CIS 365 Artificial Intelligence

Autoencoders

Dr. Denton Bobeldyk

Learning Features from an Image

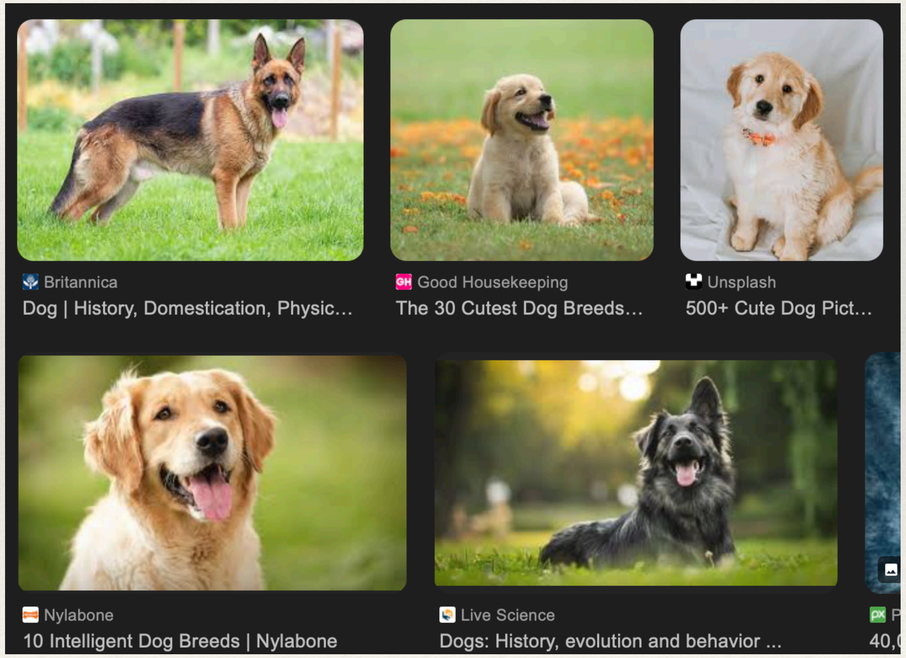
- Manually define a feature
- Determine how to extract the feature from an image
- For example, define a feature that detects texture:
  - Binarized Statistical Image Features (BSIF)
  - Local Binary Pattern (LBP)

Hand Crafted Features

- Machine interpretable features:
  - Binarized Statistical Image Features (BSIF)
  - Local Binary Pattern (LBP)
  - Scale Invariant Feature Transform (SIFT)
- Human interpretable features:
  - Does the object have wings?
  - Does the object have legs?

Hand Crafted Features Individual Exercise

Determine 4 human interpretable features to discriminate between the following two classes:



Source: Screen capture for a search on google

Representation Learning

- Allow the system to learn ‘representations’ that we can later use to classify
- The representations that are learned are, for the most part, machine interpretable and not human understandable

