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 lines

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 lines 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 lines 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 lines 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.

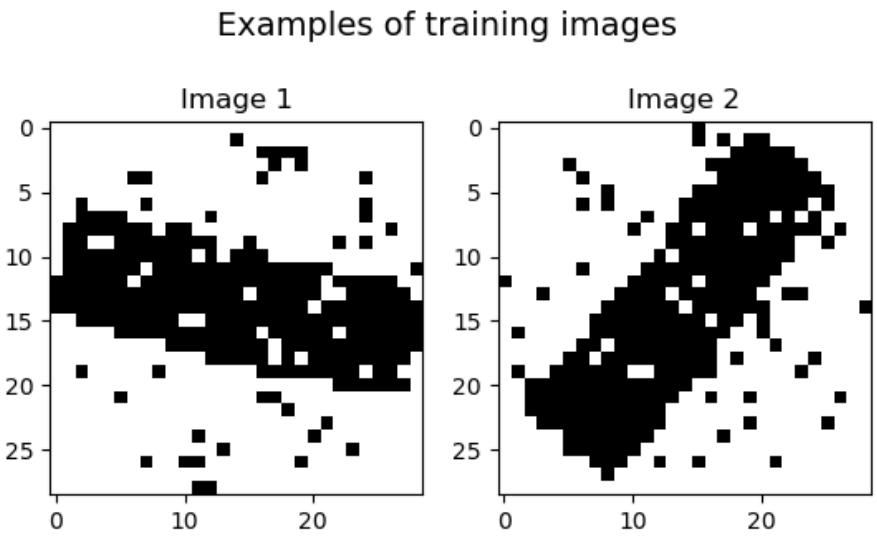

Figure 1.1: Two examples of the generated training data in this experiment.

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

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

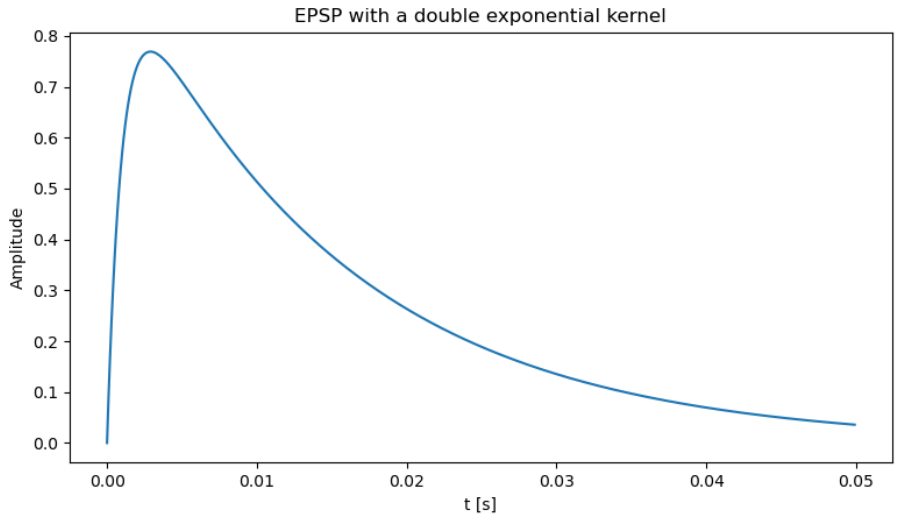

Figure 1.2: Form of an excitatory post synaptic potential generated by an input neuron over time. A double exponential kernel was used to generate this signal. These signals are fed to the next layer of the network.

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 lines in a coherent area. Each of those areas should ideally be of equal size. As the lines 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

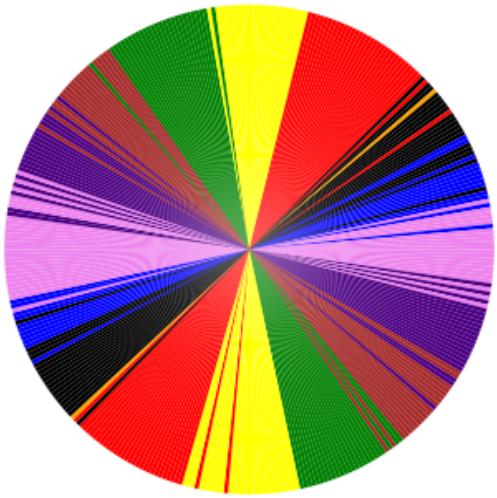

Figure 1.3: Most active output neuron depending on orientation of the training image during the training process. $c = 1, \lambda = 10^{-2}$

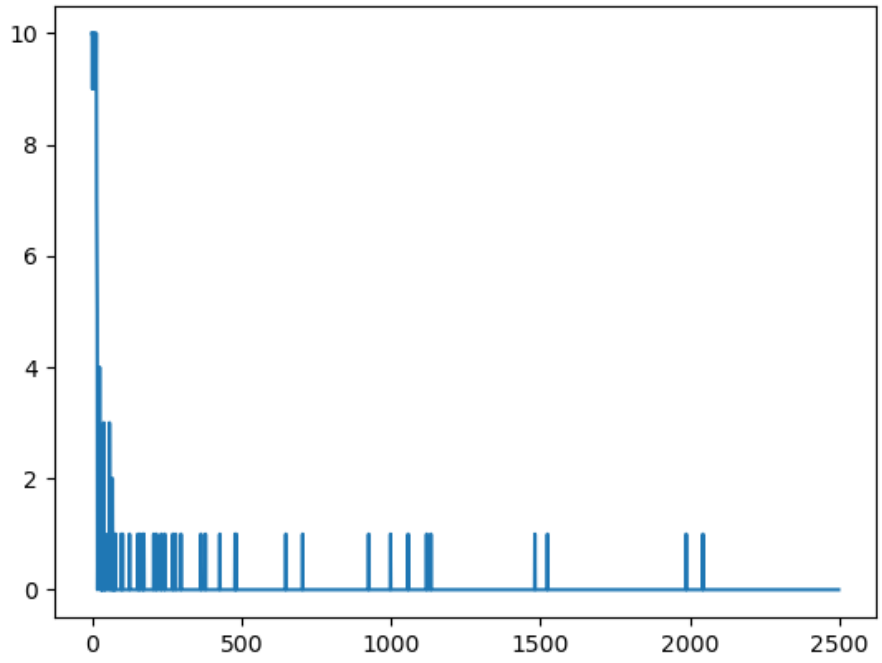

Figure 1.4: Number of distinct output neurons active during the presentation duration of each training image. x-axis: image shown, y-axis: distinct output neurons active, $c = 1, \lambda = 10^{-2}$

