Pattern Analysis & Machine Intelligence Praktikum: MLPR-19

Martin Mundt, Dr. Iuliia Pliushch, Prof. Dr. Visvanathan Ramesh

Goethe Uni Frankfurt

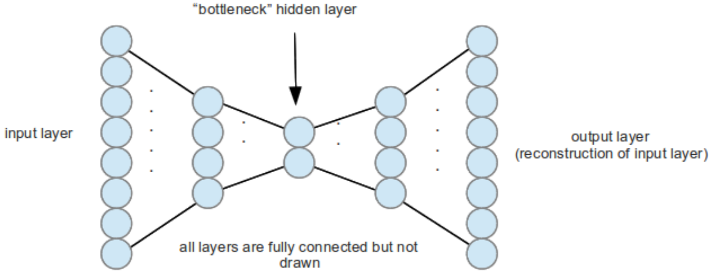
Introduction to Unsupervised Neural Networks

General concepts

- Make use of large quantities of **unlabeled data**
- Learn the **structure of the data**. Could be used for e.g. clustering (k-means, mixture models) for information retrieval, data compression, statistical data analysis etc.
- In NNs for **"Representation Learning"**: a meaningful & complete set of features describing the data.

Autoencoders

- Learn a meaningful representation ("encoding") of the complete data
- An **encoder** maps to a **"code"** or **"latent variables"/"latent representations"** and is then fed into a **decoder** to **reconstruct** the input.
- In the simplest version a decoder is a "flipped" version of the encoder (with shared weights).
- The loss function could be e.g. a mean squared error between the original image and the reconstructed (decompressed) version.



Linear Autoencoders and PCA

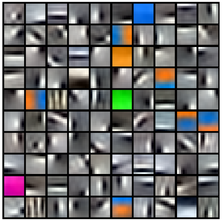
What's the relationship between Autoencoders and PCA?

- A single layer linear autoencoder with $\mathbf{h} = \mathbf{W_1 x} + \mathbf{b_1}$ and $\mathbf{x}' = \mathbf{W_2 h} + \mathbf{b_2}$ is said to apply PCA to the input, if the dimensionality of the embedding $\mathbf{W_1} \in \mathcal{R}^{n \times m}$ and $\mathbf{W_2} \in \mathcal{R}^{m \times n}$ is smaller than the input $n < m$.
- This is because the the autoencoder projects the data onto a low dimensional principal subspace. We can take the learned weight and apply singular value decomposition to recover the m vectors.
- No sorting of bottleneck output $\mathbf{h}$ in terms of variance, but SVD in the autoencoder case can be applied on a $n \times m$ matrix, whereas $\mathbf{x}$ is a $n \times n$ matrix.
- The main point of autoencoders is not to find a way to do PCA for big datasets, but we might receive a nice intuition. Practically, all our autoencoders will be nonlinear.

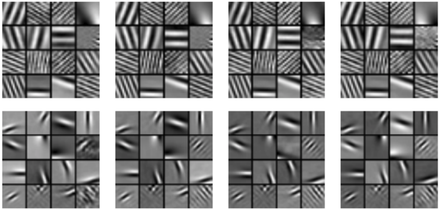
See https://arxiv.org/abs/1804.10253 for more information and exact relationships.

Autoencoders

- Can come in all sorts of flavors (fully-connected layers, convolutions etc.)
- Can be trained and stacked (greedy-layer-wise training)
- Can be extended to deep networks by adding more hidden layers in both encoder and decoder
- For large quantities of unlabeled data, can be used as pre-training for semi-supervised learning



https://www.groundai.com/project/zero-bias-autoencoders-and-the-benefits-of-co-adapting-features



https://arxiv.org/abs/1306.3162

Autoencoder semi-supervised pre-training

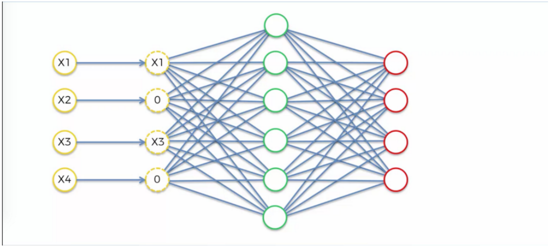
Given a large amount of unlabelled and smaller amount of data with labels:

1. Unsupervised training of autoencoder
2. "Detach" decoder / transfer encoder with pre-trained weights
3. Add a classifier (e.g. a single linear layer) to the pre-trained encoder
4. Fine-tune with the labelled data (mostly with a lower learning rate or even just the classifier's weights)

Denoising Autoencoders

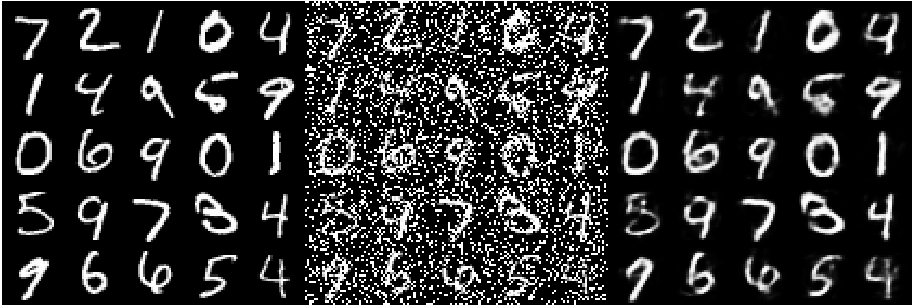
What if we are not reducing dimensionality?

