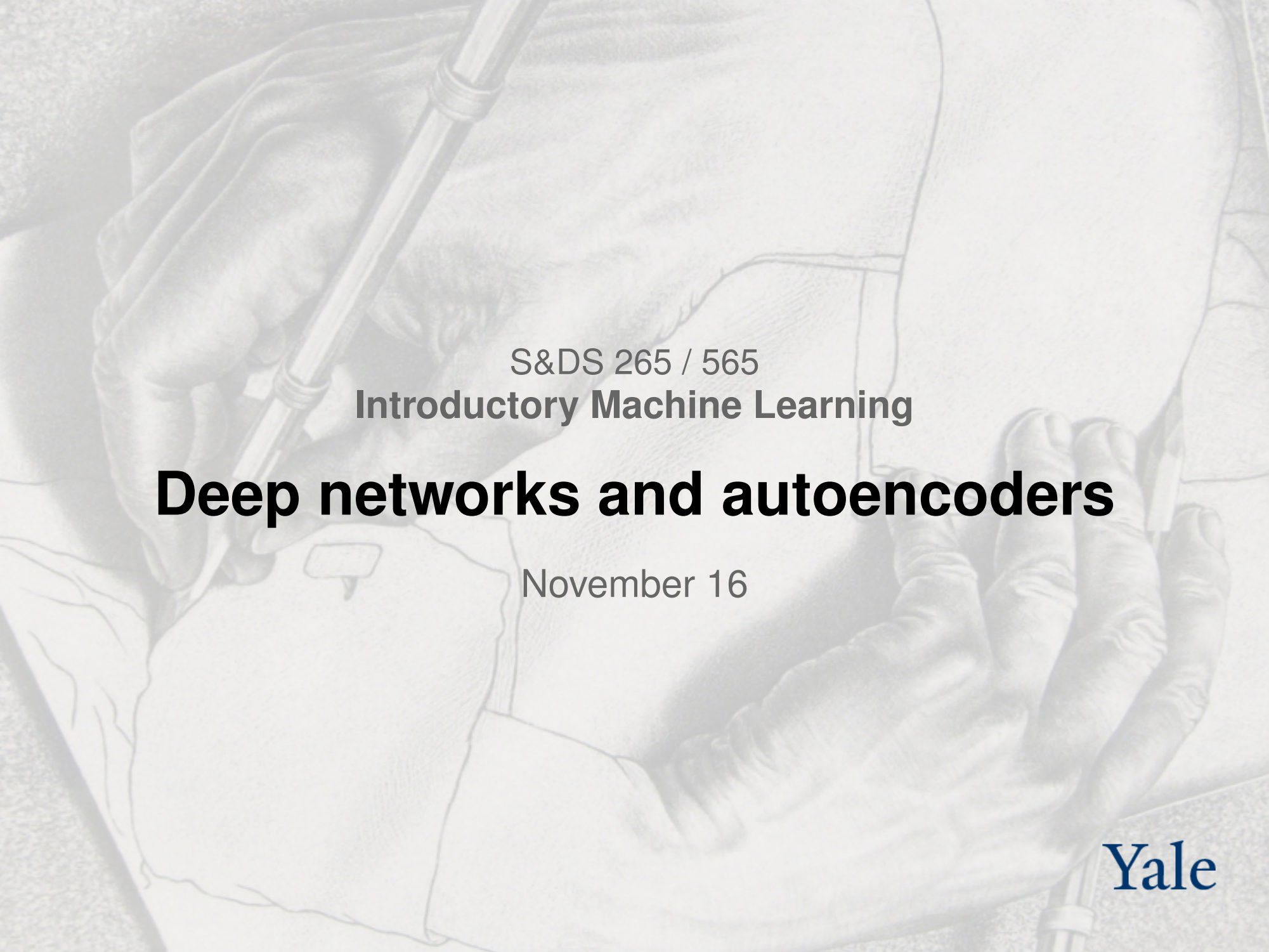


S&DS 265 / 565

Introductory Machine Learning

Deep networks and autoencoders

November 16

Yale