Representation Learning - CNN Example

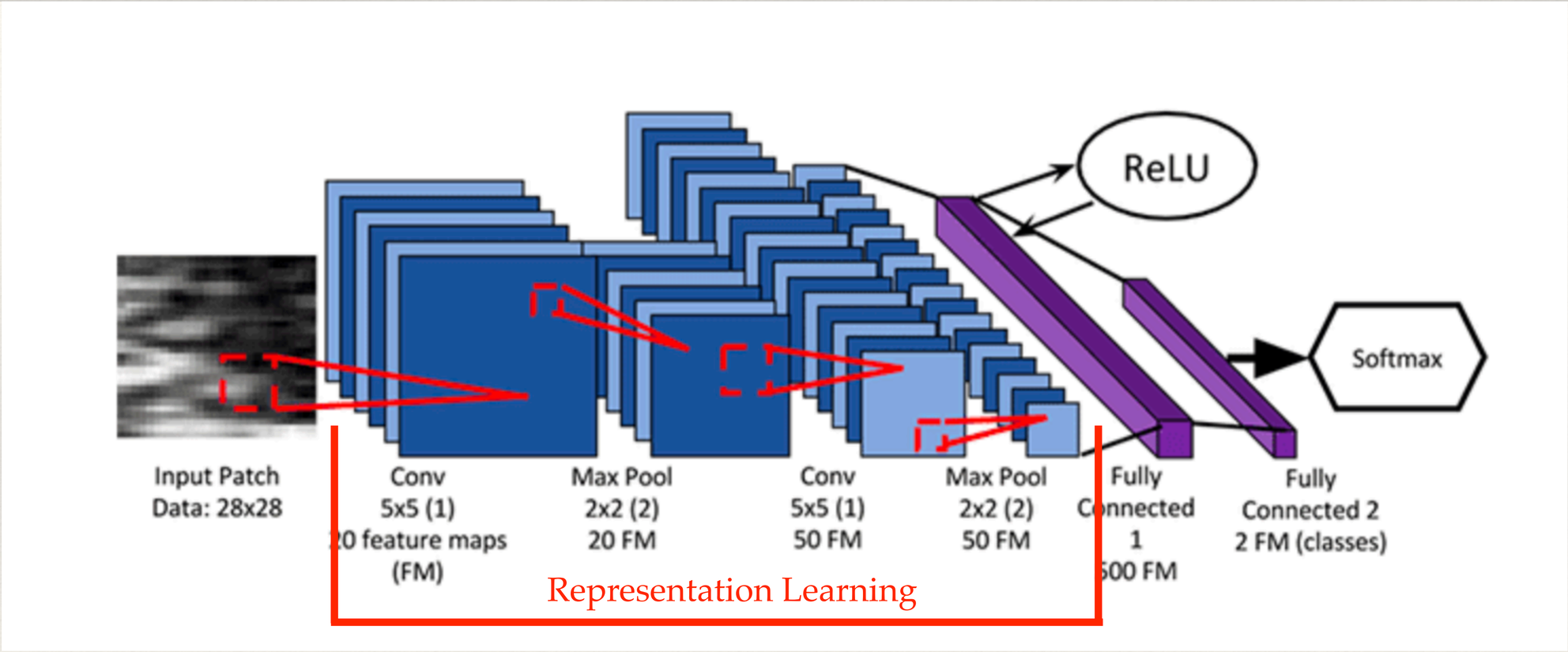


Image sourced from: https://pyimagesearch.com/2021/07/19/pytorch-training-your-first-convolutional-neural-network-cnn/

