presentation duration of each training image.

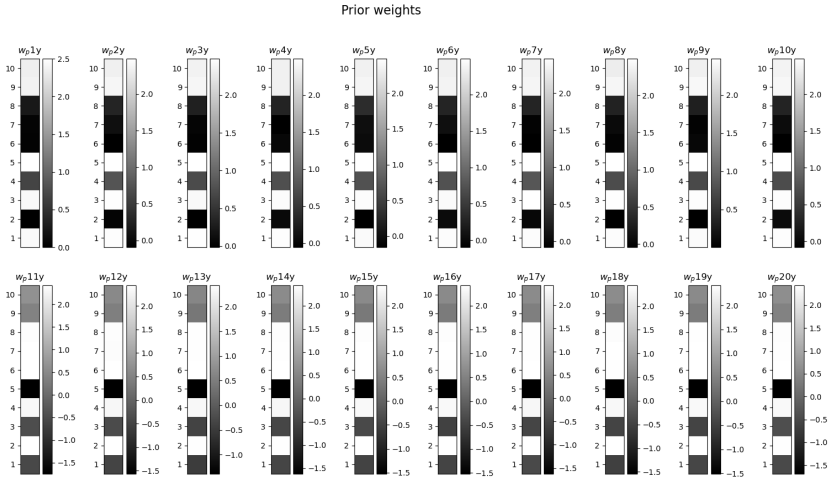

Figure 1.31: Learned weights of the connections between prior and output neurons.

1.5 Experiment 4: Mathematical analysis and simulation of a 1D network

1.5.1 Introduction

The purpose of this experiment was to mathematically analyse a simpler network, thus providing a definitive way to verify the performance of the simulation. The network was scaled down to be one-dimensional with 18 input neurons, four prior neurons and four output neurons, making it easier to analyse. All weights were calculated by hand and then used to determine the theoretical posterior probability of the network. The calculated weights were also given to the simulation and the distribution of the output spikes was used to calculate the posterior probability of the simulation. The posterior probabilities of both the mathematical analysis and the simulation are compared and by varying three parameters of the simulation it is tuned to approximate the analytical solution as closely as possible.

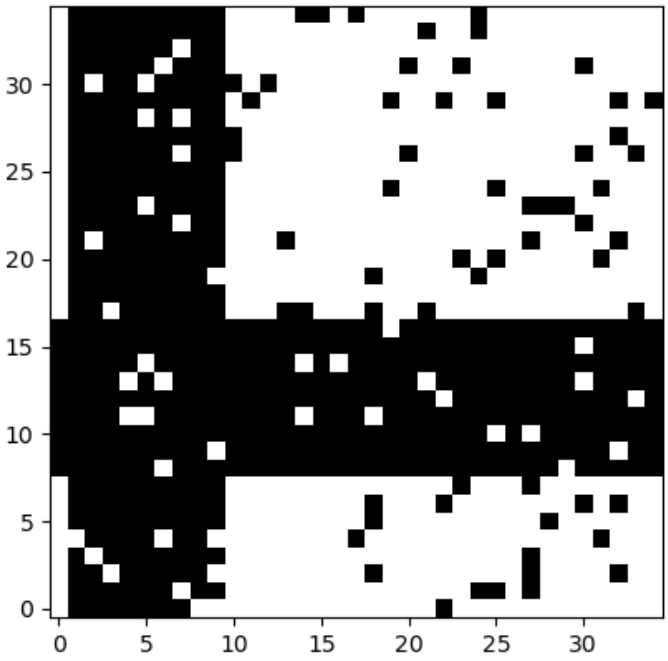

Figure 1.32: Cross image fed to the network.