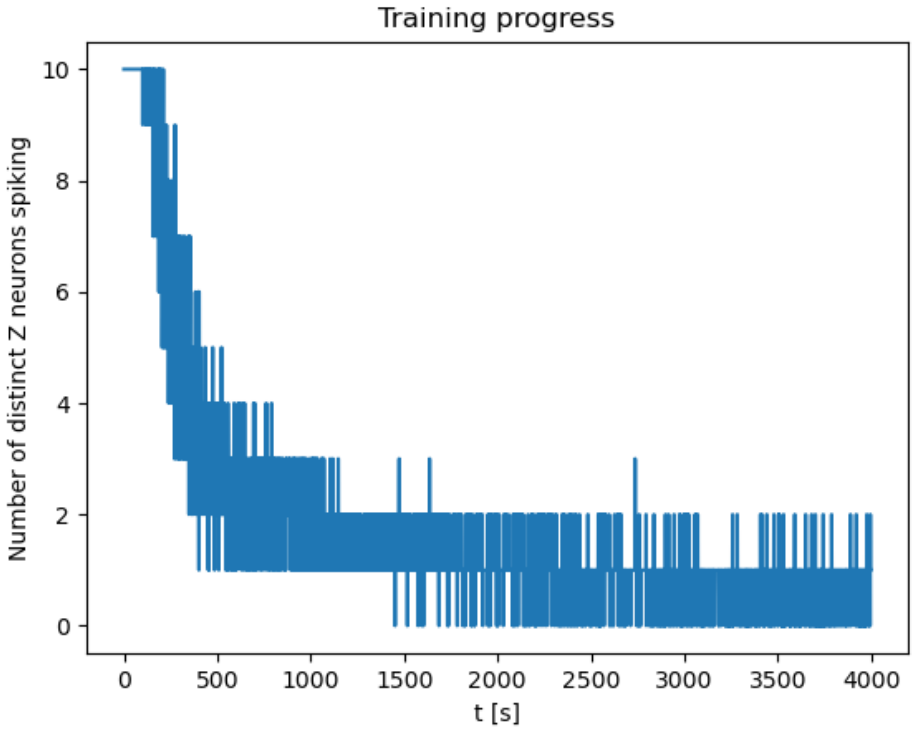

Figure 1.5: Number of distinct output neurons active during the presentation duration of each training image. x-axis: image shown, y-axis: distinct output neurons active, $c = 10, \lambda = 10^{-3}$

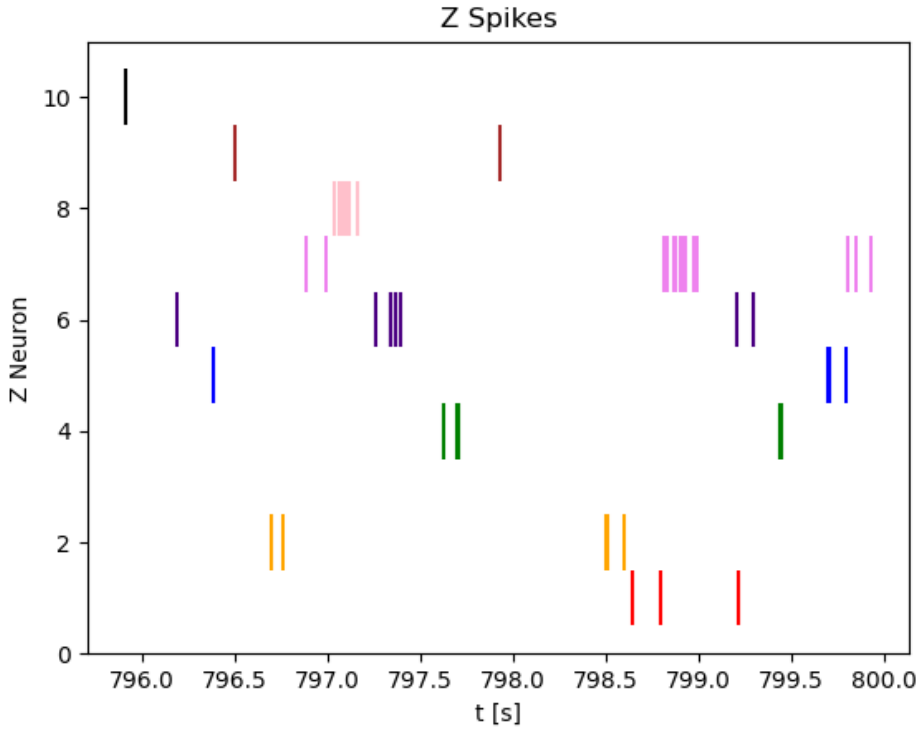

Figure 1.6: Last 50 output neuron spikes, $c = 10, \lambda = 10^{-3}$

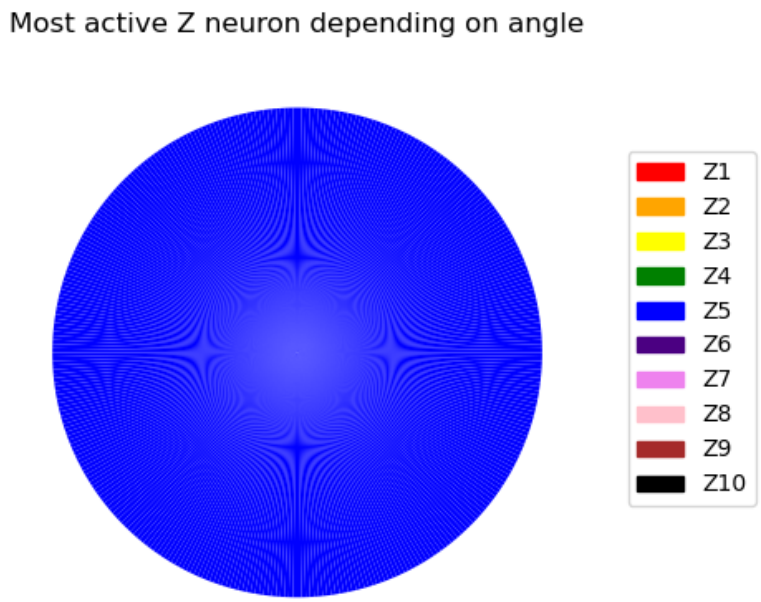

Figure 1.7: Most active output neuron depending on orientation of the training image during the training process. $c = 20, \lambda = 10^{-2}$