Representation Learning - CNN Example

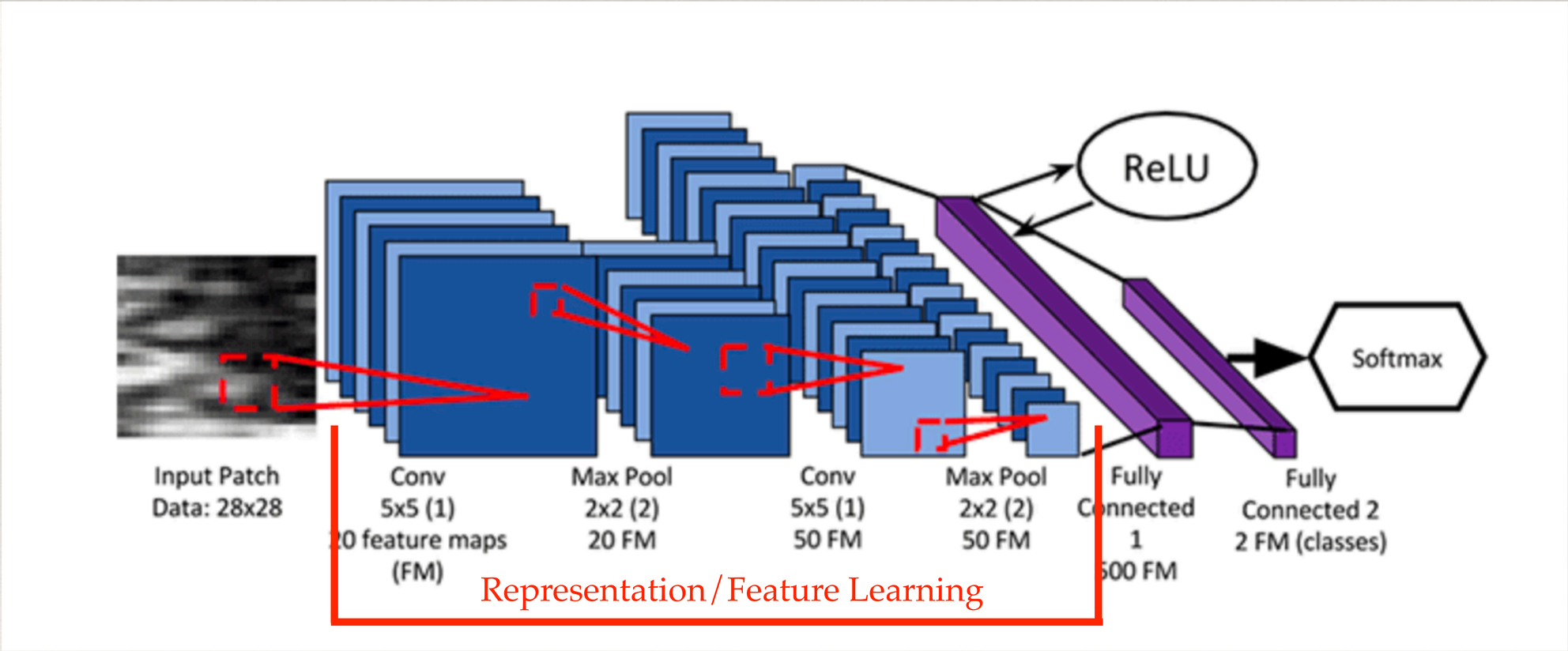
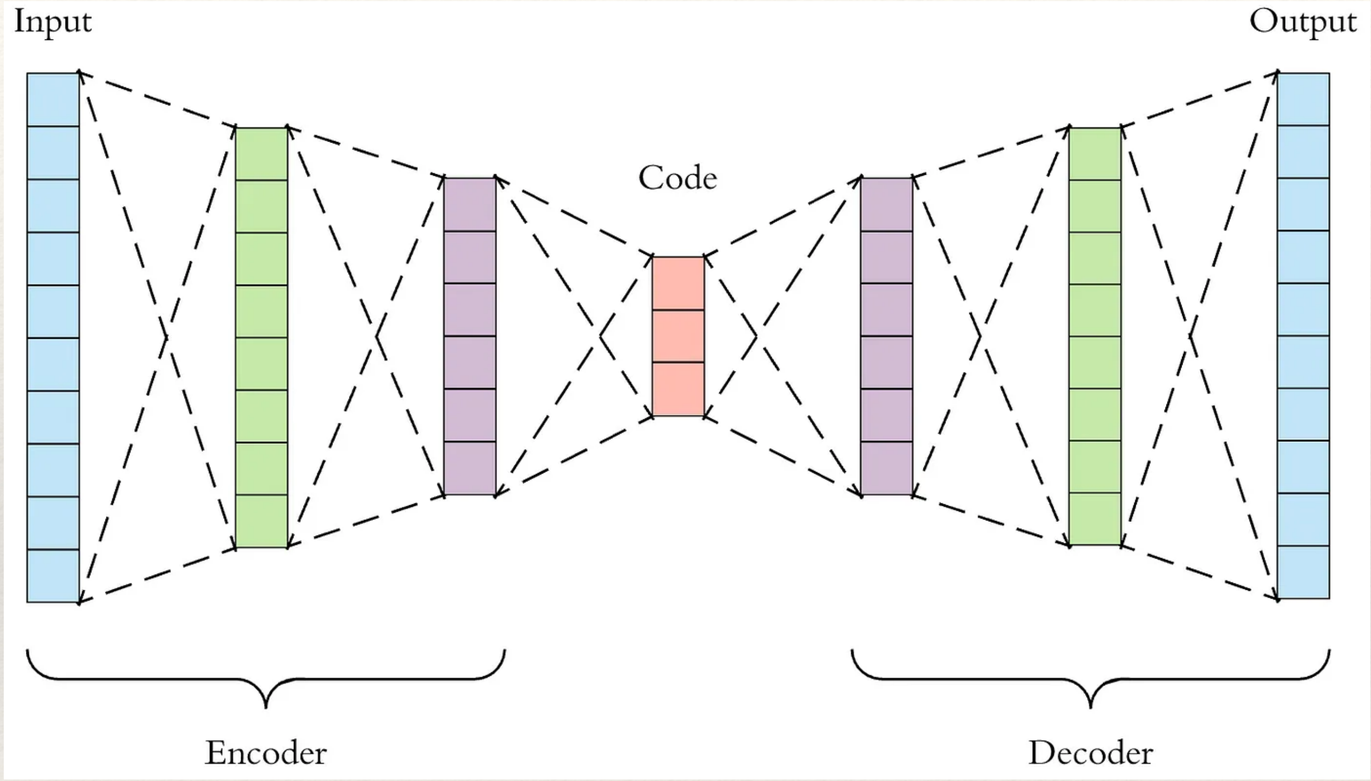


