Generative Models (part 1)

Unsupervised learning with autoencoders

Rafał Nowak May 20, 2025

following “Lecture 19.pdf” (see SKOS)
slides 7-38

U-Net (recap)

U-Net: Convolutional Networks for Biomedical Image Segmentation

Olaf Ronneberger, Philipp Fischer, and Thomas Brox

Computer Science Department and BIOSS Centre for Biological Signalling Studies,
University of Freiburg, Germany
ronneber@informatik.uni-freiburg.de,
WWW home page: http://lmb.informatik.uni-freiburg.de/

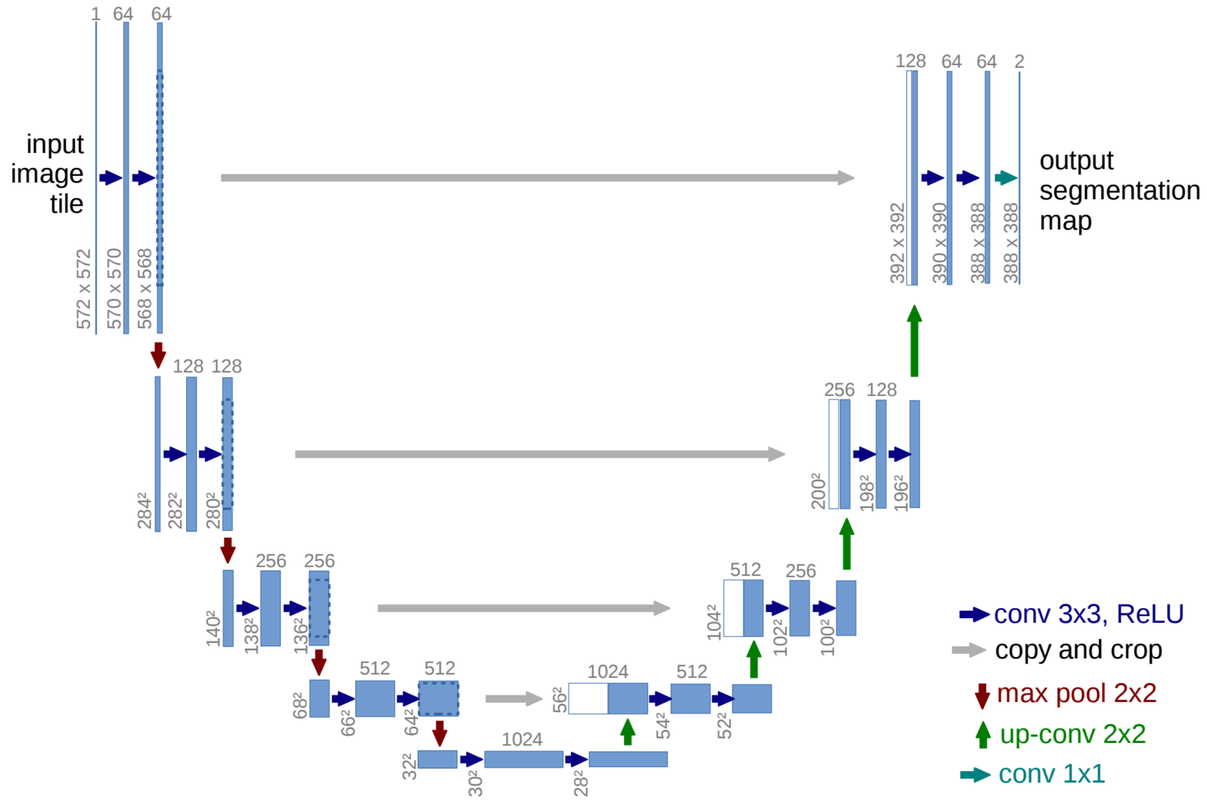


Fig. 1. U-net architecture (example for 32x32 pixels in the lowest resolution). Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations.