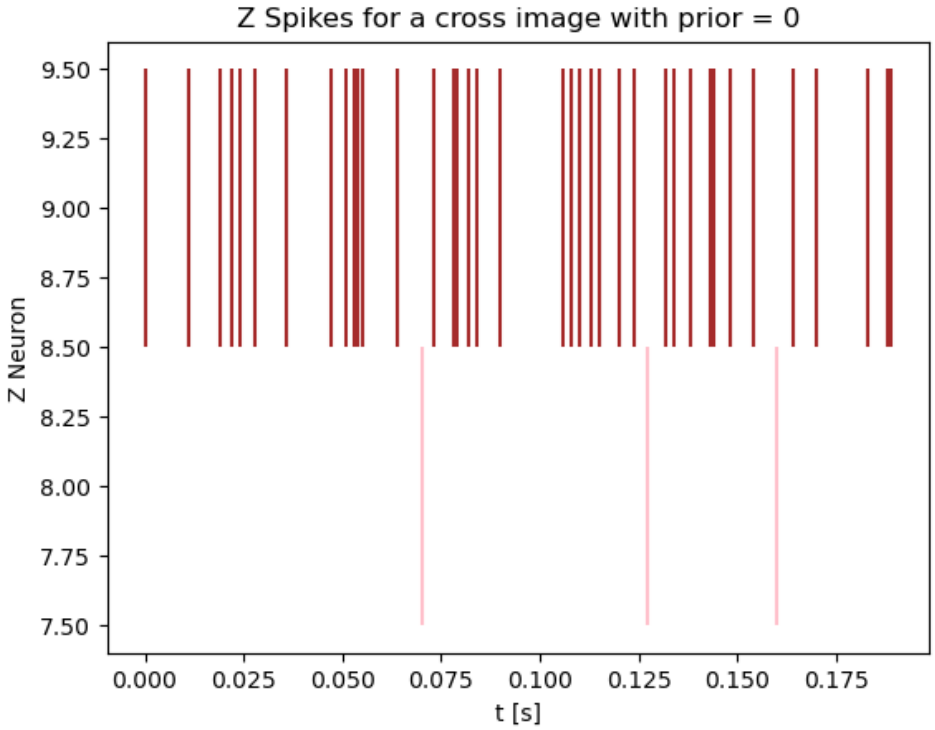

Figure 1.33: Spiking activity of the output neurons with prior = 0.

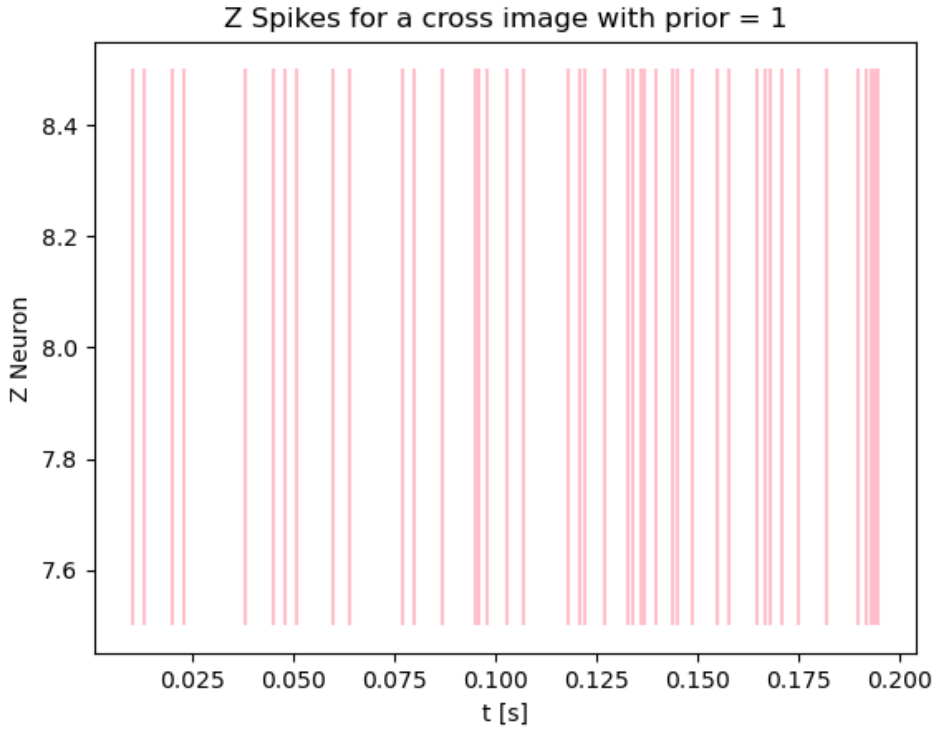

Figure 1.34: Spiking activity of the output neurons with prior = 1.