Image sourced from: https://pyimagesearch.com/2021/07/19/pytorch-training-your-first-convolutional-neural-network-cnn/

Representation Learning

Are there other ways we can force a system to learn representations?

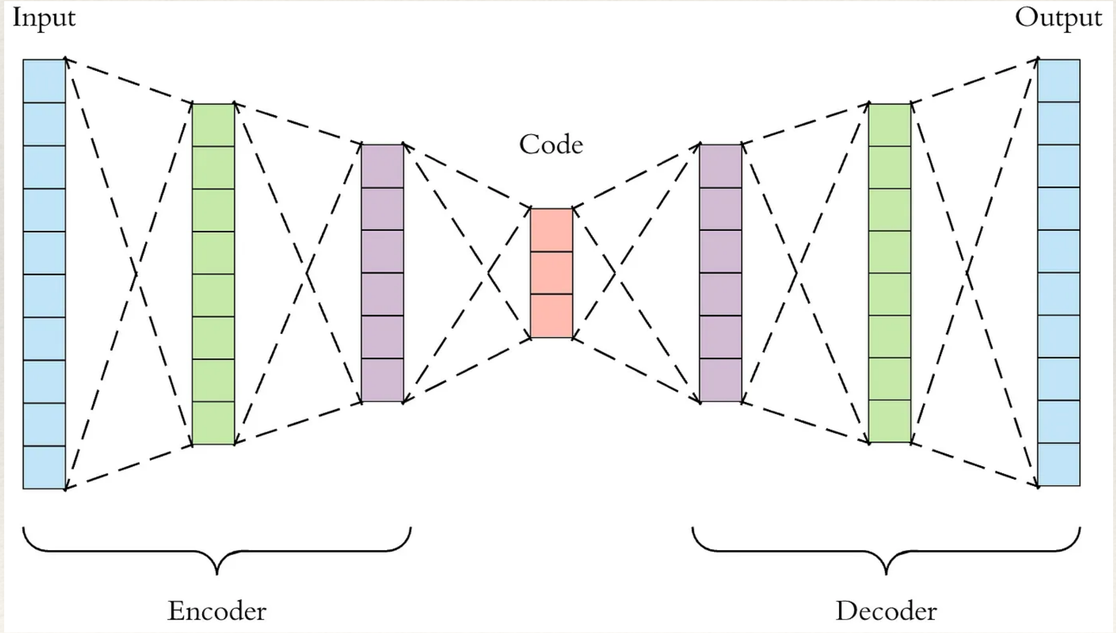
Autoencoder



https://towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798

Autoencoder

Encoder: Compresses the input into a latent space representation
Decoder: Reconstruct the input from the latent space ‘code’
Code: Latent space learned from the network