SegNet

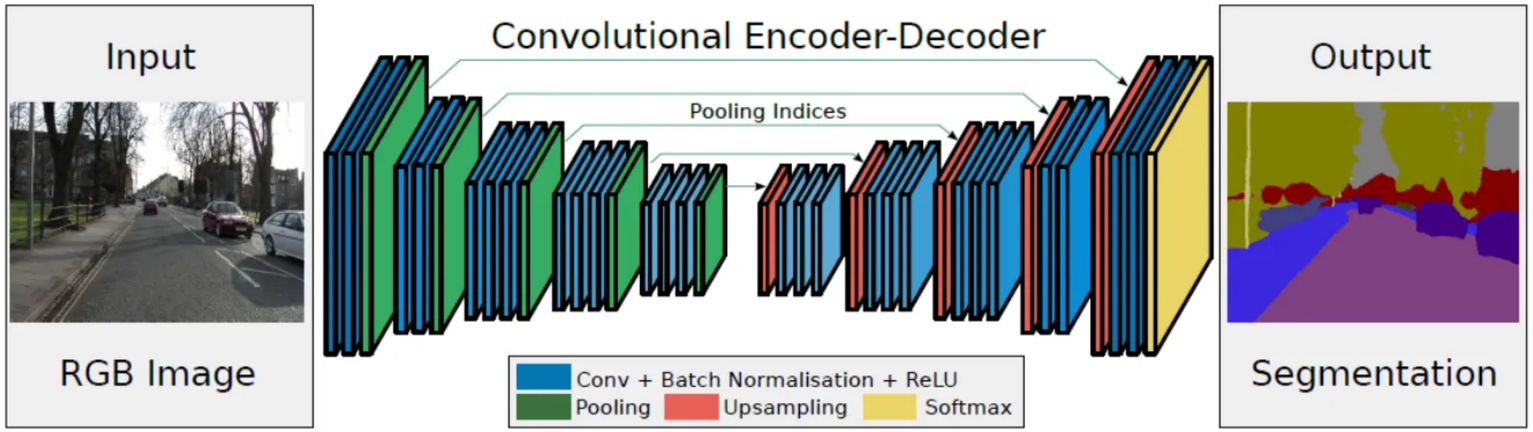
A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation

SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation

Vijay Badrinarayanan, Alex Kendall, Roberto Cipolla, *Senior Member, IEEE,*

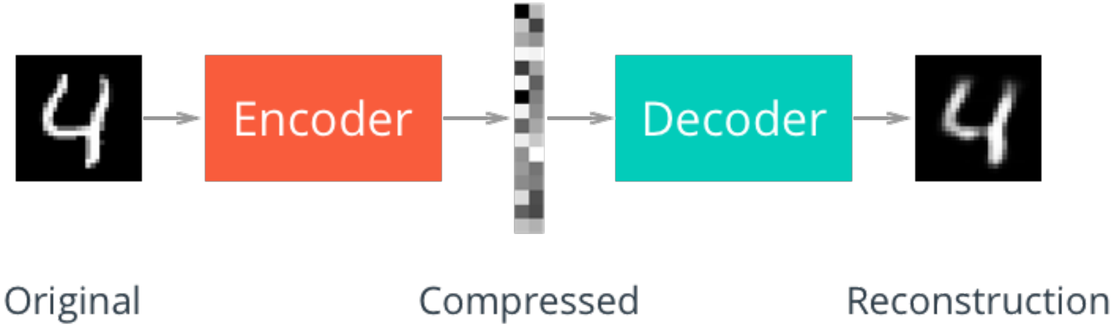
Abstract—We present a novel and practical deep fully convolutional neural network architecture for semantic pixel-wise segmentation termed SegNet. This core trainable segmentation engine consists of an encoder network, a corresponding decoder network followed by a pixel-wise classification layer. The architecture of the encoder network is topologically identical to the 13 convolutional layers in the VGG16 network [1]. The role of the decoder network is to map the low resolution encoder feature maps to full input resolution feature maps for pixel-wise classification. The novelty of SegNet lies is in the manner in which the decoder upsamples its lower resolution input feature map(s). Specifically, the decoder uses pooling indices computed in the max-pooling step of the corresponding encoder to perform non-linear upsampling. This eliminates the need for learning to upsample. The upsampled maps are sparse and are then convolved with trainable filters to produce dense feature maps. We compare our proposed architecture with the widely adopted FCN [2] and also with the well known DeepLab-LargeFOV [3], DeconvNet [4] architectures. This comparison reveals the memory versus accuracy trade-off involved in achieving good segmentation performance.
SegNet was primarily motivated by scene understanding applications. Hence, it is designed to be efficient both in terms of memory and computational time during inference. It is also significantly smaller in the number of trainable parameters than other competing architectures and can be trained end-to-end using stochastic gradient descent. We also performed a controlled benchmark of SegNet and other architectures on both road scenes and SUN RGB-D indoor scene segmentation tasks. These quantitative assessments show that SegNet provides good performance with competitive inference time and most efficient inference memory-wise as compared to other architectures. We also provide a Caffe implementation of SegNet and a web demo at http://mi.eng.cam.ac.uk/projects/segnet/.