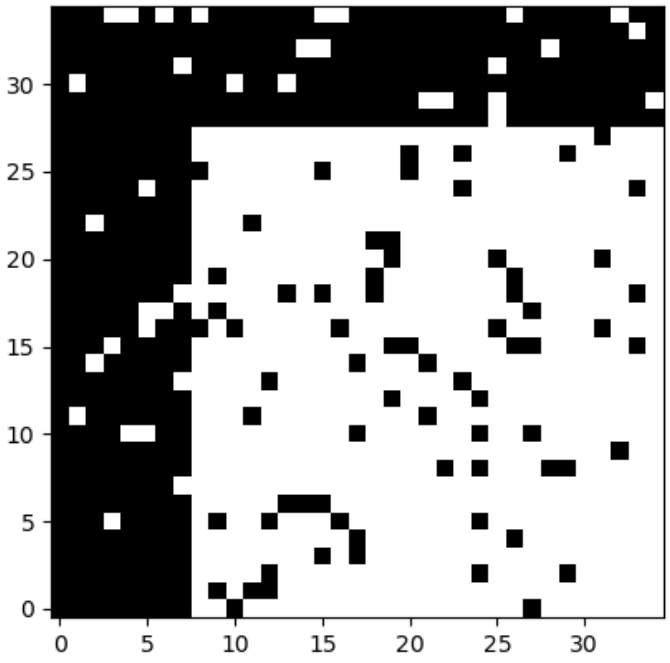

Figure 1.35: Cross image fed to the network.

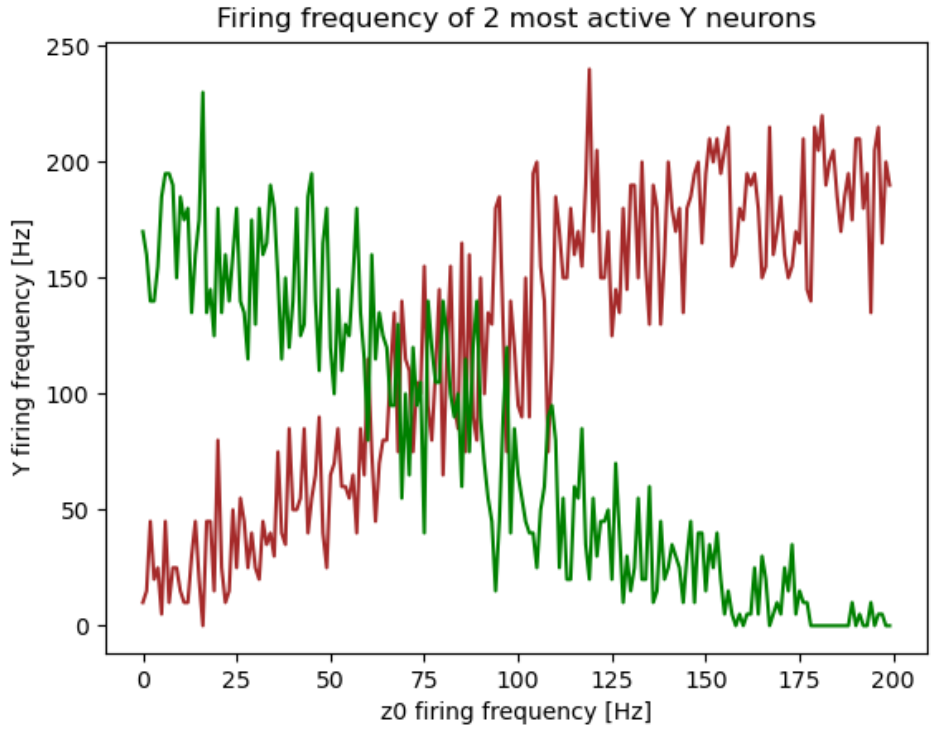

Figure 1.36: Firing frequency of the 2 most active output neurons depending on prior firing frequency.