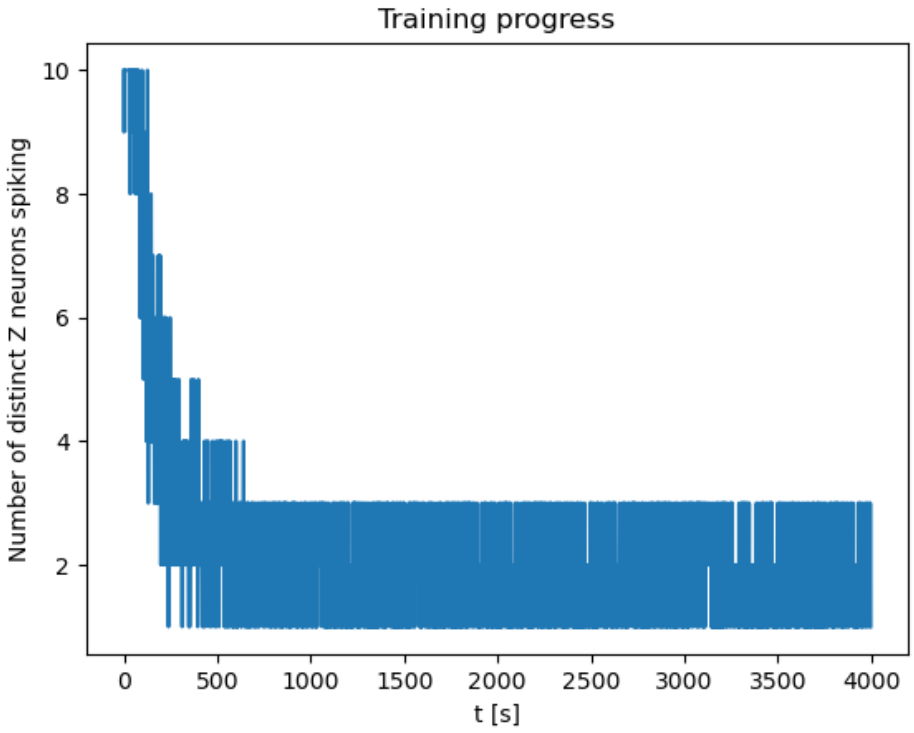

Figure 1.8: Number of distinct output neurons active during the presentation duration of each training image. x-axis: image shown, y-axis: distinct output neuron active, $c = 20, \lambda = 10^{-3}$

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.

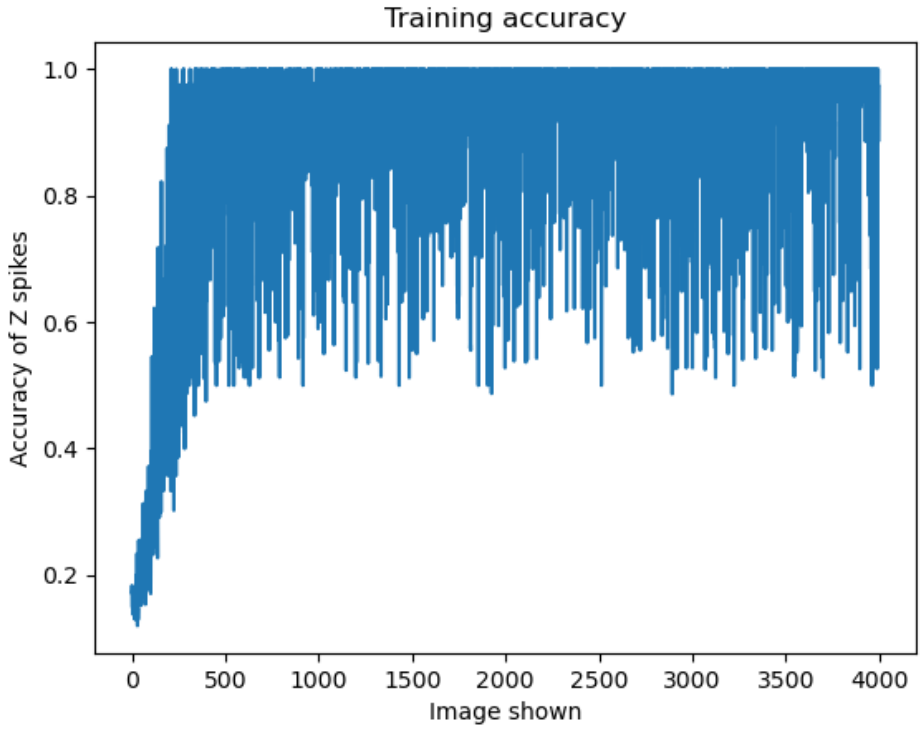

Figure 1.9: Proportion (accuracy) of most active output neuron to activity of all other output neurons during the presentation duration of each training image. $c = 20, \lambda = 10^{-3}$