Index Terms—Deep Convolutional Neural Networks, Semantic Pixel-Wise Segmentation, Indoor Scenes, Road Scenes, Encoder, Decoder, Pooling, Upsampling.



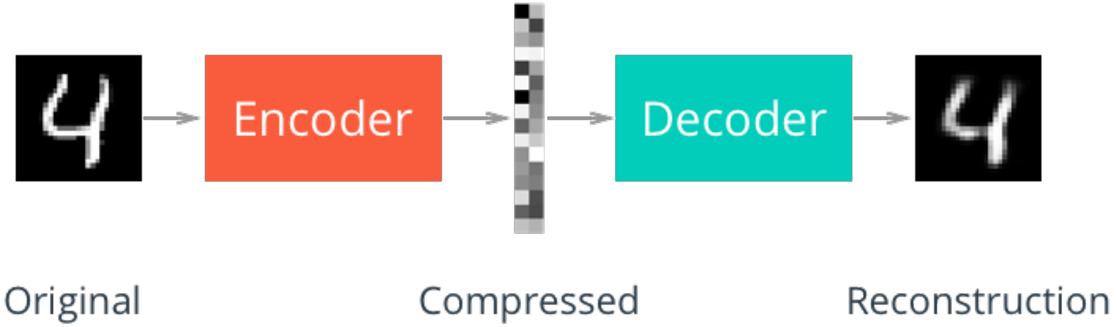
SegNet: Encoder Decoder Architecture

Autoencoder

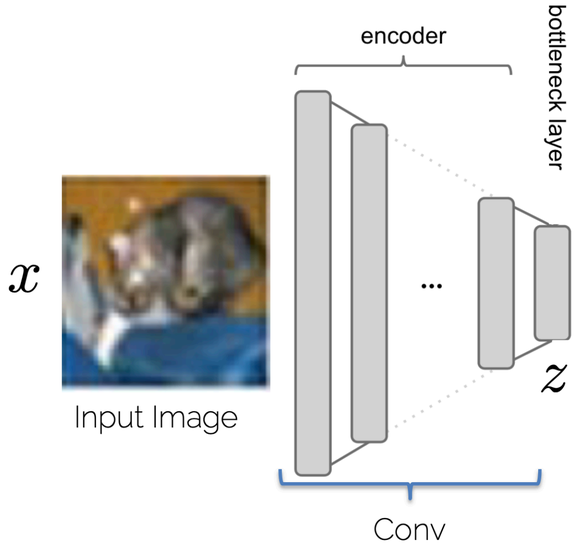


- Unsupervised approach for learning a lower-dimensional feature representation from unlabeled training data
- Autoencoding is training a network to **replicate** its input to its output
- Applications:
  - Data compression
  - Learning **embeddings** to support information retrieval
  - Unlabeled pre-training for semi-supervised learning
  - Generation of new instances similar to those in the training set

Autoencoder



Autoencoders



- From an input image to a feature representation (bottleneck layer)
- Encoder: a CNN in our case