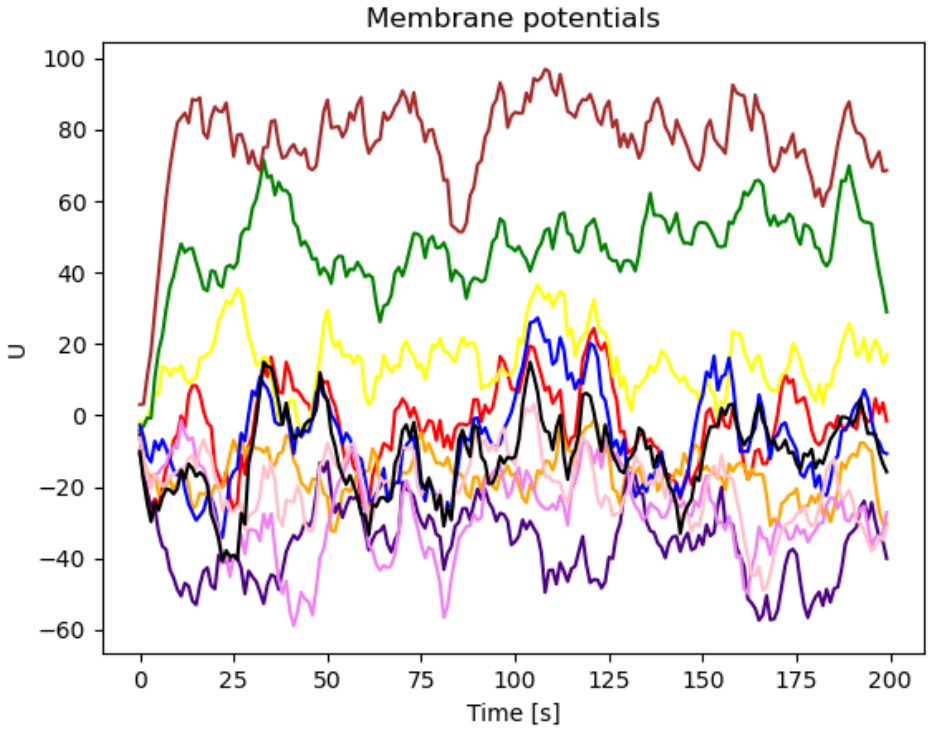

Figure 1.37: Membrane potentials of output neurons for vertical prior neurons spiking with 200 Hz and horizontal prior neurons spiking with 0 Hz.

1.5.2 Methods

Network structure The architecture of this network remained the same as in previous experiments (with adaptive inhibition and a noise level of 10%), only the amount of neurons was changed. The input image consists of nine pixels in a horizontal line. Within those nine pixels four output classes can be represented. Each output class consists of three pixels next to each other. This results in each output class overlapping its neighbour classes by one pixel. As there have to be 2 input neurons for each pixel, one neuron being active if the pixel is white and one if the pixel is black, the network has 18 input neurons. Furthermore four prior neurons were implemented, of which only one is being active for one of the output classes at a time. Lastly the network has four output neurons.

Mathematical Analysis The posterior probability $P(Y = i|X = x, Z = j)$ of the network was calculated by using

$$P(Y = i|X = x, Z = j) = \frac{P(X = x|Y = i)P(Y = i|Z = j)}{\Sigma_k P(X = x, Y = k)P(Y = k|Z = j)}. \tag{1.13}$$

$P(X = x|Y = y)$ and $P(Y = y|Z = z)$ were derived by hand corresponding to the paradigm of the experiment. For $P(X = x|Y = y)$ this was done by first writing down a matrix $P(X|Y)$ of all probabilities for each y and x. This matrix was then multiplied with the input image resulting in a probability for each output class. Next $1 - P(X|Y)$ and $1 - inputimage$ were multiplied to include the effect of the input neurons that are active for non active (white) pixels of the input image. The results of both calculations were then multiplied to yield $P(X = x|Y = y)$. $P(Y = y|Z = z)$ was calculated by taking the matrix $P(Y|Z)$ and multiplying it with the input image.

Simulation The input weights for the simulation were derived from the matrices $P(X|Y)$ and the prior weights were derived from $P(Y|Z)$ by taking the natural logarithm of each matrix. To include the effects of the input neurons representing non active input pixels, separate inverted input weights were calculated by taking the natural logarithm of $1 - P(X|Y)$. Each set of input weights was applied separately to the corresponding input neurons.

After the image presentation period of each input image the amount of the output spikes of different classes were counted and their proportions were calculated to yield the "simulation output probabilities" in later plots. The three parameters input firing rate f_{input}, prior firing rate f_{prior} and the membrane constant τ_{decay} are varied to inspect their influence on the result, as well as to approximate the analytical solution as closely as possible. Each input image is presented to the network for 20 seconds to reduce the variance between runs.

1.5.3 Results

First the matrix $P(X|Y)$ was analytically calculated

$$P(X|Y) = \begin{bmatrix} 0.9 & 0.9 & 0.9 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.9 & 0.9 & 0.9 & 0.1 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.1 & 0.9 & 0.9 & 0.9 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.9 & 0.9 & 0.9 \end{bmatrix} \tag{1.14}$$

and also the matrix

$$P(Y|Z) = \begin{bmatrix} 0.9 & 0.0333 & 0.0333 & 0.0333 \\ 0.0333 & 0.9 & 0.0333 & 0.0333 \\ 0.0333 & 0.0333 & 0.9 & 0.0333 \\ 0.0333 & 0.0333 & 0.0333 & 0.9 \end{bmatrix}. \tag{1.15}$$

These two matrices and their inverses were then multiplied with each input image as described in Methods to yield $P(Y = i|X = x, Z = j)$ also called "Analysis output probabilities" in later plots.

