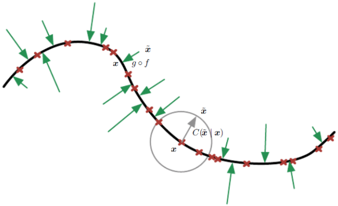
Deep Learning

Regularized Autoencoders



Learning goals

- Overcomplete AEs
- Sparse AEs
- Denoising AEs
- Contractive AEs

Overcomplete Autoencoders

OVERCOMPLETE AE – PROBLEM

Overcomplete AE (code dimension $\geq$ input dimension): even a linear AE can copy the input to the output without learning anything useful.

How can an overcomplete AE be useful?

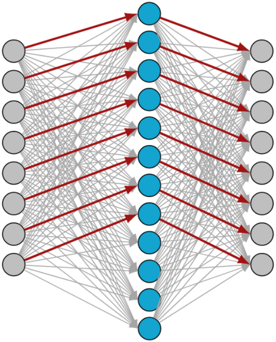


Figure: Overcomplete AE that learned to copy its inputs to the hidden layer and then to the output layer (Ponti et al., 2016).

REGULARIZED AUTOENCODER

- Goal: choose code dimension and capacity of encoder/decoder based on the problem.
- **Regularized AEs** modify the original loss function to:
  - prevent the network from trivially copying the inputs.
  - encourage additional properties.
- Examples:
  - **Sparse AE:** sparsity of the representation.
  - **Denoising AE**: robustness to noise.
  - **Contractive AE**: small derivatives of the representation w.r.t. input.

$\Rightarrow$ A regularized AE can be overcomplete and nonlinear but still learn something useful about the data distribution!

Sparse Autoencoder

SPARSE AUTOENCODER

- **Idea:** Regularization with a sparsity constraint

$$L(\mathbf{x}, dec(enc(\mathbf{x}))) + \lambda \|\boldsymbol{z}\|_1$$

- Try to keep the number of active neurons per training input low.
- Forces the model to respond to unique statistical features of the input data.

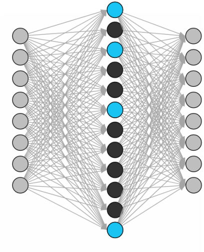


Figure: Sparse Autoencoder (Ponti et al., 2016).

Denoising Autoencoders

DENOISING AUTOENCODERS (DAE)

The denoising autoencoder (DAE) is an autoencoder that receives a corrupted data point as input and is trained to predict the original, uncorrupted data point as its output.

- Idea: representation should be robust to introduction of noise.
- Produce corrupted version $\tilde{\mathbf{x}}$ of input $\mathbf{x}$, e.g. by
  - random assignment of subset of inputs to 0.
  - adding Gaussian noise.
- Modified reconstruction loss: $L(\mathbf{x}, dec(enc(\tilde{\mathbf{x}})))$
  $\rightarrow$ denoising AEs must learn to undo this corruption.

DENOISING AUTOENCODERS (DAE)

- With the corruption process, we induce stochasticity into the DAE.
- Formally: let $C(\tilde{\mathbf{x}}|\mathbf{x})$ present the conditional distribution of corrupted samples $\tilde{\mathbf{x}}$, given a data sample $\mathbf{x}$.
- Like feedforward NNs can model a distribution over targets $p(\mathbf{y}|\mathbf{x})$, output units and loss function of an AE can be chosen such that one gets a stochastic decoder $p_{decoder}(\mathbf{x}|\mathbf{z})$.
- E.g. linear output units to parametrize the mean of Gaussian distribution for real valued $\mathbf{x}$ and negative log-likelihood loss (which is equal to MSE).
- The DAE then learns a reconstruction distribution $p_{reconstruct}(\mathbf{x}|\tilde{\mathbf{x}})$ from training pairs $(\mathbf{x}, \tilde{\mathbf{x}})$.
- (Note that the encoder could also be made stochastic, modelling $p_{encoder}(\mathbf{z}|\tilde{\mathbf{x}})$.)