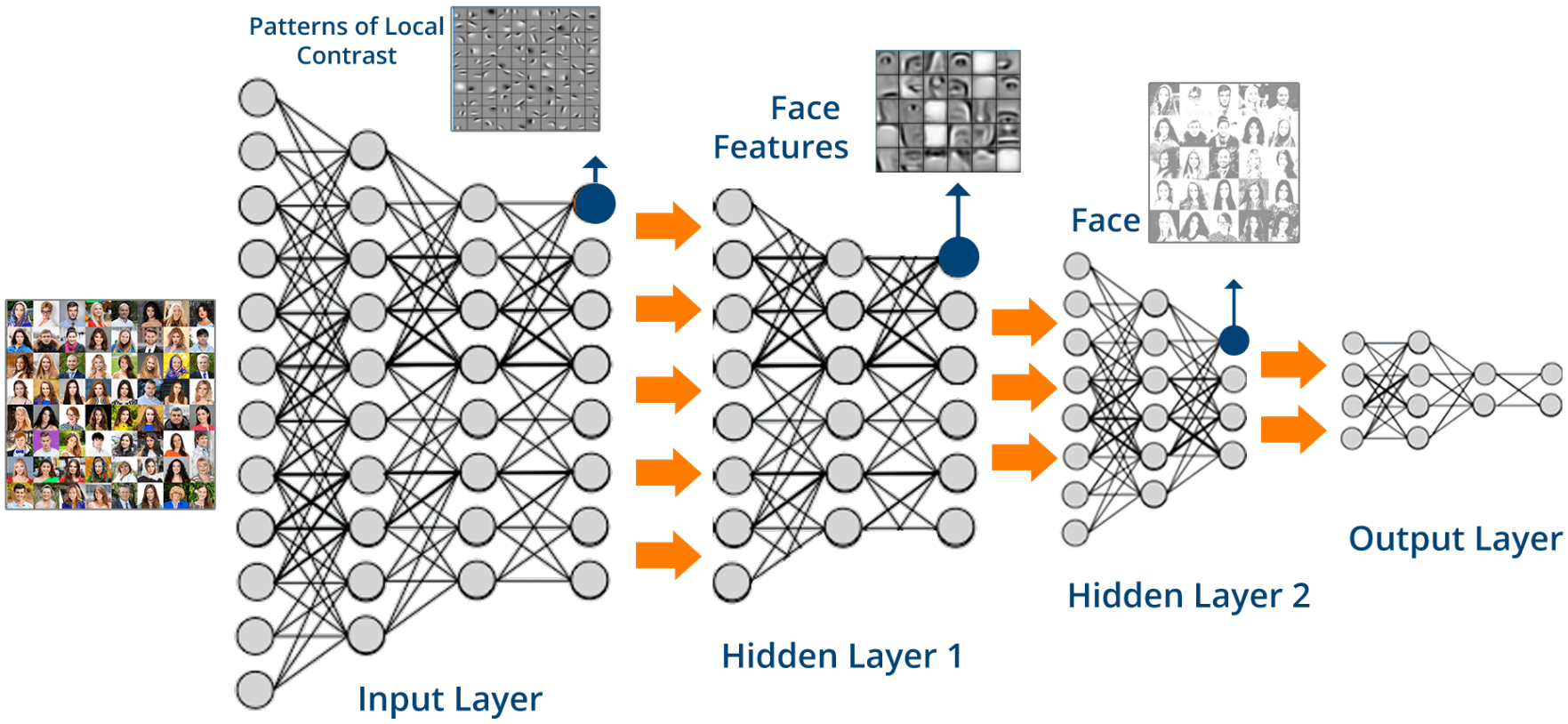
Reminders

- Quiz 5 available on Canvas
- Assignment 5 (last!) is out
- Final exam, Dec 15 at 2pm