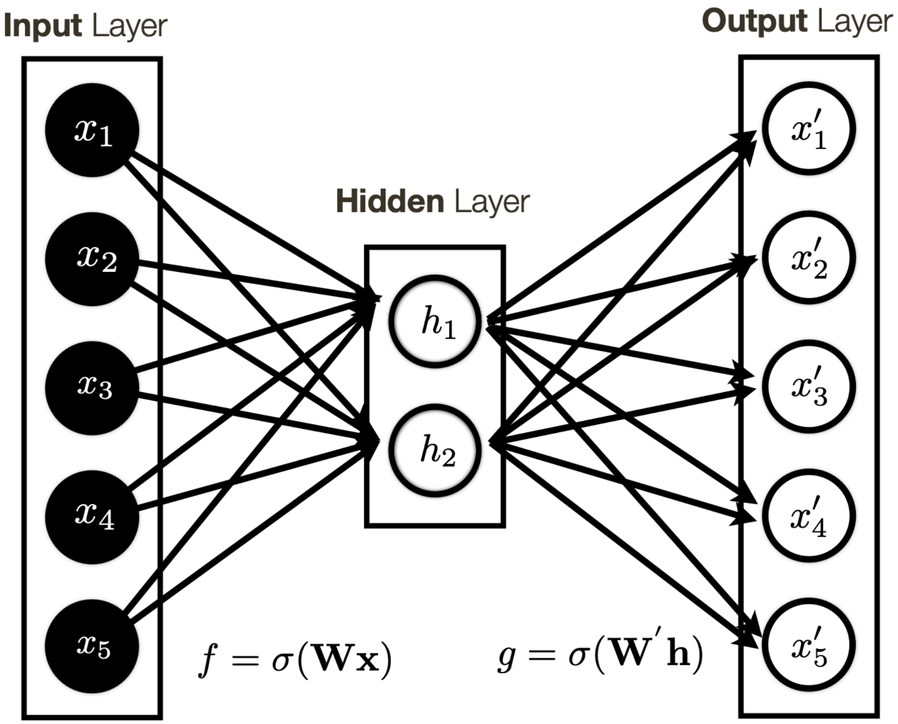
Denoising autoencoder

Idea: add noise to input but learn to reconstruct the original

— Leads to better representations

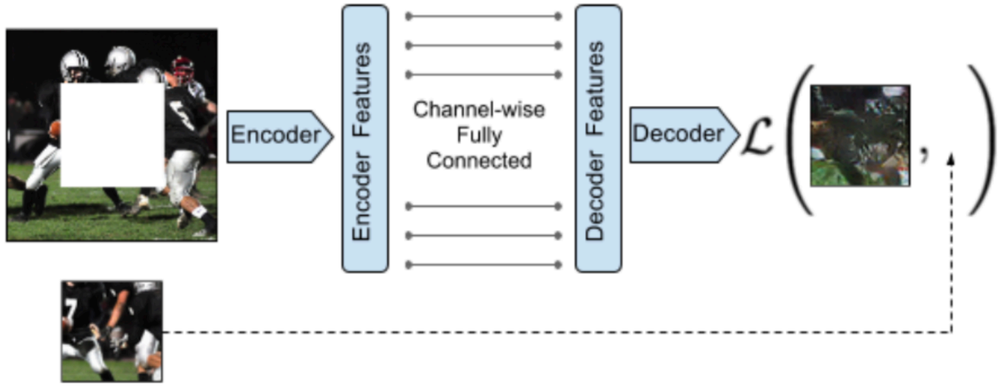
— Prevents copying

Note: different noise is added during each epoch



*slide from Louis-Philippe Morency

Context autoencoder



Context Encoders: Feature Learning by Inpainting

Deepak Pathak Philipp Krähenbühl Jeff Donahue Trevor Darrell Alexei A. Efros
University of California, Berkeley
{pathak,philkr,jdonahue,trevor,efros}@cs.berkeley.edu

Abstract

We present an unsupervised visual feature learning algorithm driven by context-based pixel prediction. By analogy with auto-encoders, we propose Context Encoders – a convolutional neural network trained to generate the contents of an arbitrary image region conditioned on its surroundings. In order to succeed at this task, context encoders need to both understand the content of the entire image, as well as produce a plausible hypothesis for the missing part(s). When training context encoders, we have experimented with both a standard pixel-wise reconstruction loss, as well as a reconstruction plus an adversarial loss. The latter produces much sharper results because it can better handle multiple modes in the output. We found that a context encoder learns a representation that captures not just appearance but also the semantics of visual structures. We quantitatively demonstrate the effectiveness of our learned features for CNN pre-training on classification, detection, and segmentation tasks. Furthermore, context encoders can be used for semantic inpainting tasks, either stand-alone or as initialization for non-parametric methods.

(a) Input context (b) Human artist

(c) Context Encoder ($L2$ loss) (d) Context Encoder ($L2$ + Adversarial loss)

Figure 1: Qualitative illustration of the task. Given an image with a missing region (a), a human artist has no trouble inpainting it (b). Automatic inpainting using our *context encoder* trained with $L2$ reconstruction loss is shown in (c), and using both $L2$ and adversarial losses in (d).