DENOISING AUTOENCODERS (DAE)

The general structure of a DAE as a computational graph:

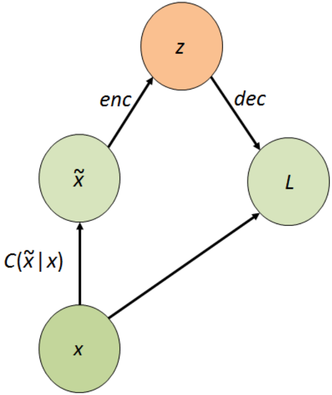


Figure: Denoising autoencoder: "making the learned representation robust to partial corruption of the input pattern."

DENOISING AUTOENCODERS (DAE)

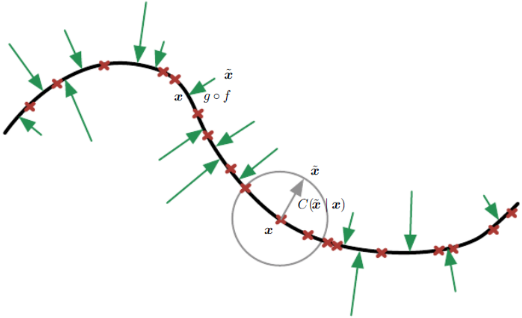


Figure: Denoising autoencoders - “manifold perspective” (Goodfellow et al., 2016).

A DAE is trained to map a corrupted data point $\tilde{\mathbf{x}}$ back to the original data point $\mathbf{x}$.

DENOISING AUTOENCODERS (DAE)

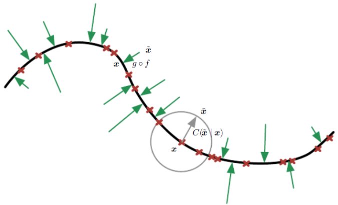


Figure: Denoising autoencoders - “manifold perspective” (Goodfellow et al., 2016).

- The corruption process $C(\tilde{\mathbf{x}}|\mathbf{x})$ is displayed by the gray circle of equiprobable corruptions
- Training a DAE by minimizing $||dec(enc(\tilde{\mathbf{x}})) - \mathbf{x}||^2$ corresponds to minimizing $\mathbb{E}_{\mathbf{x},\tilde{\mathbf{x}}\sim p_{data}(\mathbf{x})C(\tilde{\mathbf{x}}|\mathbf{x})}[-\log p_{decoder}(\mathbf{x}|f(\tilde{\mathbf{x}}))]$.

DENOISING AUTOENCODERS (DAE)

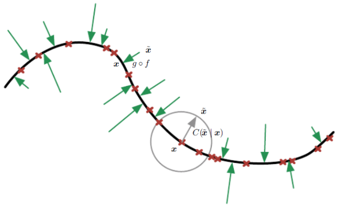


Figure: Denoising autoencoders - “manifold perspective” (Goodfellow et al., 2016).

- The vector $dec(enc(\tilde{\mathbf{x}})) - \tilde{\mathbf{x}}$ points approximately towards the nearest point in the data manifold, since $dec(enc(\tilde{\mathbf{x}}))$ estimates the center of mass of clean points $\mathbf{x}$ which could have given rise to $\tilde{\mathbf{x}}$.
- Thus, the DAE learns a vector field $dec(enc(\tilde{\mathbf{x}})) - \mathbf{x}$ indicated by the green arrows.

DENOISING AUTOENCODERS (DAE)

An example of a vector field learned by a DAE.

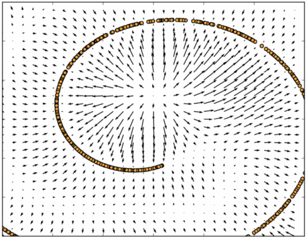


Figure: (Goodfellow et al., 2016)

EXPERIMENT: ENCODE MNIST WITH A DAE

- We will now corrupt the MNIST data with Gaussian noise and then try to denoise it as good as possible.

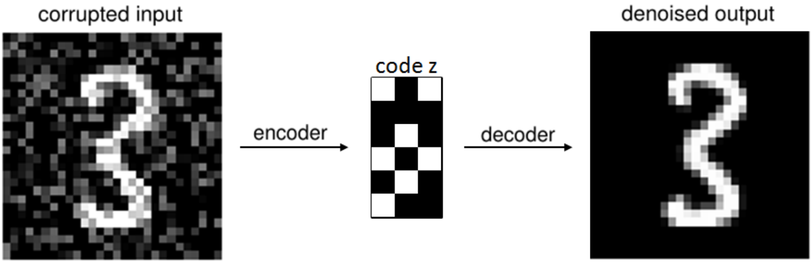


Figure: Flow chart of our autoencoder: denoise the corrupted input.