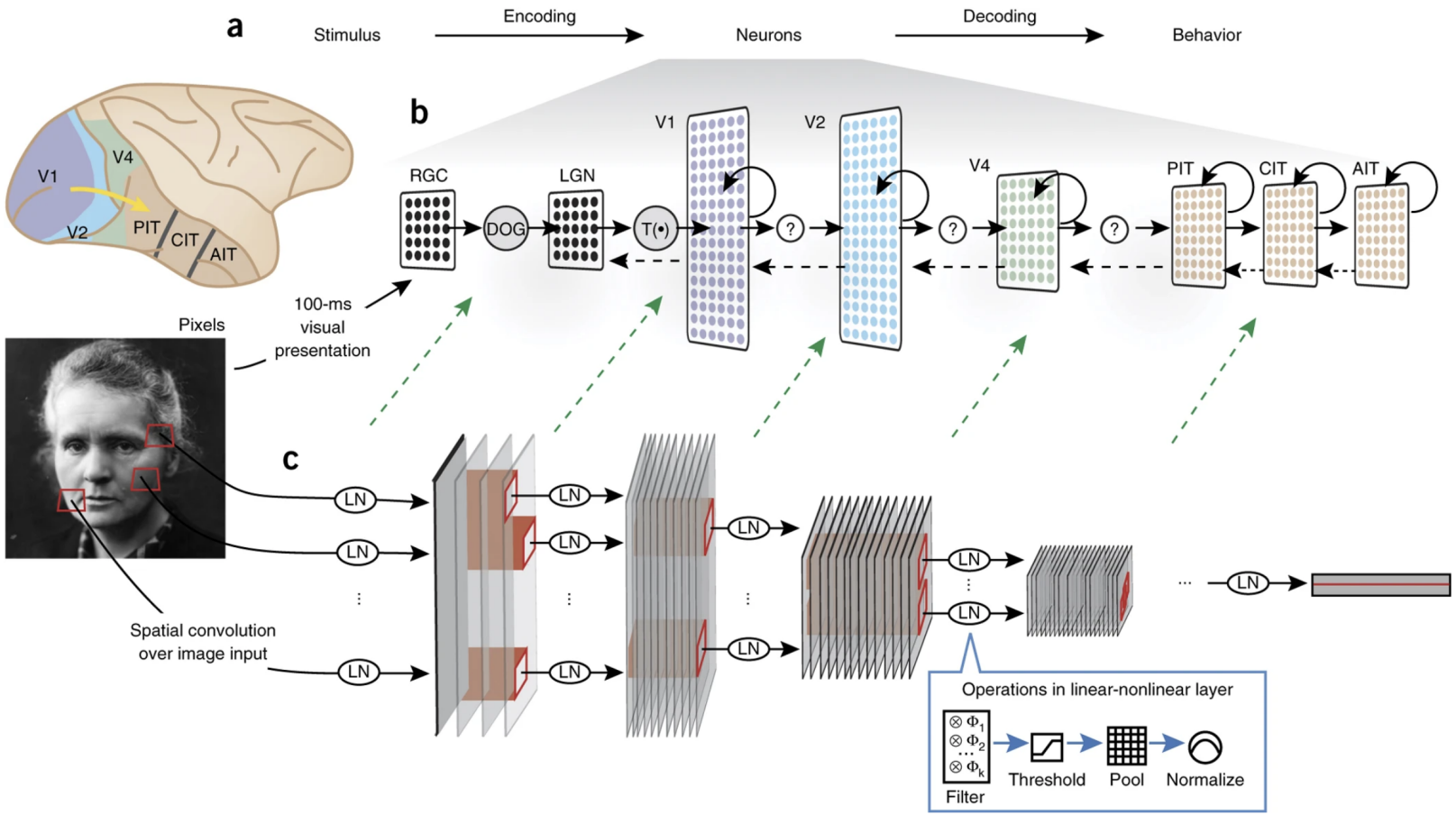
For today

- Deep networks
- Tensorflow playground
- Variants of autoencoders

Deep neural networks

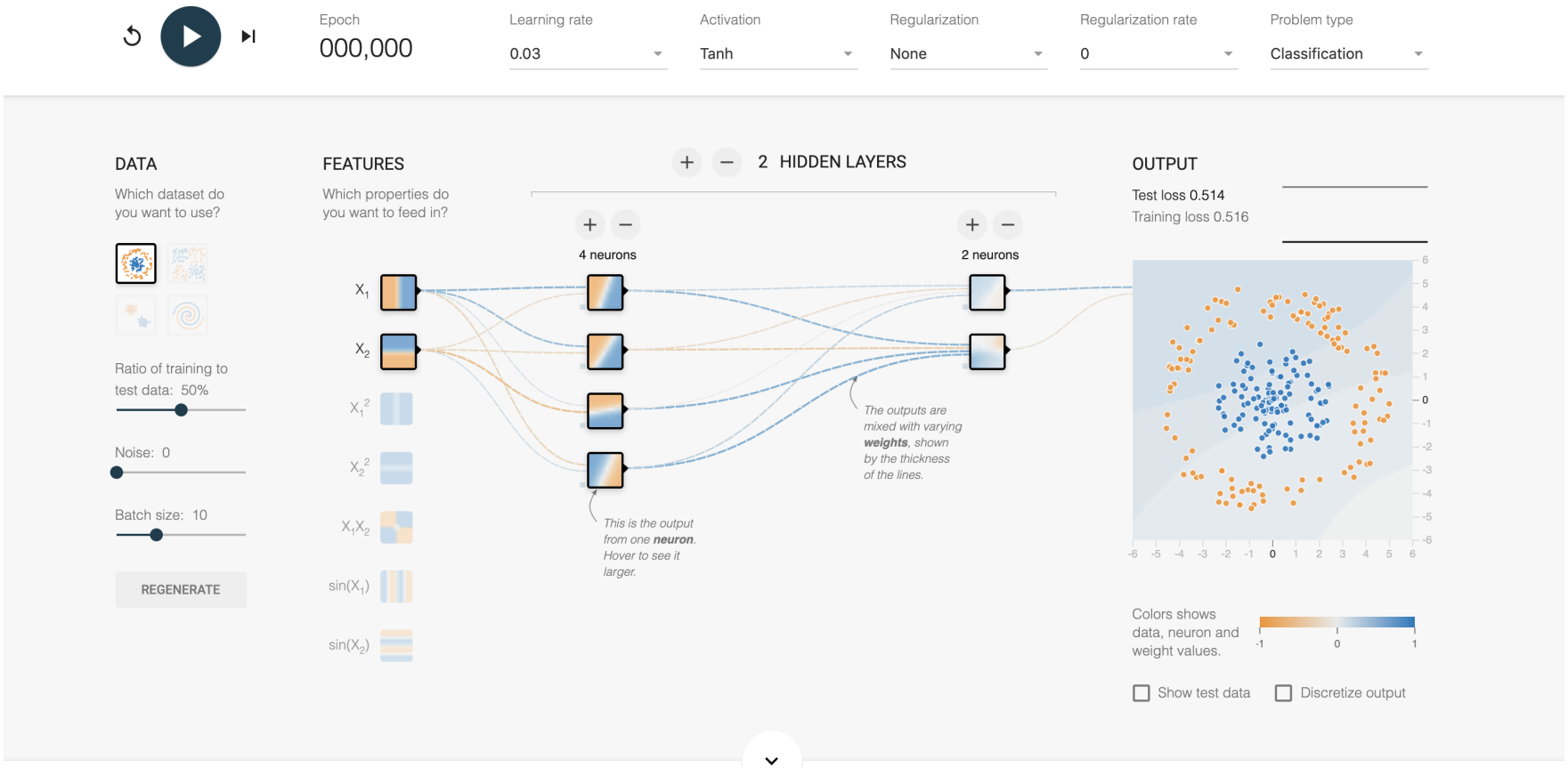


Deep neural networks



Using goal-driven deep learning models to understand sensory cortex, Yamins and DiCarlo, 2016,
https://www.nature.com/articles/nn.4244

Interactive visualizations



http://playground.tensorflow.org

It cuts both ways...