- If the architecture doesn't have a bottleneck (i.e. more hidden nodes than input nodes) the AE can just learn an identity
- We can randomly corrupt the data on purpose to solve this issue, that is add a noise process
- Keep in mind that we still compute the reconstruction error with the original data!



https://towardsdatascience.com/denoising-autoencoders-explained-dbb82467fc2

Denoising Autoencoders



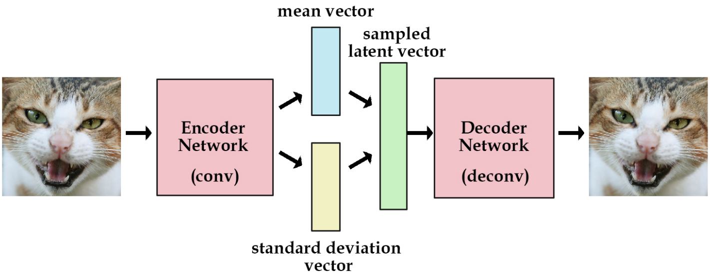
https://www.doc.ic.ac.uk/ js4416/163/website/img/autoencoders/denoising-example.png

Other variants of Autoencoders

- **Sparse Autoencoders**: similar to Denoising AE; useful for large number of hiddens by imposing sparsity
- **Contractive Autoencoders**: also similar to Denoising AE: adds explicit regularizer to objective function to learn robustness to variations
- **Variational Autoencoders**: our content in next week's class!

Variational Autoencoders

- Our AE has a latent embedding/variables in the hidden layer connecting encoder to decoder. But difficult to grasp as it is unconstrained.
- We can add a constraint such as forcing the latent vector to follow a unit Gaussian (e.g. by optimizing KL divergence in addition to reconstruction, which measures how close we are to a unit Gaussian).

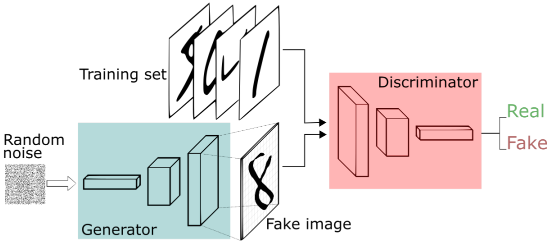


http://kvfrans.com/content/images/2016/08/vae.jpg

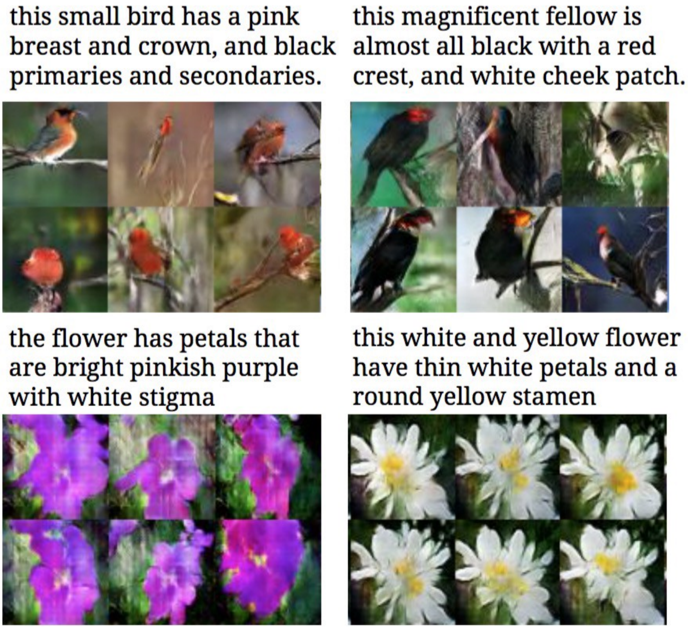
Generative Adversarial Networks (GAN)

- Various methods to use NNs to generate data
- GANs are one further option: decoder is trained to generate output that resembles training data to fool a different "discriminator" (C)NN.
- Training is based on the discriminator being able to distinguish real from generated samples and generator in turn learning to produce more "realistic" data.

We will delve into GANs in our class in two weeks!



https://deeplearning4j.org/generative-adversarial-network



https://medium.com/intuitionmachine/navigating-the-unsupervised-learning-landscape-951bd5842df9

- Unsupervised pre-training (semi-supervised learning) doesn't necessarily hold-up to its promise when enough labeled data is available. Supervised learning has more meaningful features with respect to a task.
- It is difficult to distinguish "background concepts" from "content" in unsupervised learning.
- In images, learning is conducted with metrics on a pixel-level (and not e.g. concepts or entities).
- In clustering the objective is not taken into account in the distance metric. Number of clusters typically not known a priori.
- Stability of very deep NN optimization, especially for GANs.
- Validation of generative models (such as GANs) or clustering algorithms can be difficult.

Our next practical session

1. Let's take our FashionMNIST or Kuzushiji dataset and adapt our existing neural networks to both autoencoders and convolutional autoencoders. We can reuse all the pieces but no longer require the data labels. Apart from monitoring the loss, we should visualize the reconstructed images during training.
2. Re-use the learned feature base of any of your autoencoders & train a classifier (e.g. a single linear layer) on top using a subset of the train data (with labels).
3. (Optional): We can adapt the above autoencoders to denoising autoencoders in representation learning with higher dimensional spaces.