EXPERIMENT: ENCODE MNIST WITH A DAE

- To corrupt the input, we randomly add or subtract values from a uniform distribution to each of the image entries.

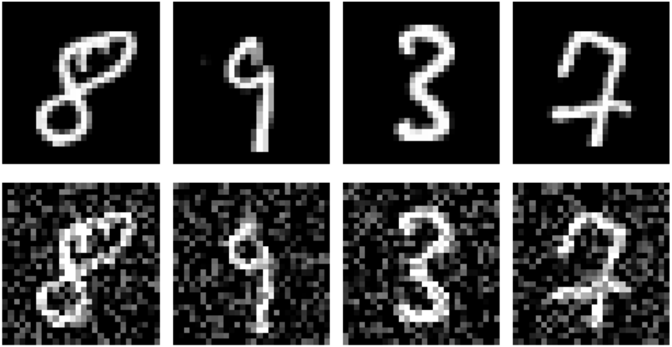


Figure: Top row: original data, bottom row: corrupted mnist data.

EXPERIMENT: ENCODE MNIST WITH A DAE

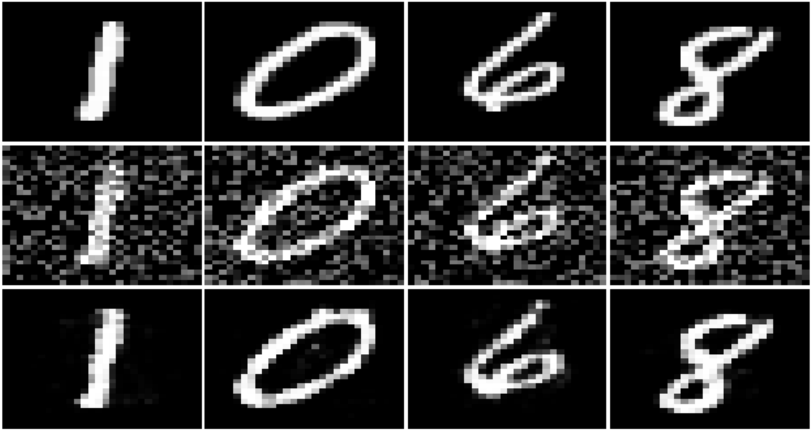


Figure: The top row shows the original digits, the intermediate one the corrupted and the bottom row the denoised/reconstructed digits (prediction).

- $dim(\boldsymbol{z}) = 1568$ (overcomplete).

EXPERIMENT: ENCODE MNIST WITH A DAE

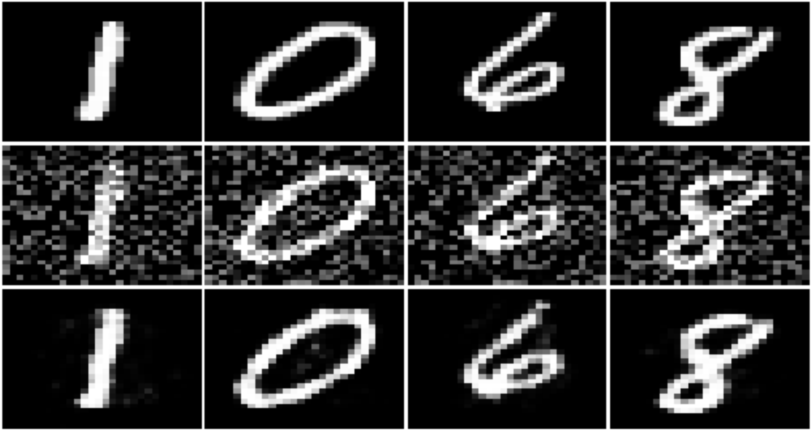


Figure: The top row shows the original digits, the intermediate one the corrupted and the bottom row the denoised/reconstructed digits (prediction).

- $dim(\boldsymbol{z}) = 784$ $(= dim(\mathbf{x}))$.