`numpy`

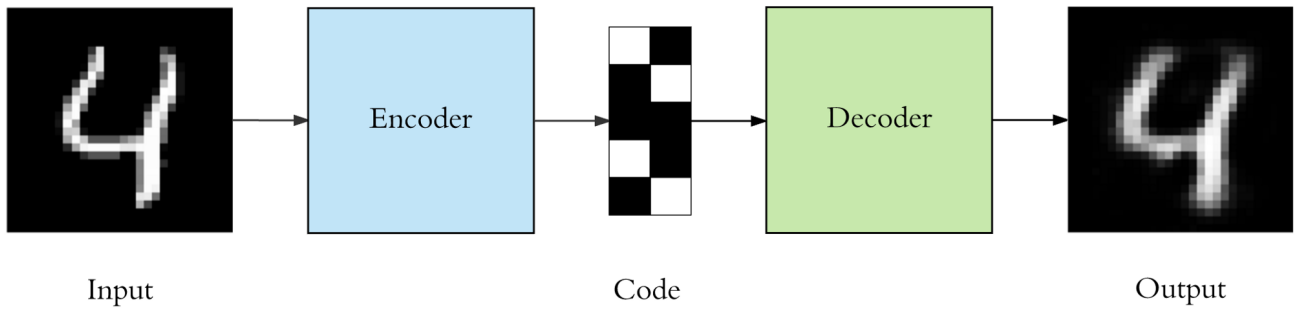
`keras`


THIS IS YOUR MACHINE LEARNING SYSTEM?
YUP! YOU POUR THE DATA INTO THIS BIG PILE OF LINEAR ALGEBRA, THEN COLLECT THE ANSWERS ON THE OTHER SIDE.
WHAT IF THE ANSWERS ARE WRONG?
JUST STIR THE PILE UNTIL THEY START LOOKING RIGHT.
DATA
ANSWERS

Autoencoders

- Unsupervised learning methods
- Squeeze high dimensional data through a “bottleneck” of lower dimension
- Train to minimize reconstruction error