First a simulation with $f_{prior} = 500Hz$ and $\tau_{decay} = 15ms$ was performed with different values for f_{input} ranging from 70 Hz to 200 Hz. Out of all these variations $f_{input} = 150Hz$ yielded the best result and can be seen in Figure 1.38. In this figure six different input images can be seen marked with the letter "A". The active pixels are shown in black. The center points of the three-wide output classes are at position 1, 3, 5 and 7. The value of the prior is indicated by the red border which is three-wide and centered at the center position of the corresponding output class. The posterior probability

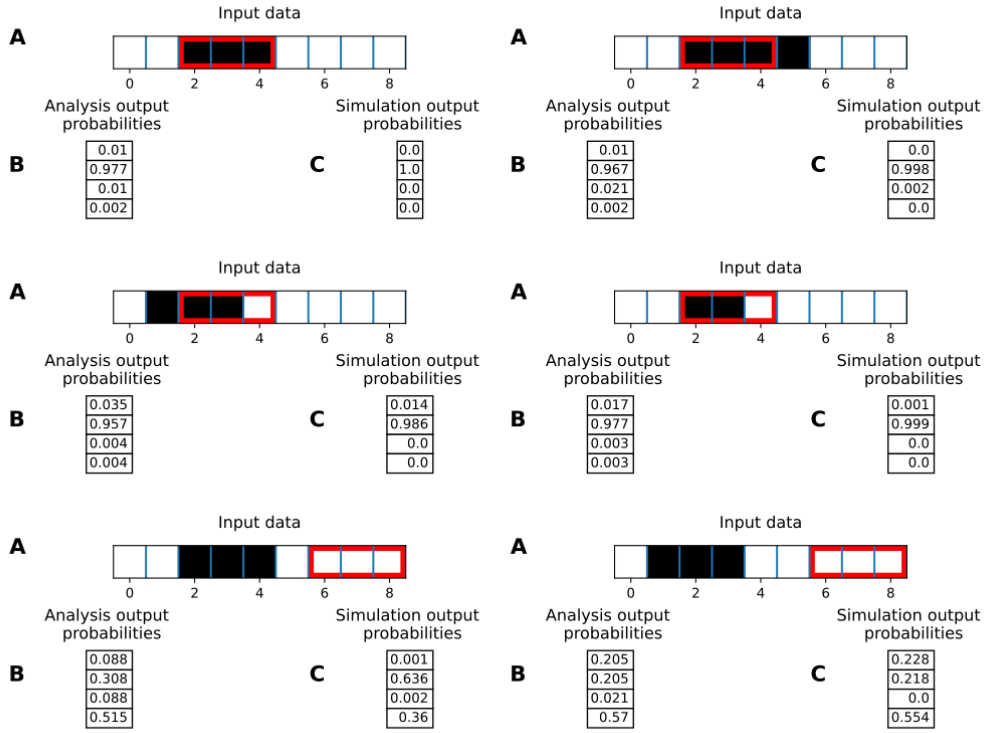

Figure 1.38: **Analysis and simulation result. Parameters:** $f_{input} = 150Hz, f_{prior} = 500Hz, \tau_{decay} = 15ms$ **A** Input images with 9 x 1 pixels. The red borders indicate the class of the prior. **B** Analytically calculated posterior probabilities. **C** Proportions of the spikes of the output neurons during the simulation.

of the mathematical analysis can be seen next to the letter "B". And finally the proportion of output spikes of the different output neurons can be seen next to the letter "C".

When f_{input} was 70 Hz the results differed more as can be seen in Figure 1.39.

As the simulation result with these parameters yielded too low probabilities for the areas adjacent to the active areas of the input images, smaller values for τ_{decay} were tried. The results for $\tau_{decay} = 4ms$ were the most promising. After simulating values for f_{input} between 70 Hz and 150 Hz in increasing