EXPERIMENT: ENCODE MNIST WITH A DAE

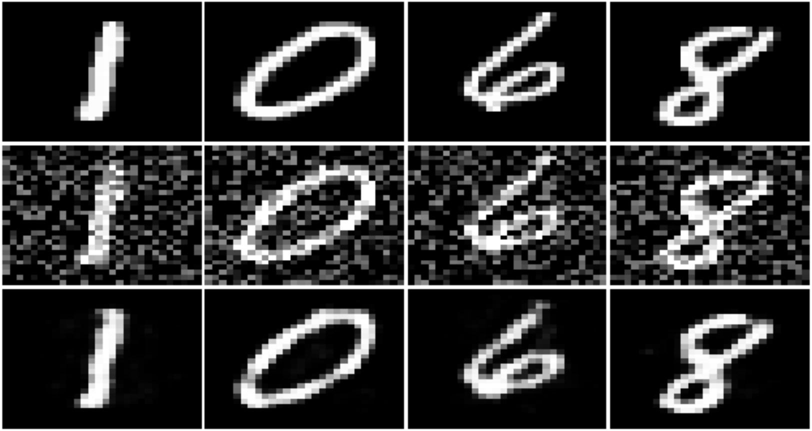


Figure: The top row shows the original digits, the intermediate one the corrupted and the bottom row the denoised/reconstructed digits (prediction).

- $dim(\boldsymbol{z}) = 256$.

EXPERIMENT: ENCODE MNIST WITH A DAE

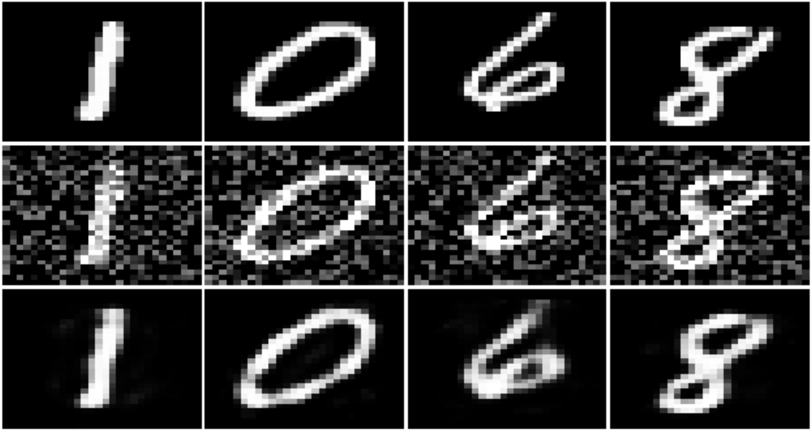


Figure: The top row shows the original digits, the intermediate one the corrupted and the bottom row the denoised/reconstructed digits (prediction).

- $dim(\boldsymbol{z}) = 64$.

EXPERIMENT: ENCODE MNIST WITH A DAE

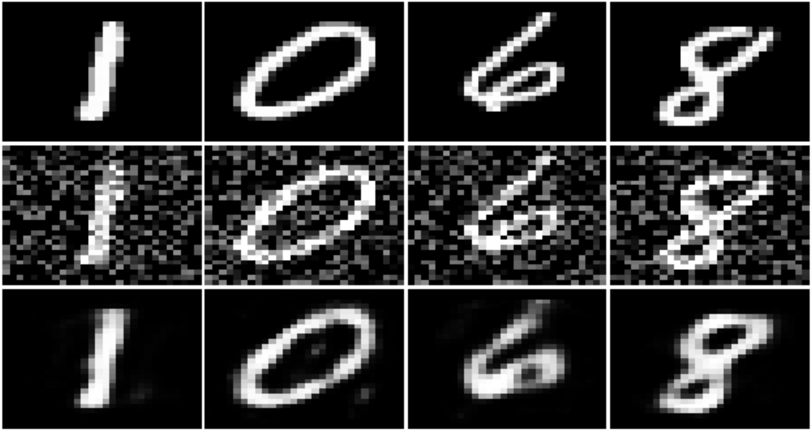


Figure: The top row shows the original digits, the intermediate one the corrupted and the bottom row the denoised/reconstructed digits (prediction).

- $dim(\boldsymbol{z}) = 32$.