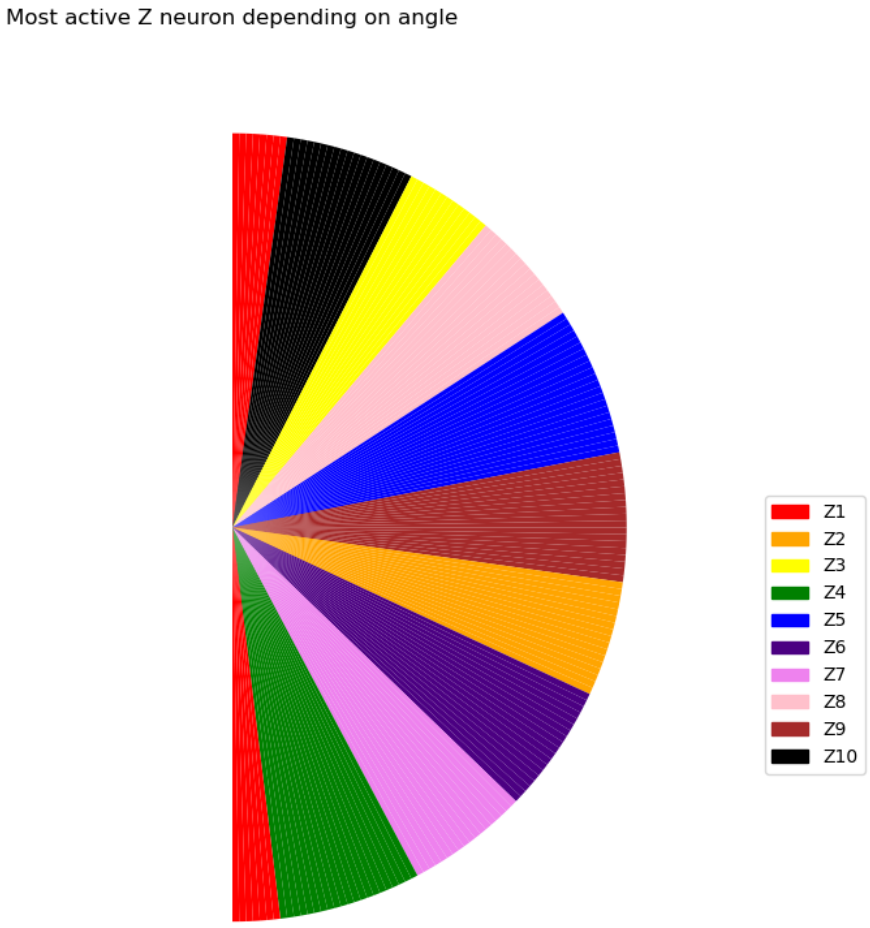

Figure 1.10: Most active output neuron depending on orientation of the training image during the training process. $c = 20, \lambda = 10^{-3}$

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 line 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 lines 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

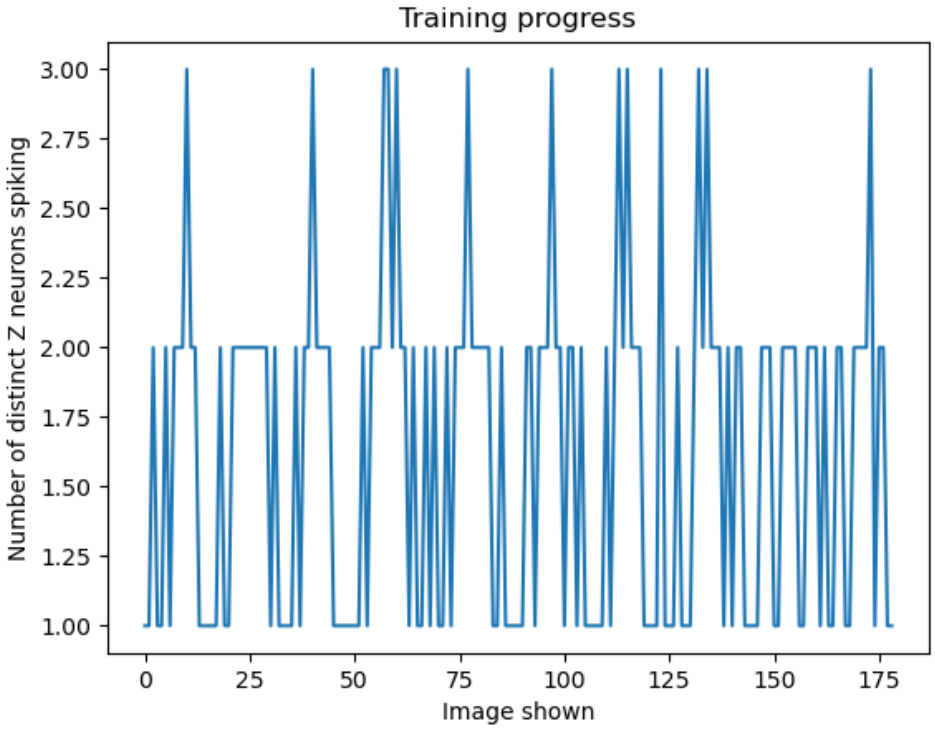

Figure 1.11: Number of distinct output neurons active during the presentation duration of each training image. x-axis: image shown, y-axis: distinct output neuron active, $c = 20, \lambda = 10^{-3}$

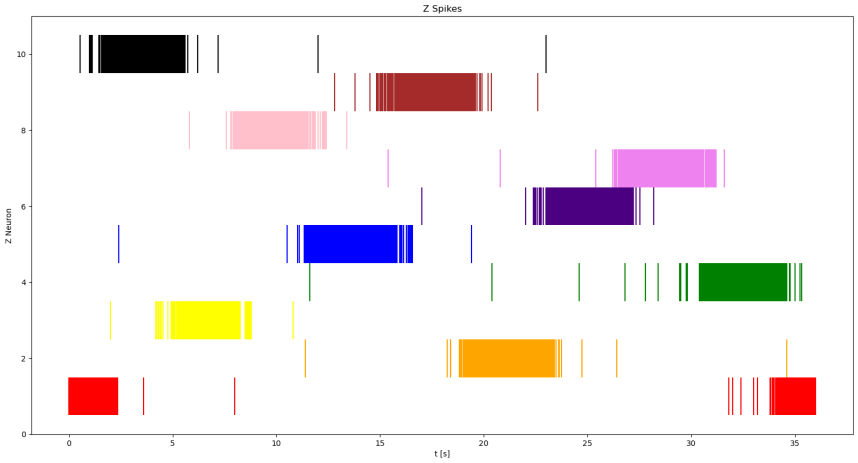

Figure 1.12: Output spikes over the presentation of input images from 0 to 179°, $c = 20, \lambda = 10^{-3}$

raising the membrane potential needlessly.

1.2 Experiment 2: Horizontal and vertical lines

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 lines 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 lines

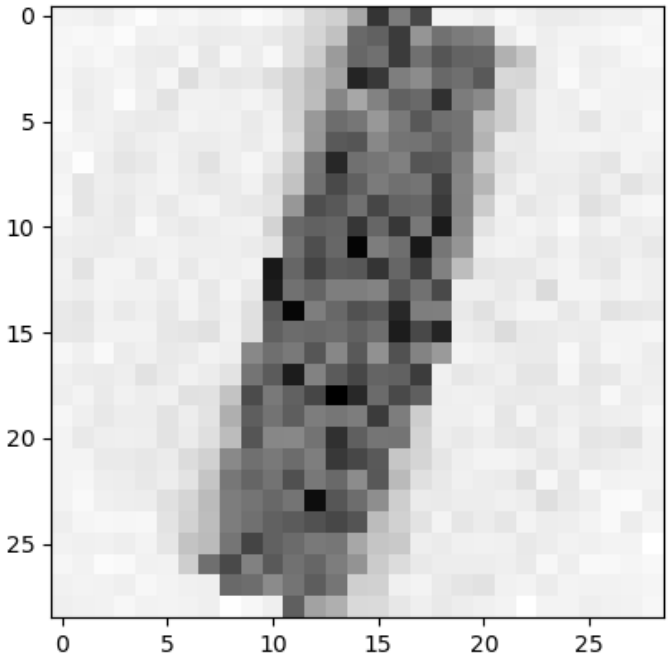

Figure 1.13: Visualization of the learned weights of y_{10}.