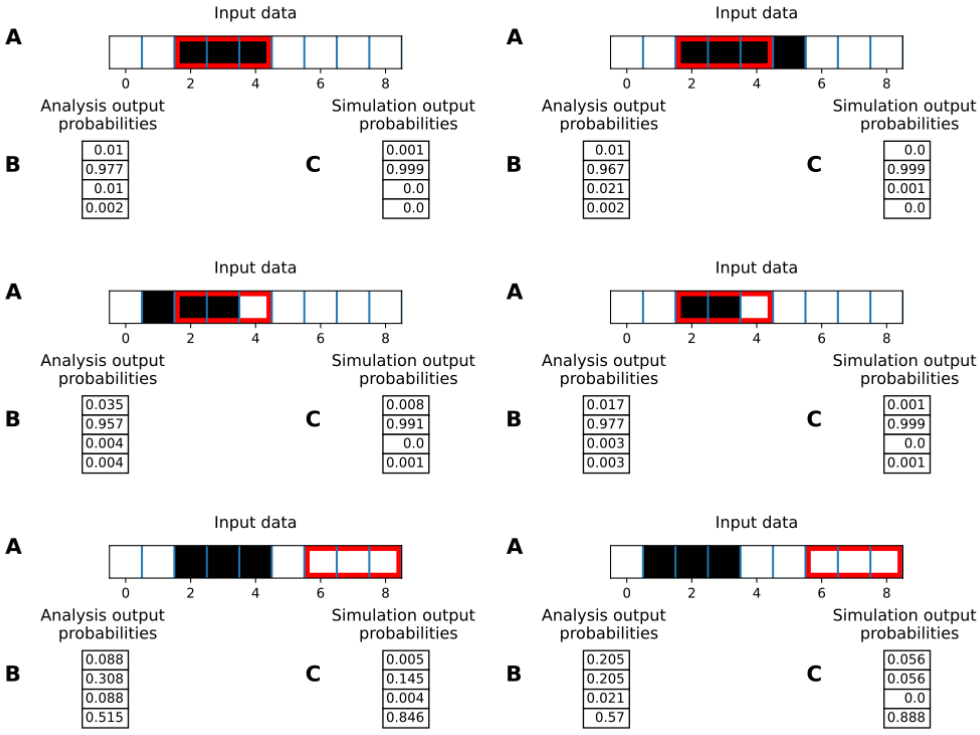

Figure 1.39: **Analysis and simulation result. Parameters:** $f_{input} = 70Hz, f_{prior} = 500Hz, \tau_{decay} = 15ms$ **A** Input images with 9 x 1 pixels. The red borders indicate the class of the prior. **B** Analytically calculated posterior probabilities. **C** Proportions of the spikes of the output neurons during the simulation.

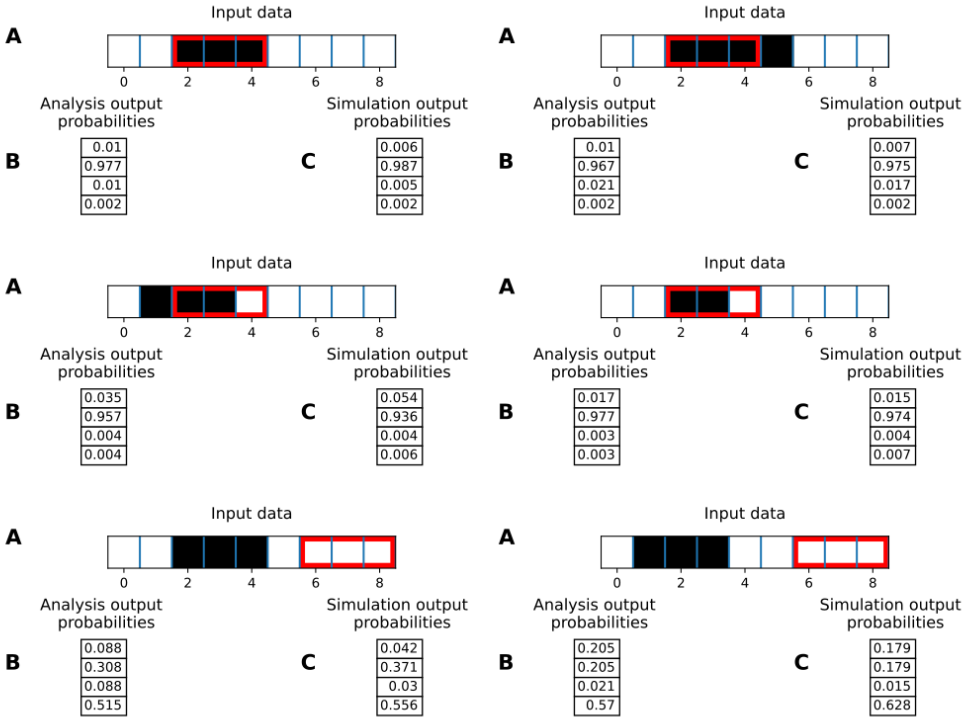

Figure 1.40: **Analysis and simulation result. Parameters:** $f_{input} = 110Hz, f_{prior} = 500Hz, \tau_{decay} = 4ms$ **A** Input images with 9 x 1 pixels. The red borders indicate the class of the prior. **B** Analytically calculated posterior probabilities. **C** Proportions of the spikes of the output neurons during the simulation.

steps of 10 Hz an input frequency of 110 Hz yielded the best result. This can be seen in Figure 1.40.

As last step of the parameter fitting process different values between 400 and 500 Hz for f_{prior} in steps of 10 Hz were tried. A prior frequency of 460 Hz yielded the best result. The plot with the final parameters can be seen in Figure 1.41.