EXPERIMENT: ENCODE MNIST WITH A DAE

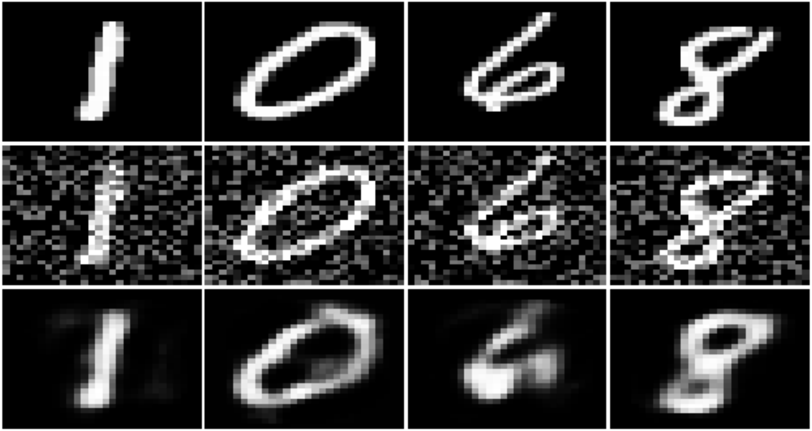


Figure: The top row shows the original digits, the intermediate one the corrupted and the bottom row the denoised/reconstructed digits (prediction).

- $dim(\boldsymbol{z}) = 16$.

EXPERIMENT: ENCODE MNIST WITH A DAE

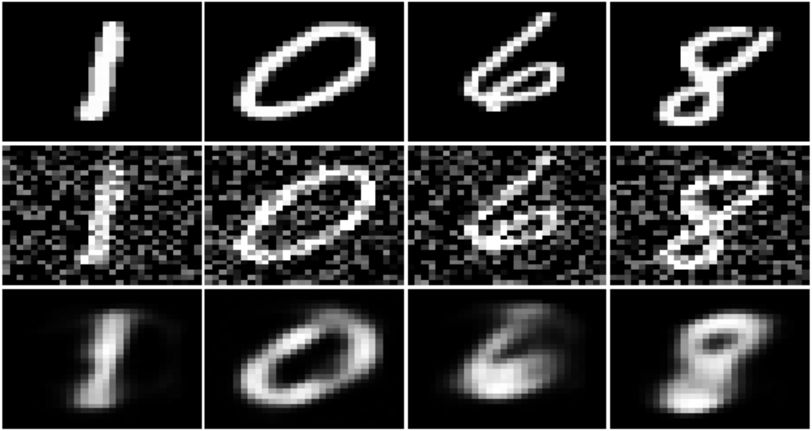


Figure: The top row shows the original digits, the intermediate one the corrupted and the bottom row the denoised/reconstructed digits (prediction).

- $dim(\boldsymbol{z}) = 8$.

Contractive Autoencoder

CONTRACTIVE AUTOENCODER

- Goal: For very similar inputs, the learned encoding should also be very similar.
- We can train our model in order for this to be the case by requiring that the **derivative of the hidden layer activations are small** with respect to the input.
- In other words: The encoded state $enc(\mathbf{x})$ should not change much for small changes in the input.
- Add explicit regularization term to the reconstruction loss:

$$L(\mathbf{x}, dec(enc(\mathbf{x})) + \lambda \| \frac{\partial enc(\mathbf{x})}{\partial \mathbf{x}} \|_F^2$$

DAE VS. CAE

DAE & **CAE**
the *decoder* function is trained to resist infinitesimal perturbations of the input. & the *encoder* function is trained to resist infinitesimal perturbations of the input.

- Both the denoising and contractive autoencoders perform well.
- Advantage of denoising autoencoder: simpler to implement
  - requires adding one or two lines of code to regular AE.
  - no need to compute Jacobian of hidden layer.
- Advantage of contractive autoencoder: gradient is deterministic
  - can use second order optimizers (conjugate gradient, LBFGS, etc.).
  - might be more stable than the denoising autoencoder, which uses a sampled gradient.

REFERENCES

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.

M. A. Ponti, L. S. F. Ribeiro, T. S. Nazare, T. Bui and J. Collomosse, "Everything You Wanted to Know about Deep Learning for Computer Vision but Were Afraid to Ask," *2017 30th SIBGRAPI Conference on Graphics, Patterns and Images Tutorials (SIBGRAPI-T)*, Niteroi, Brazil, 2017, pp. 17-41, doi: 10.1109/SIBGRAPI-T.2017.12.