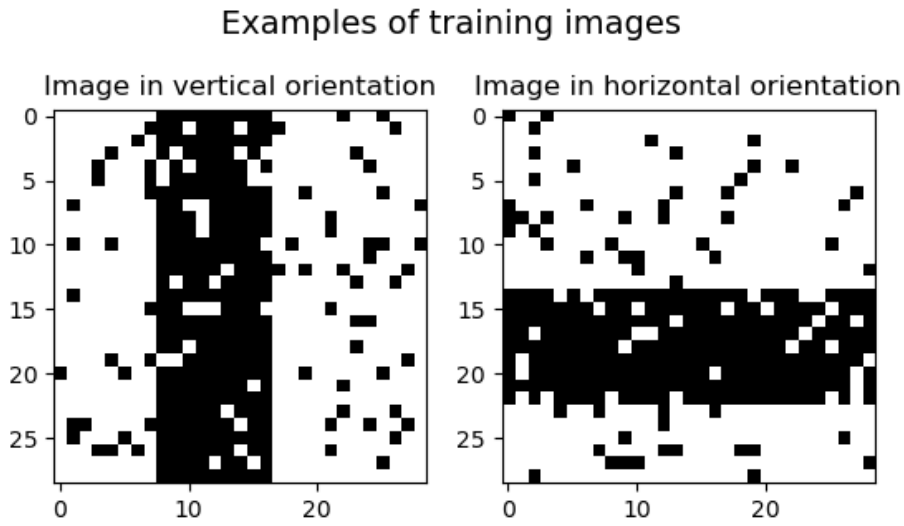

Figure 1.14: Training images generated for experiment 2. One image of each possible orientation at a random position.

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 lines 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.

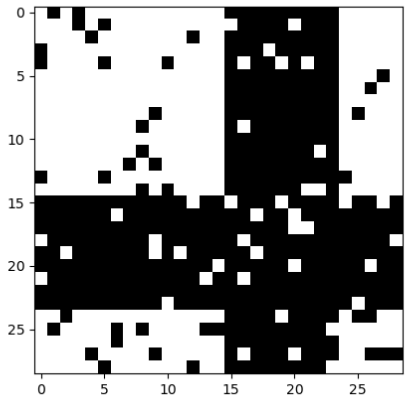

Figure 1.15: Generated cross image which can represent either horizontal or vertical orientation.

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 line 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 lines in every position and in the second plot for vertical lines 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 line 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 line 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 line can only be

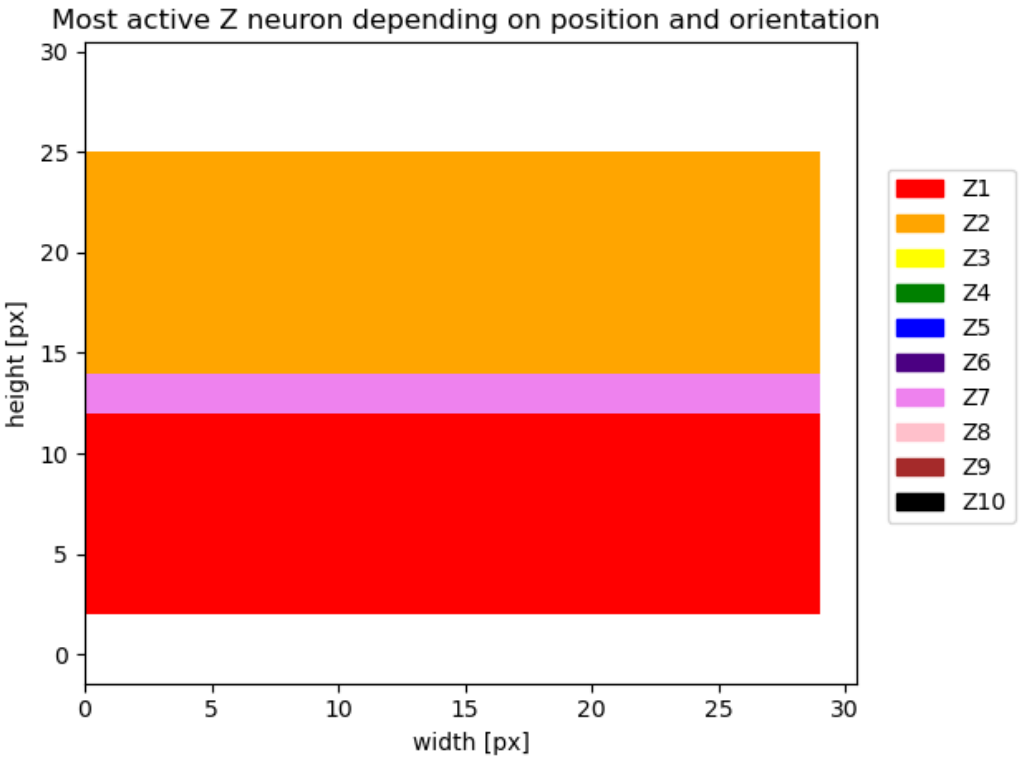

Figure 1.16: Most active output neuron for horizontal orientation and position on the y-axis of the training image during the training process. $c = 20, \lambda = 10^{-3}, Z_{factor} = 20$