Autoencoders

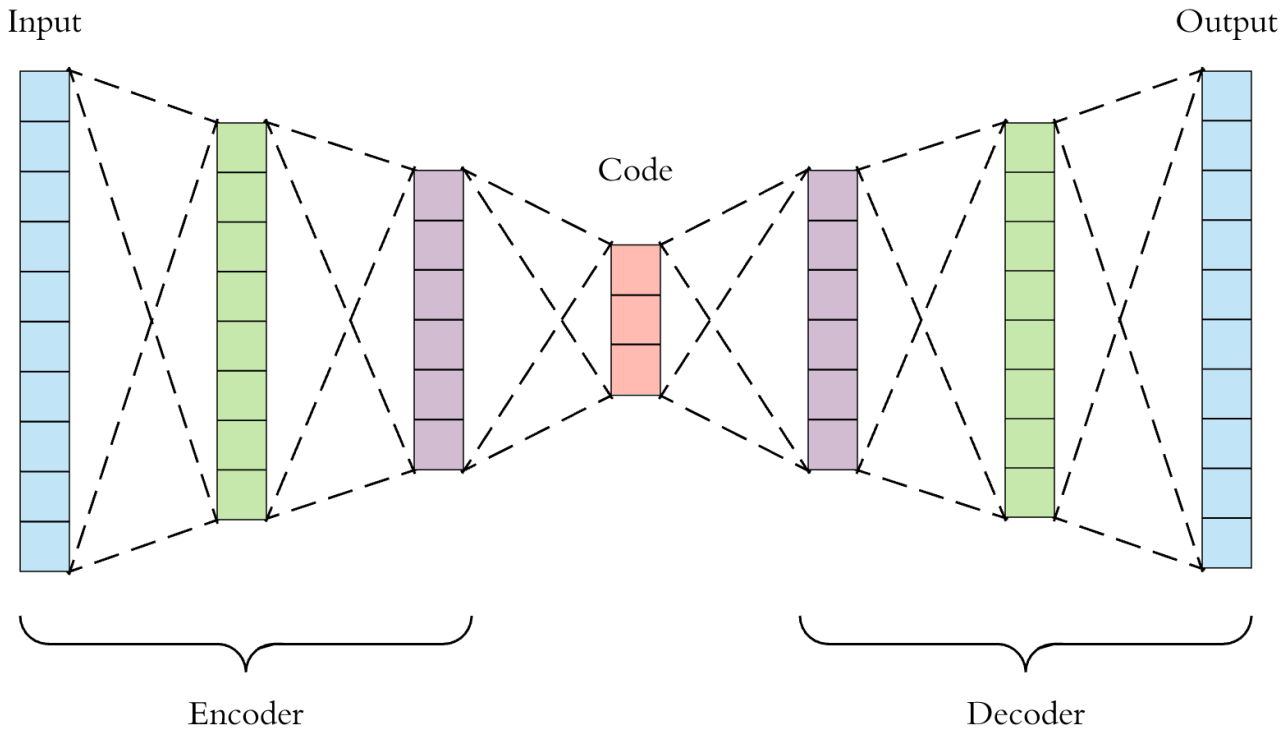


https://github.com/ardendertat/Applied-Deep-Learning-with-Keras

Important aspects

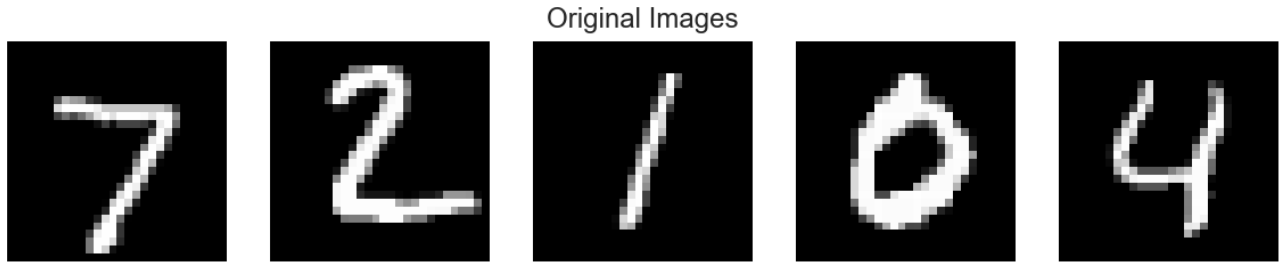
- Unsupervised: No labels used, discovers useful features of input
- Compression: Code reduces dimension of data
- Lossy: Input won't be reconstructed exactly
- Trained: The compression algorithm is learned for specific data

Deep architecture



https://github.com/ardendertat/Applied-Deep-Learning-with-Keras

MNIST data



$$28 \times 28 = 784 = D$$

Implementation using Keras

```
input_size = 784
hidden_size = 128
code_size = 32

input_img = Input(shape=(input_size,))
hidden_1 = Dense(hidden_size, activation='relu')(input_img)
code = Dense(code_size, activation='relu')(hidden_1)
hidden_2 = Dense(hidden_size, activation='relu')(code)
output_img = Dense(input_size, activation='sigmoid')(hidden_2)

autoencoder = Model(input_img, output_img)
autoencoder.compile(optimizer='adam', loss='binary_crossentropy')
autoencoder.fit(x_train, x_train, epochs=5)
```

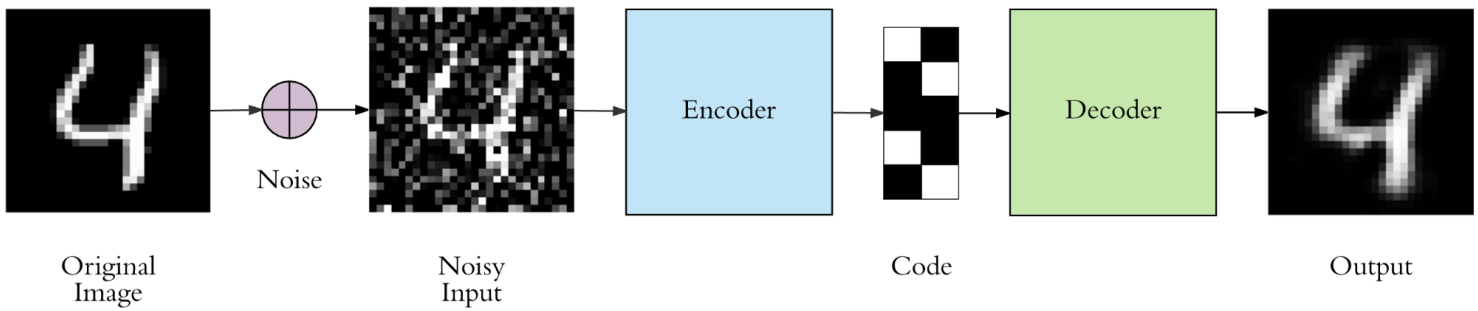
Adam optimizer

- Variant of stochastic gradient descent where separate learning rate (step size) is maintained for each network weight (parameter)
- Each step size adapted as learning progresses based on moments of the derivatives