At last f_{input} was set to 500 Hz to demonstrate its influence. The results of this last parameter configuration can be seen in Figure 1.42.

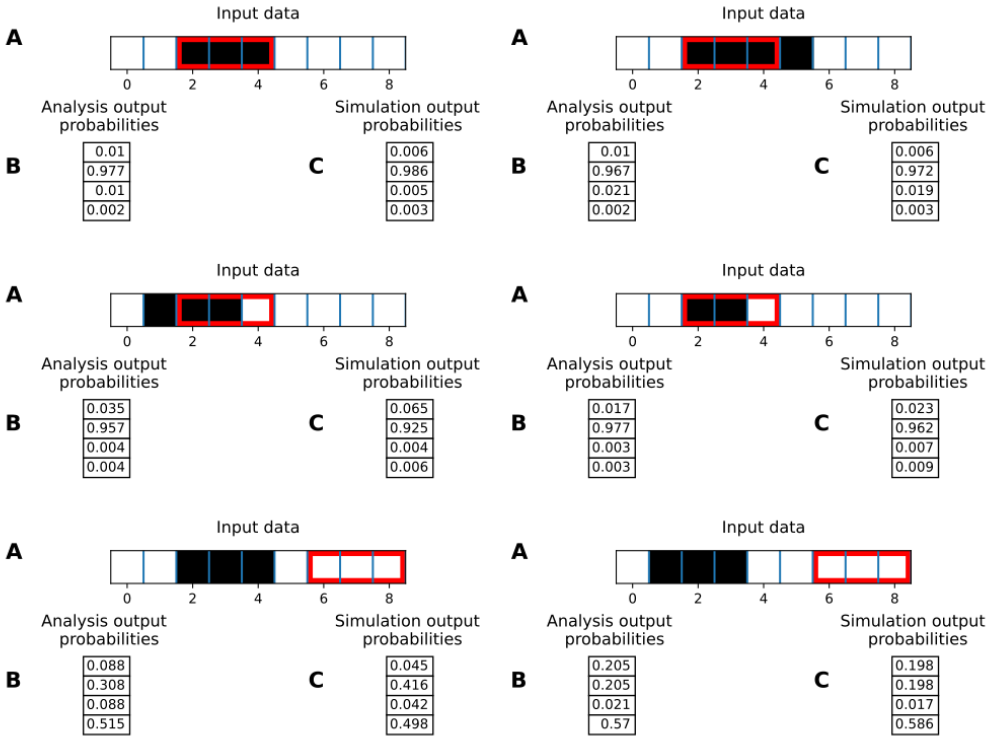

Figure 1.41: **Analysis and simulation result. Parameters:** $f_{input} = 110Hz, f_{prior} = 460Hz, \tau_{decay} = 4ms$ **A** Input images with 9 x 1 pixels. The red borders indicate the class of the prior. **B** Analytically calculated posterior probabilities. **C** Proportions of the spikes of the output neurons during the simulation.