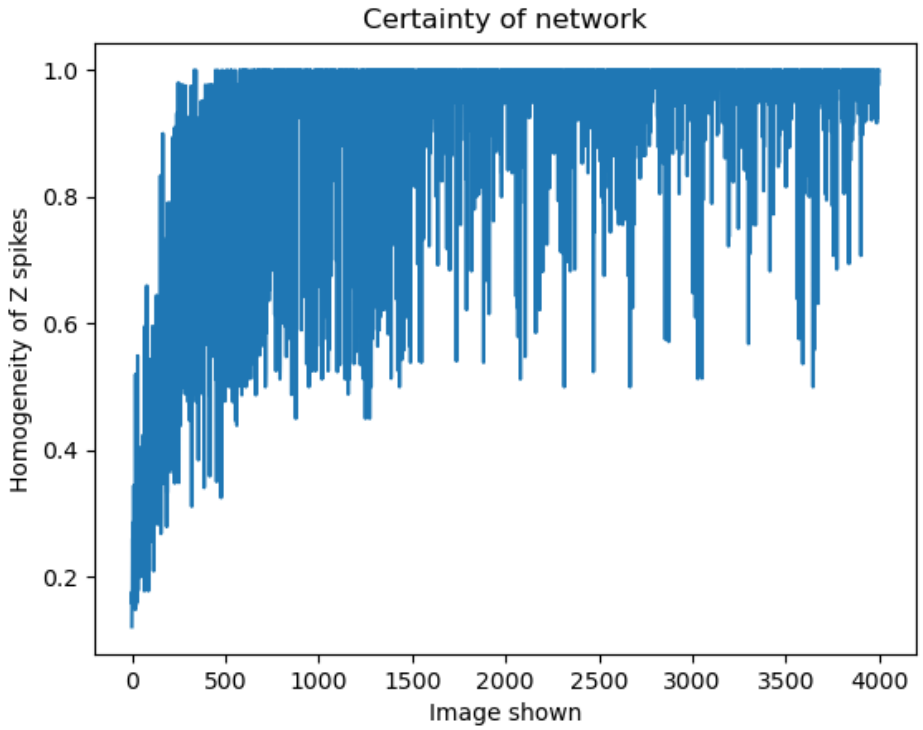

Figure 1.17: Proportion (accuracy) of most active output neuron to activity of all other output neurons during the presentation duration of each training image. $c = 20, \lambda = 10^{-3}, Z_{factor} = 5$

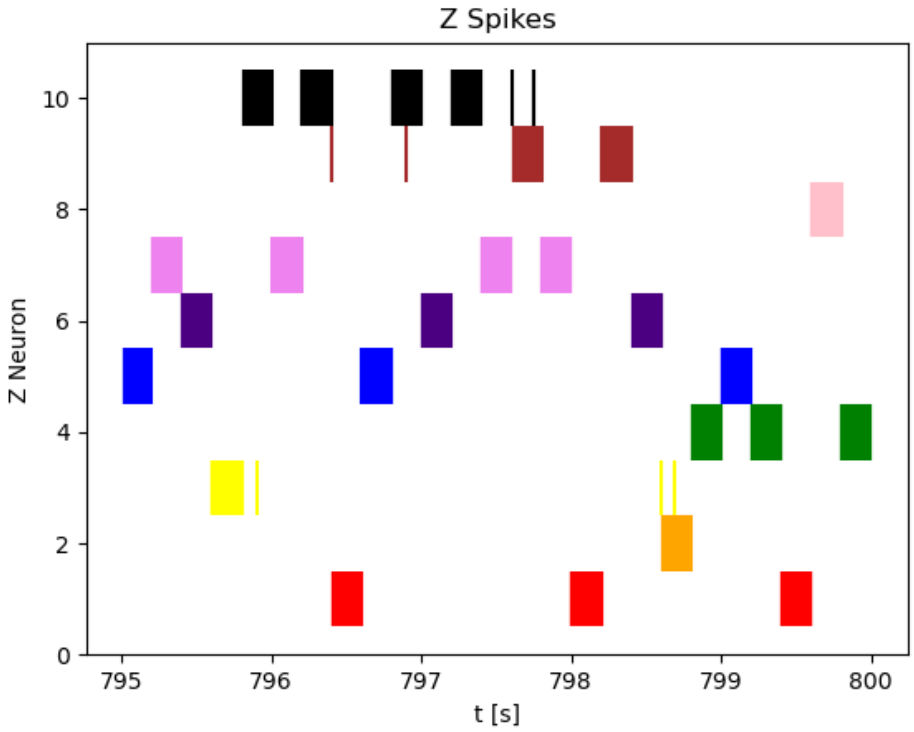

Figure 1.18: Last 1000 output neuron spikes, $c = 20, \lambda = 10^{-3}$, $Z_{factor} = 5$

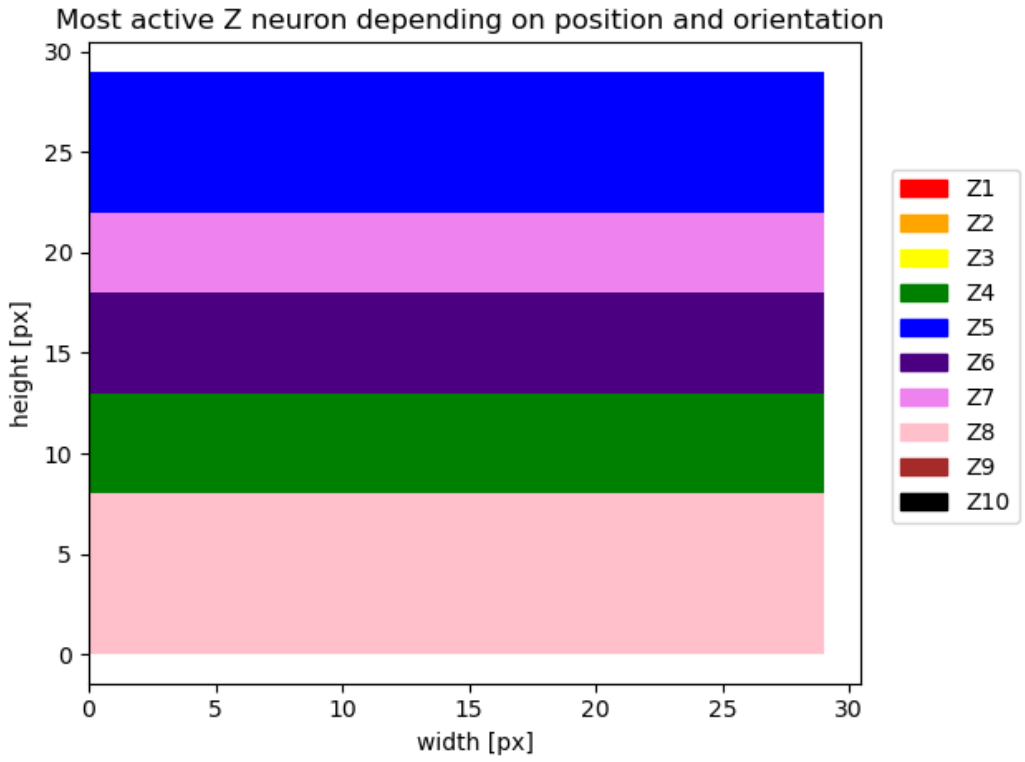

Figure 1.19: Most active output neuron for images of horizontal orientation and position on the x-axis of during the validation process. $c = 20, \lambda = 10^{-3}, Z_{factor} = 5$

