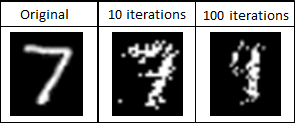
Exercise session 4: Generative models

**Restricted Boltzmann Machines**

The Boltzmann machine is a parameterized generative model that represents a probability distribution. It models the relations between the variables in a stochastic neural network. Training the general Boltzmann machine is impractical due to the large number of parameters. Learning is made more efficient in a Restricted Boltzmann Machine. The restriction implies that there are no hidden to hidden or visible to visible connections. The provided file is used to gain insight in the effect of the parameters.

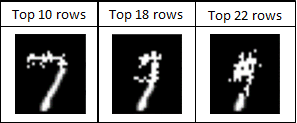
**The *learning rate* is an important hyperparameter that has to be correctly determined so that learning can be done efficiently and successfully. Training for more *epochs* leads to better results, but also increases the computational time. The optimal amount of epochs depends on the learning rate because a lower learning rate will need more epochs.** Making use of more ***components*** allows to model more complex combinations, which leads to better results. After a certain point, adding more components only leads to small improvements. Increasing the number of components also increases the learning time because more parameters have to be estimated. In order to efficiently get good results, all these parameters have to be determined properly.

Gibbs sampling is used to generate the images of the digits. This method determines the most probable state of every pixel given the values of all the other pixels. The ***number of Gibbs sampling steps*** determines how many times the procedure is applied. Figure 4.1 shows how the Gibbs sampling steps adjust the original image of a seven. Through the first iterations the image evolves to an average seven. This is obvious because the middle dash of a standard seven starts to appear. After a lot of sampling steps the image evolves to an ambiguous figure in which multiple digits could be recognized. This is because the probability distribution is trained for multiple digits.



*Figure 4.1: Visualisation of the adjusted images of the digit 7 after multiple Gibbs sampling steps.*

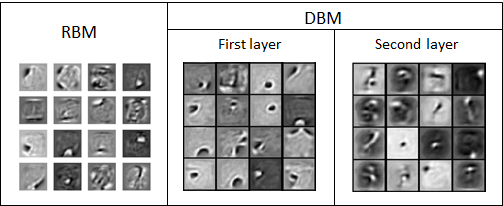
The trained model can be used for reconstructing missing parts of images. The reconstruction quality depends on the parameters in the same way as described above. The model is able to make reasonable reconstructions as long as not too much information is removed from the original image. It returns the most probable values of the deleted variables based on the remaining information. The removal of more rows leads to a higher information loss. Also the place of the removal is important, because some places are more determinative for the figure. The higher the information loss, the worse the quality of the reconstructions. This effect is shown in figure 4.2 for the removal of the top part. Similar results are found for the removal of the bottom and middle rows



*Figure 4.2: Visualisation of the reconstruction of the deleted pixels.*

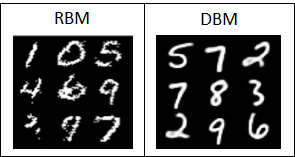
**Deep Boltzmann Machines**

The deep Boltzmann machine is composed of multiple layers of hidden units. There is full connectivity between the layers, but no connectivity within layers or between non-neighbouring layers. Figure 4.3 shows a selection of the interconnection weights of the RBM and the DBM. The RBM layer and the first layer of the DBM are directly linked to the pixels of the image. So the combination of these features determines the final image. The DBM has a second layer on top of the first layer, which allows to combine the features of the first layer into more complex features.



*Figure 4.3: Visualisation of 16 of the interconnections weights in the RBM and DBM layers.*

The features in the RBM are more comprehensive because it has to capture as much information as possible in the only layer. The DBM can capture more primitive features, like edges in the first layer because they can be combined to more complex features in the layer above. The use of multiple hidden layers significantly improves the quality of the generated digits. The difference is shown visually in figure 4.4. It can be seen that the images generated by the DBM are better readable. These digits are smoother and less noisy because the deep architecture can capture more complexity.

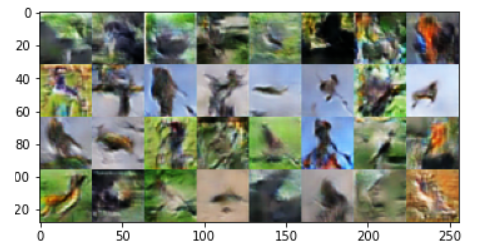
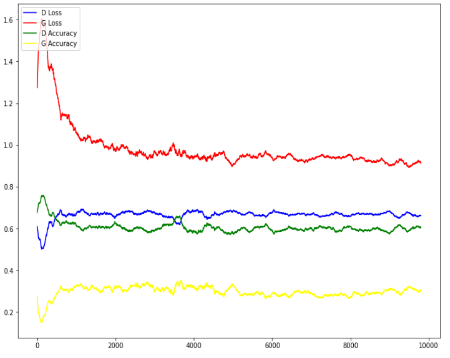


*Figure 4.4: Visualisation of generated digits of the RBM and the DBM.*

**Generative Adversarial Networks**

A generative adversarial network consists of two competing networks. The generator generates new examples while the discriminator tries to discriminate these generated examples from the original examples. Both networks adapt their weights during the training phase to achieve better results.