1. Introduction

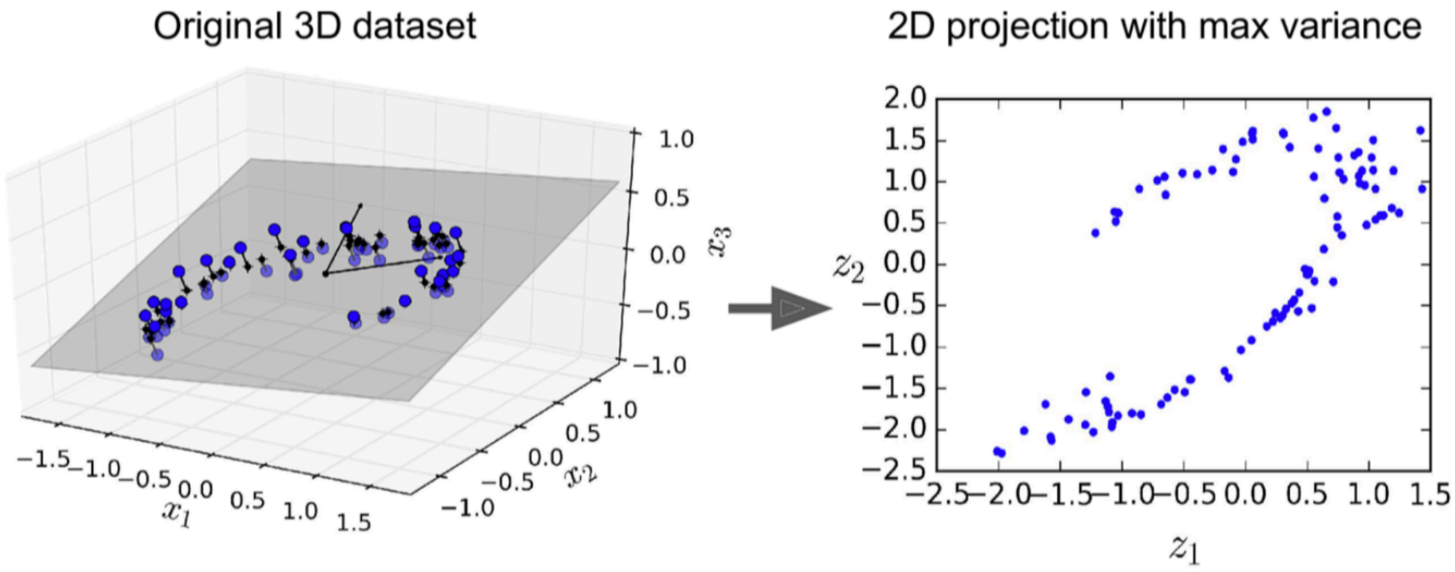
Our visual world is very diverse, yet highly structured, and humans have an uncanny ability to make sense of this

Autoencoders

Basic idea

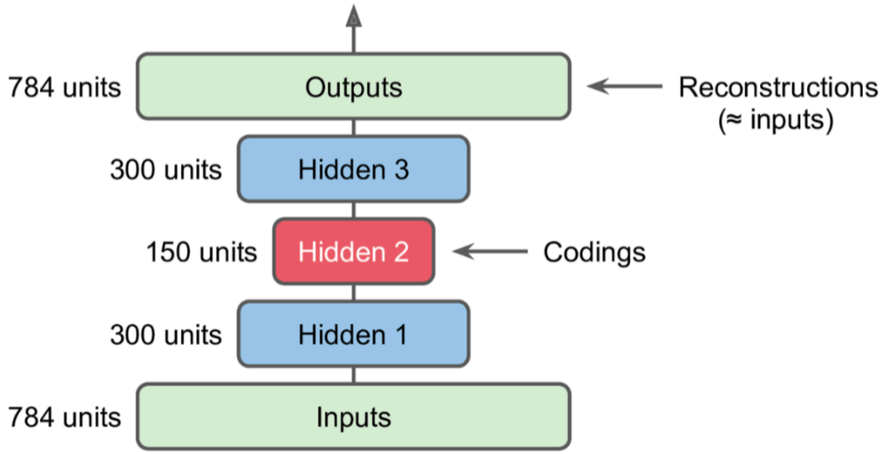
- General types of autoencoders based on size of hidden layer
  - **Undercomplete** autoencoders have hidden layer size smaller than input layer size
    - ⇒ Dimension of embedded space lower than that of input space
    - ⇒ Cannot simply memorize training instances
  - **Overcomplete** autoencoders have much larger hidden layer sizes
    - ⇒ Regularize to avoid overfitting, e.g., enforce a **sparsity** constraint