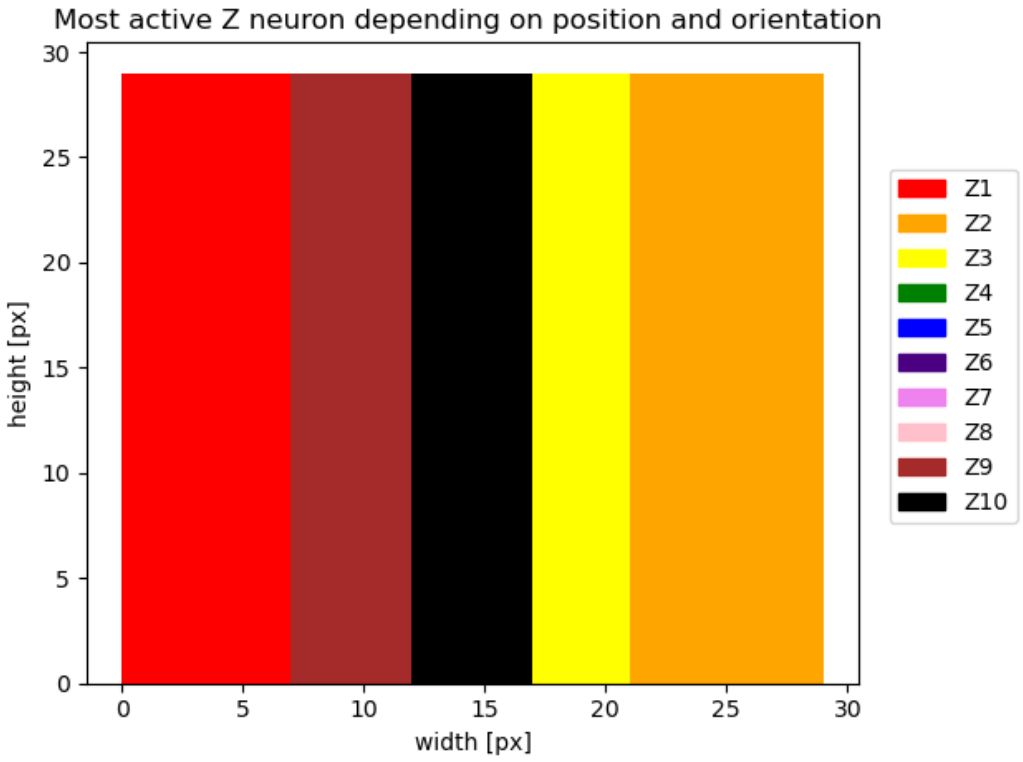

Figure 1.20: Most active output neuron for images of vertical orientation and position on the x-axis of during the validation process. $c = 20, \lambda = 10^{-3}, Z_{factor} = 5$

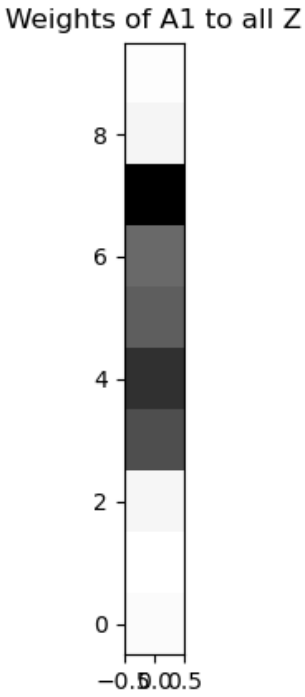

Figure 1.21: Values of w_k1, darker color means higher value. $c = 20, \lambda = 10^{-3}, Z_{factor} = 5$

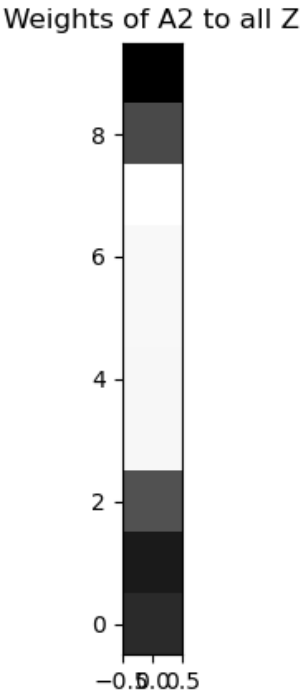

Figure 1.22: Values of $w_k 2$, darker color means higher value. $c = 20, \lambda = 10^{-3}, Z_{factor} = 5$

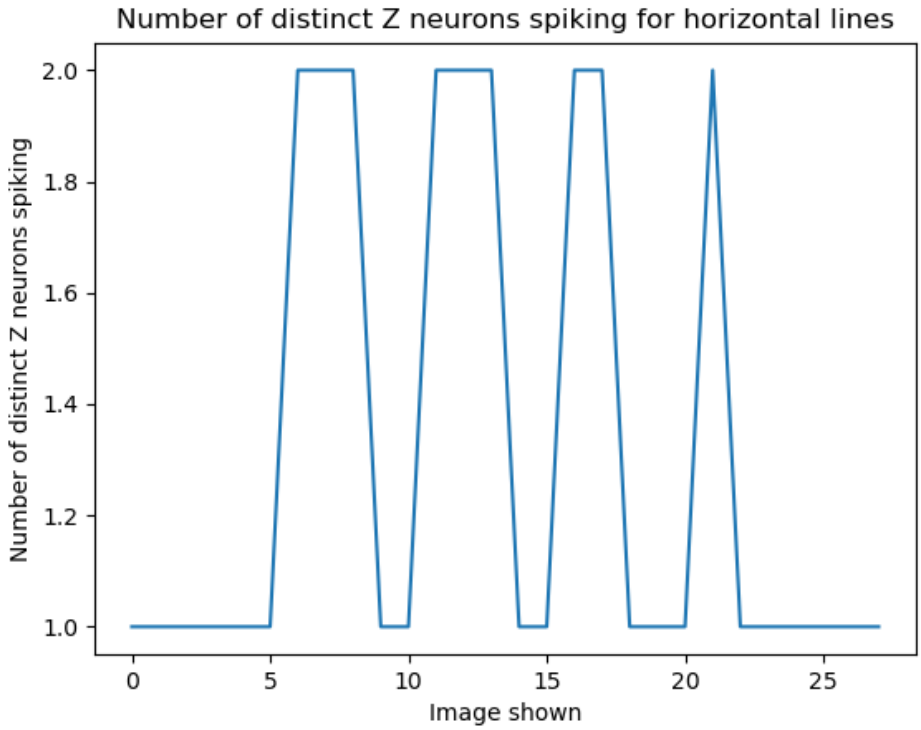

Figure 1.23: Distinct output neurons spiking during the presentation of all possible horizontal oriented validation images, $c = 20, \lambda = 10^{-3}$, $Z_{factor} = 5$

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 lines 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.