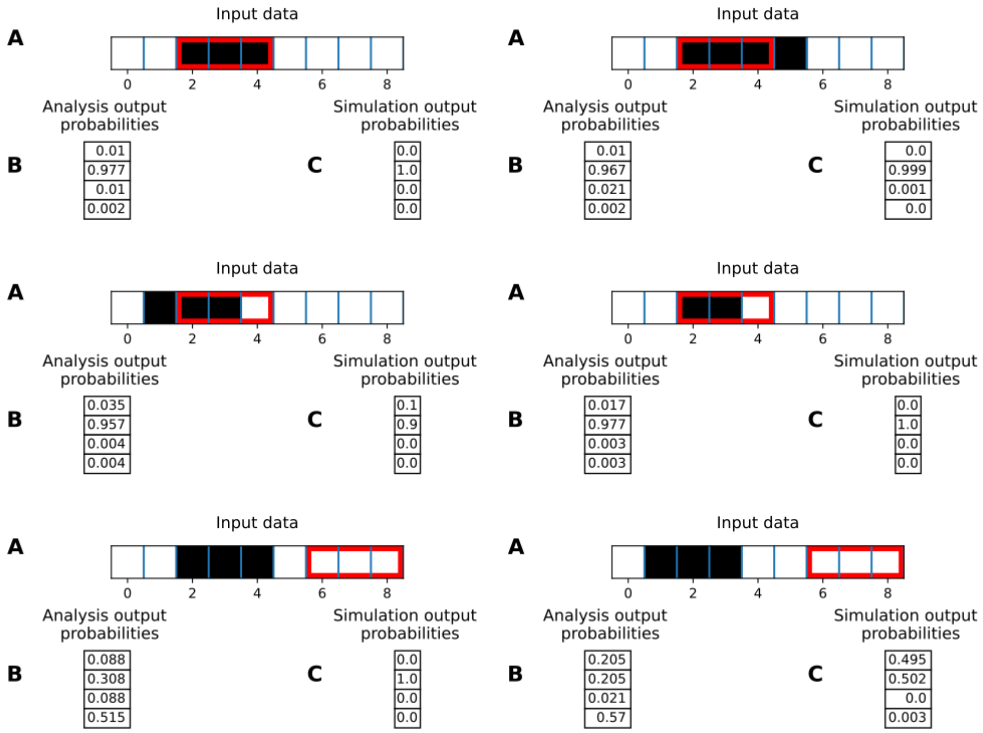

Figure 1.42: **Analysis and simulation result. Parameters:** $f_{input} = 500Hz, f_{prior} = 460Hz, \tau_{decay} = 4ms$ **A** Input images with 9 x 1 pixels. The red borders indicate the class of the prior. **B** Analytically calculated posterior probabilities. **C** Proportions of the spikes of the output neurons during the simulation.

1.5.4 Discussion

In this experiment the impact of the three network parameters f_{input}, f_{prior} and τ_{decay} was analysed.

f_{input} controls how strongly the information of the pixels of the input image is weighed. This means that by raising the impact of the active pixels increases, while the impact of the prior neuron decreases comparatively. However this is not the only impact this parameter has. When raising f_{input} it was also observed that the probabilities for output classes adjacent to the active pixels decreased. This can be observed when comparing Figures 1.41 and 1.42 where f_{input} was raised from 110 Hz to 500 Hz. For example when looking at the input image in the middle row on the right it can be seen that for $f_{input} = 110Hz$ the simulation output probability for class 1 is 0.023. This greater than zero probability is mostly due to the active pixel number 2, which belongs to class 1 and 2 at the same time. When now increasing f_{input} to 500 Hz one might expect the probabilities for class 1 and 2 to rise. This however does not happen, instead the simulation output probability for class 1 falls to zero, while for class 2 it rises. It is assumed that this happens because the membrane potentials of the output neurons never are normalized. As the input neurons spike more quickly the membrane potential of output neuron 1 rises slower than the membrane potential of output neuron 2. This leads to shifted firing rates in favour of output neuron 2. However this behaviour might be expected, as faster spiking input neurons correspond statistically to taking more samples of a probability distribution. This means that the more samples the network takes, the more certain it becomes in the more likely option, which is output class 2.

f_{prior} increases the impact of the prior neurons as it is increased. It could best be fitted by looking at input images where all active pixels are outside of the area that the prior indicates and then adjusting the prior firing rate until the simulation output probability approximated the analysis output probability for the output class of the prior.

τ_{decay} determines for how long and how strongly an input or prior spike contributes to the membrane potentials of the output neurons. With its initial value of 15 ms the input neurons were keeping the probabilities of classes adjacent to the class with active pixels, e.g. Figure 1.38 top right input image class 3, too low. When lowering f_{input} the probability of class 3 began to rise. However before the probability reached the required value f_{input} was becoming too small. Thus τ_{decay} was reduced to enable f_{input} to be increased again.

Approximating the analytic solution Although the simulation was able to approximate the analytical solution for single input images to a very high degree, it was impossible to approximate every single one of the six chosen input images to a high degree with the same set of parameters. In the bottom two input images of Figure 1.41 for the left image the prior firing frequency seems too low, while for the right image it seems too high. Furthermore f_{input} seems to be chosen well for the top right image, although being seemingly too big. On the other hand in the middle left image f_{input} seems to be too low, as the probability for class 1 is too low. Additionally there is a significant variance of the simulation output probabilities, even though each input image was presented for 20 seconds, resulting in 4000 output spikes. This was observed by viewing the results of the different values for f_{input} which were increased by 10 Hz each iteration. For some iterations the output probabilities changed in the counter-intuitive way, presumably due to stochastic variance. If necessary the final result could be improved by increasing the image presentation duration significantly.

Bibliography

Nessler, Bernhard et al. (Apr. 2013). "Bayesian Computation Emerges in Generic Cortical Microcircuits through Spike-Timing-Dependent Plasticity." In: *PLOS Computational Biology* 9.4, pp. 1–30. DOI: 10.1371/journal.pcbi.1003037. URL: https://doi.org/10.1371/journal.pcbi.1003037 (cit. on pp. 1–5).