Example: PCA



- A 3-2-3 autoencoder with linear units and square loss performs **principal component analysis**: Find linear transformation of data to maximize variance

Stacked AEs



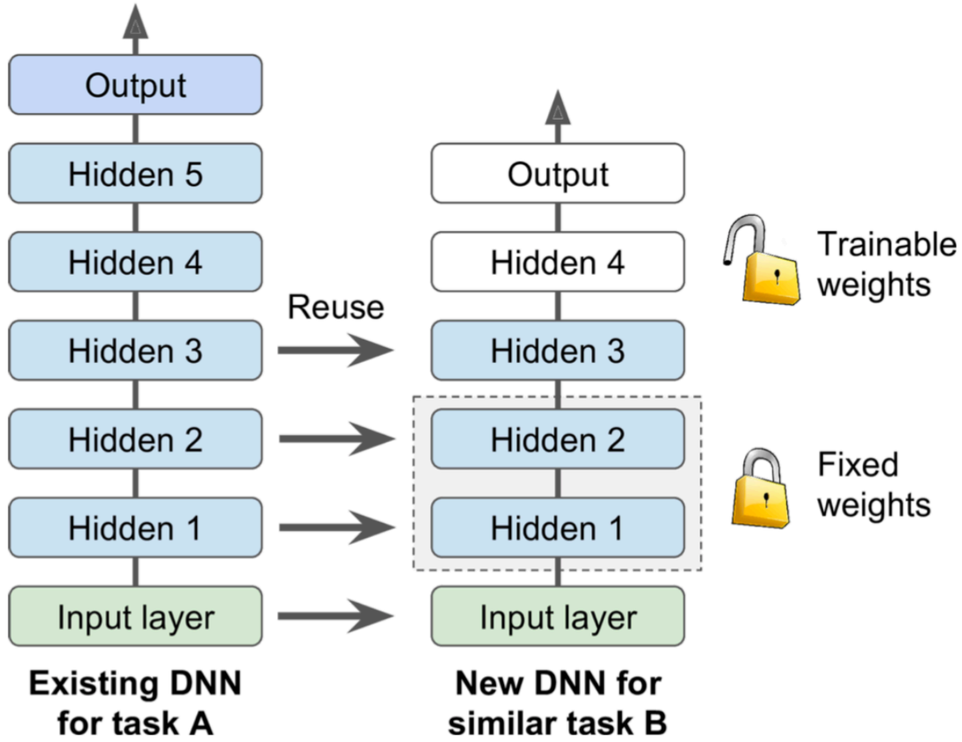
- A **stacked autoencoder** has multiple hidden layers

- Can share parameters to reduce their number by exploiting symmetry: $W_4 = W_1^\top$ and $W_3 = W_2^\top$

```
weights1 = tf.Variable(weights1_init, dtype=tf.float32, name="weights1")
weights2 = tf.Variable(weights2_init, dtype=tf.float32, name="weights2")
weights3 = tf.transpose(weights2, name="weights3")      # shared weights
weights4 = tf.transpose(weights1, name="weights4")      # shared weights
```