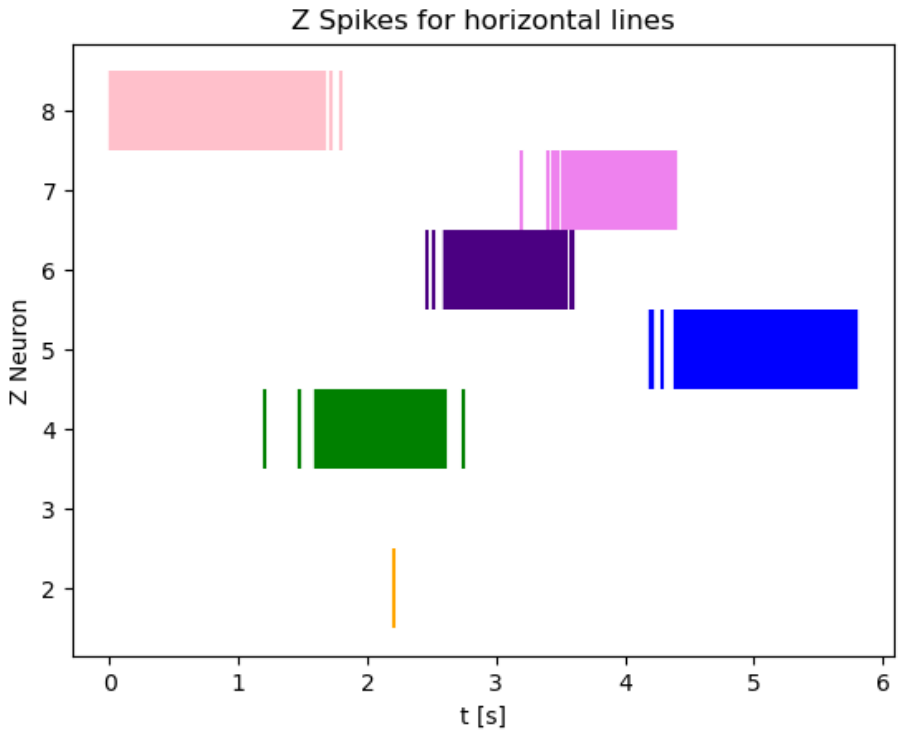

Figure 1.24: Output neuron spikes during the presentation of all possible horizontal oriented validation images, $c = 20, \lambda = 10^{-3}$, $Z_{factor} = 5$

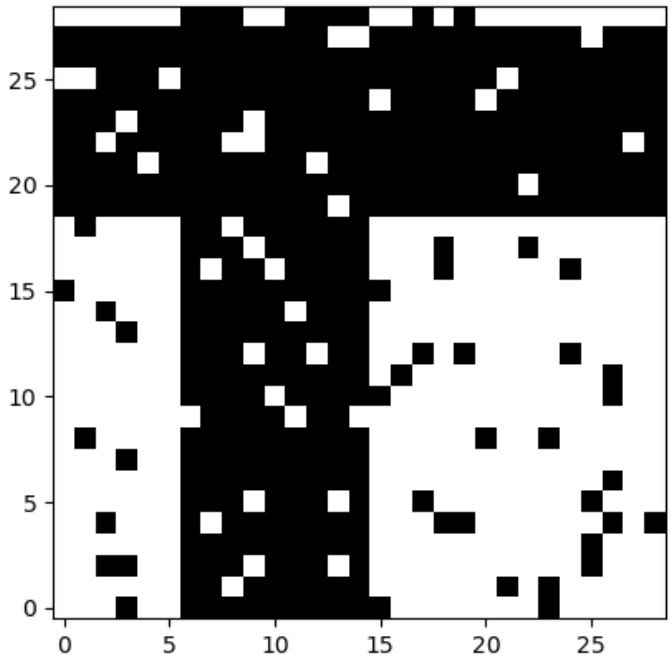

Figure 1.25: Generated validation cross image, with orientation defined as horizontal. $c = 20, \lambda = 10^{-3}, Z_{factor} = 5$

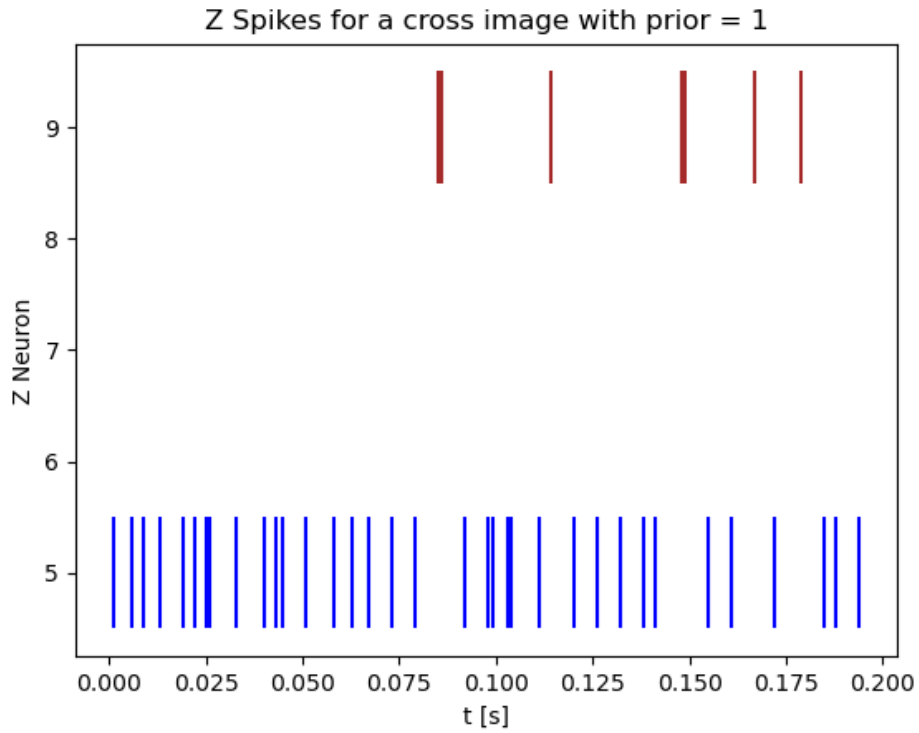

Figure 1.26: Output spikes during the presentation of the validation cross image. $c = 20, \lambda = 10^{-3}, Z_{factor} = 5$

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).