---

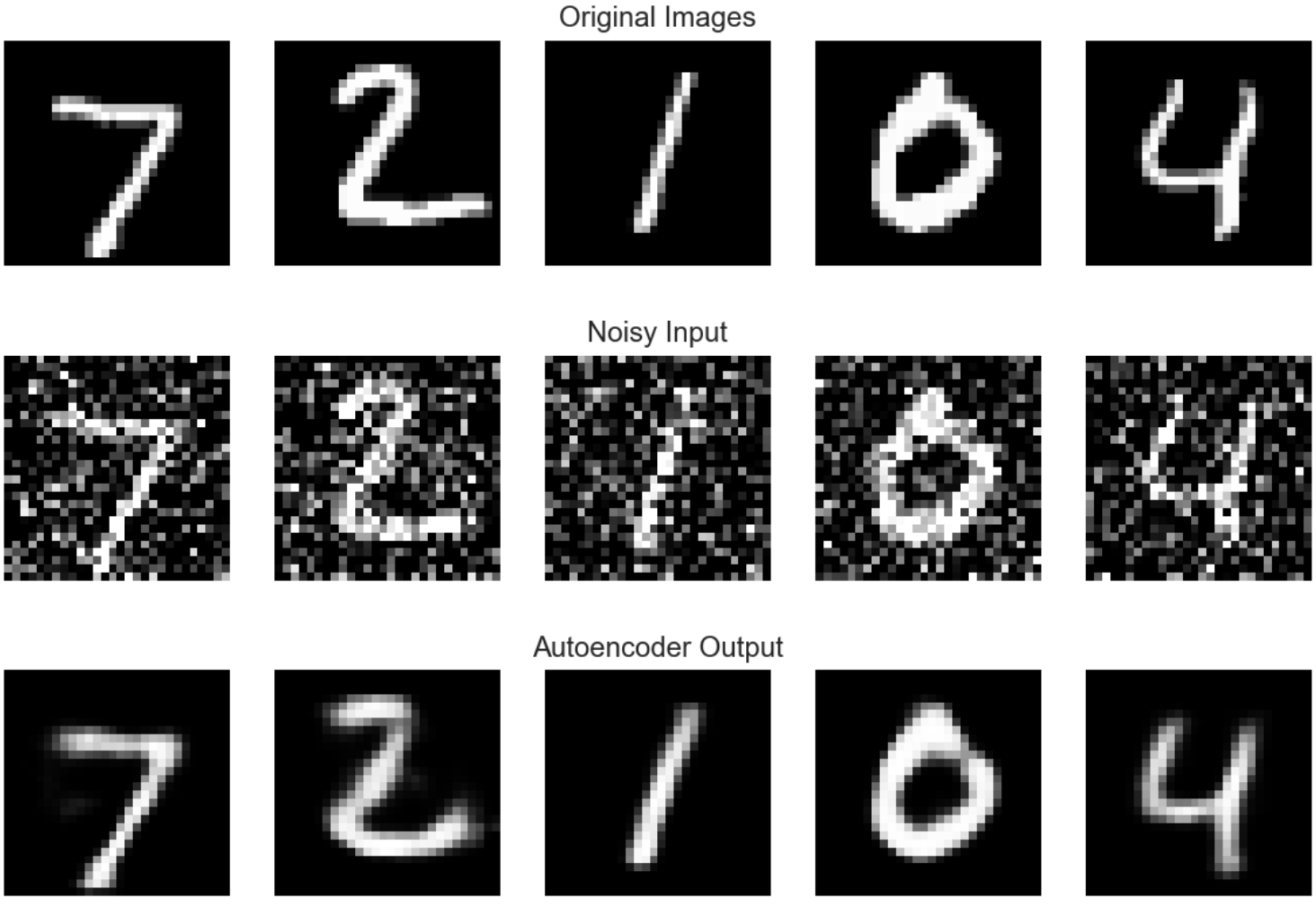
"Adam: A method for stochastic optimization," D. Kingma and J. Ba, https://arxiv.org/abs/1412.6980

Variant: Denoising autoencoder



- Feed in noisy data
- Train to match to denoised data

Example result on MNIST



Summary: What did we learn today?

- Tensorflow/keras is a power tool (Assignment 5)
- Autoencoders compress the input and then reconstruct it
- Bottleneck forces extraction of useful features
- Will overfit and “memorize” the data
- Overfitting mitigated by denoising autoencoders