Transfer learning

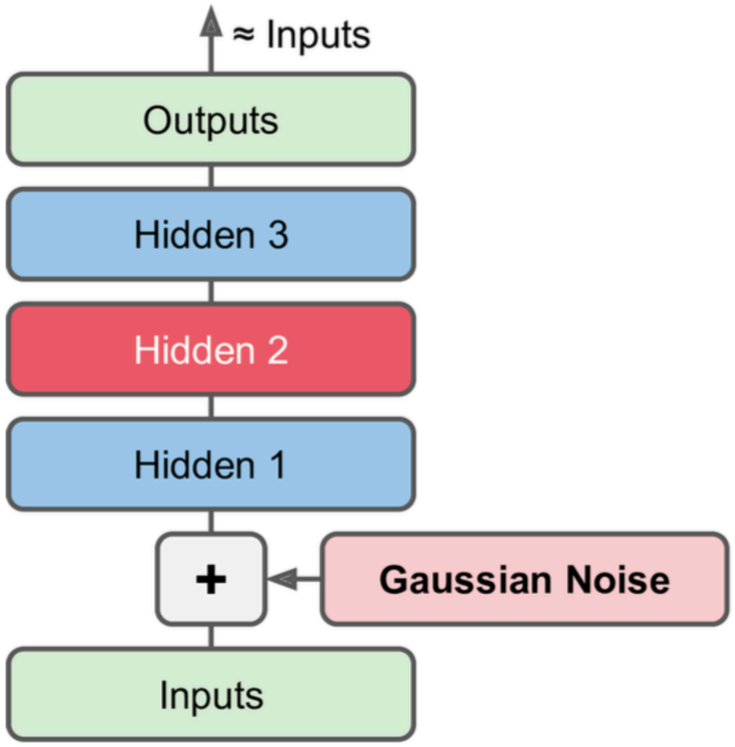
- Can also transfer from a classifier trained on different task, e.g., transfer a GoogleNet architecture to ultrasound classification

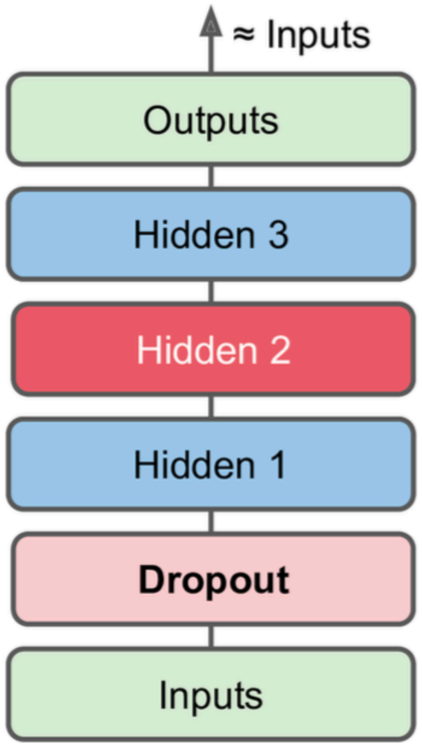


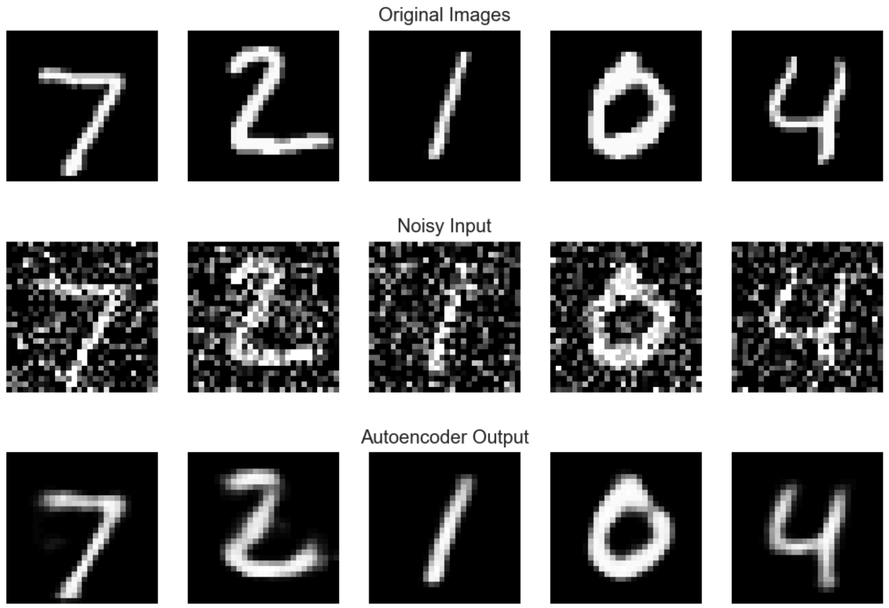
- Often choose existing one from a **model zoo**

Denoising autoencoders

Demo code







Variational Autoencoders (part 1)

following “Lecture 19.pdf” (see SKOS)
slides 65-127