The trained model for this exercise follows the architecture guidelines as described in Radford et al.[[1]](#footnote-1). The model was trained for 10000 batches, which is long enough for the network to generate real looking images. The selected CIFAR dataset contains pictures of birds in nature. The final images that are shown in figure 4.5 look good. They are still a bit vague, but the theme can be easily recognized. Training the model for a longer time could further improve the quality of the generated pictures.



*Figure 4.6: Plot of the accuracy and loss of the generator and discriminator during training.*

*Figure 4.5: Visualisation of the generated images by the GAN.*

The four curves plotted in figure 4.6 are all heavily correlated with each other. The dependence between the accuracies of the generator and the discriminator is inherit to the model’s architecture. If one network is performing well, the other has to perform worse. This can be seen in the opposite movements of the green and the yellow curve. The accuracy of a network also depends heavily on its loss. The higher the loss, the lower the accuracy. This can be seen in the opposite movements of the red and yellow curve for the generator and the green and blue curve for the discriminator.

All curves start off erratic and move around a lot before the model converges to a stable equilibrium. This means that the training of the network is stable. Stability in training is very important for the networks to keep on learning. As long as no network is getting a too high accuracy, both networks can keep improving, which leads to better generated images.

When training a GAN, there is a risk of mode collapse. This means that the generator has found a particular kind of image that successfully misleads the discriminator. This would finally lead to the generator only producing this one kind of image. This phenomenon did not happen for our trained model.

**Optimal Transport**

Optimal transport theory can be used for finding the optimal transformation to change one density function into another such that the changes made are minimal. The minimal distance between two probability functions is called the Wasserstein metric, which can also be used as a similarity measure.

To test the code for swapping the colour distributions of two images, the provided pictures of the sea at day and the sea at sunset are used. The colour distributions are swapped using two techniques of optimal transport. The first technique minimizes the earth mover distance, this minimal distance is the Wasserstein distance. The second technique minimizes the Sinkhorn distance, this metric also takes into account the information entropy and therefore results in more homogeneous distributions.

Both techniques change the colour distribution with the minimal transformation costs. The resulting images are shown in figure 4.7. It can be seen that both techniques keep the relative colour differences from the original image. The darker pixels in the original image are changed to the darker colours and vice-versa. Thanks to this, the original image is still recognizable. The first technique really takes over the colours and contrasts of the original image, while the second technique gives smoother results. These techniques are different from non-optimal colour swapping. This would just assign the colours to random pixels, leading to a colour pallet that has nothing in common with the original image.

The networks of a GAN adapt their weights during the training phase. These adaptations are dependent on the loss function that is used. A standard GAN uses the minimax loss function. For this loss function, the discriminator tries to maximize the probability that all images are correctly classified between real and generated images. The generator on the other hand tries to maximize the probability that the generated images are classified as real.

**

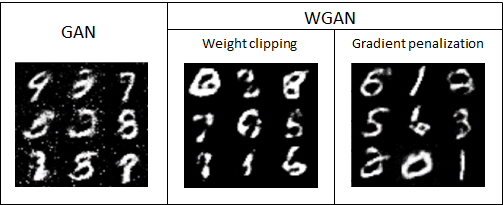
*Figure 4.7: Visualisation of the colour swapping with earth movers distance (EMD) and Sinkhorn distance.*

The actual purpose of a GAN is to replicate a probability function. So a measure for the difference between two distributions could also be a suitable loss function. The Wasserstein distance is such a measure, because it represents the minimal distance between the original data distribution and the generated data distribution. The provided file is used to compare the performance of these loss functions on generating images of written digits.

The Wasserstein GAN emits an unconstrained real number instead of a probability, so it behaves like a critic. To ensure successful training of the model, the weights should be restricted to a compact space to prevent the gradients to explode. The first WGAN obtains this property by using weight clipping after every update. The second WGAN more naturally obtains this property by penalizing the gradients.

The generated images start from random noise. The quality of the generated digits improves with the number of batches for every technique. The generated images after 10000 batches are shown in figure 4.8. All the models have a simple fully connected architecture. It is therefore logical that these architectures perform worse than the more specialized DCGAN above.

In the beginning of the training the losses fluctuate a lot, but after some time these losses start to converge for all the techniques. This means that the stability is good and the networks can keep on learning as described above. The generated images also show no indications of mode collapse. The tests show that the Wasserstein GAN’s start to stabilize earlier and they need less batches to generate readable digits. The quality of these digits is better because the images are smoother and less noisy. So using the Wasserstein distance as a loss function leads to better results. The divergence between the real data distribution and the generated data distribution therefore seems to be more informative than the minimax loss function. This is probably because the metric is more linked to the real problem of replicating the distribution function of the original images.



*Figure 4.8: Visualisation of the generated digits of the GAN and the WGAN after 10000 batches.*

1. Radford, Alec, Luke Metz, and Soumith Chintala. “Unsupervised representation learning with deep convolutional generative adversarial networks.” arXiv preprint arXiv:1511.06434 (2015) [↑](#footnote-ref-1)