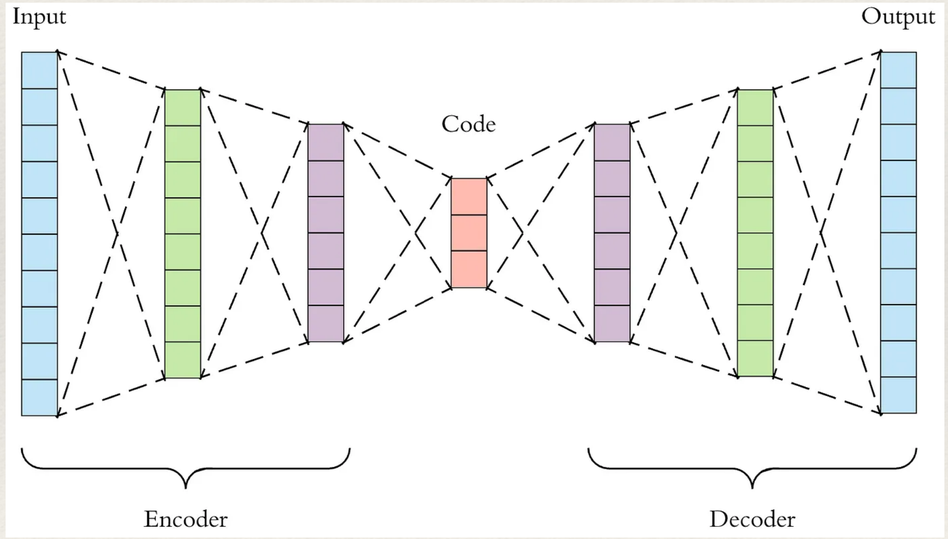


https://towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798

Autoencoder

Encoder: Compresses the input into a latent space representation
Decoder: Reconstruct the input from the latent space ‘code’
Code: Latent space learned from the network

Input Image



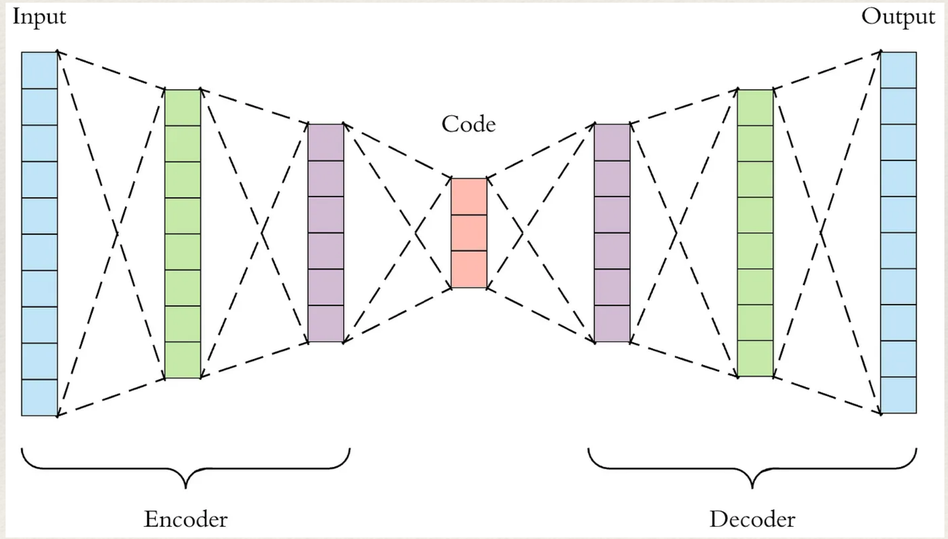
Reconstructed Input Image

https://towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798

Autoencoder

Encoder: Compresses the input into a latent space representation
Decoder: Reconstruct the input from the latent space ‘code’
Code: Latent space learned from the network

Input Image

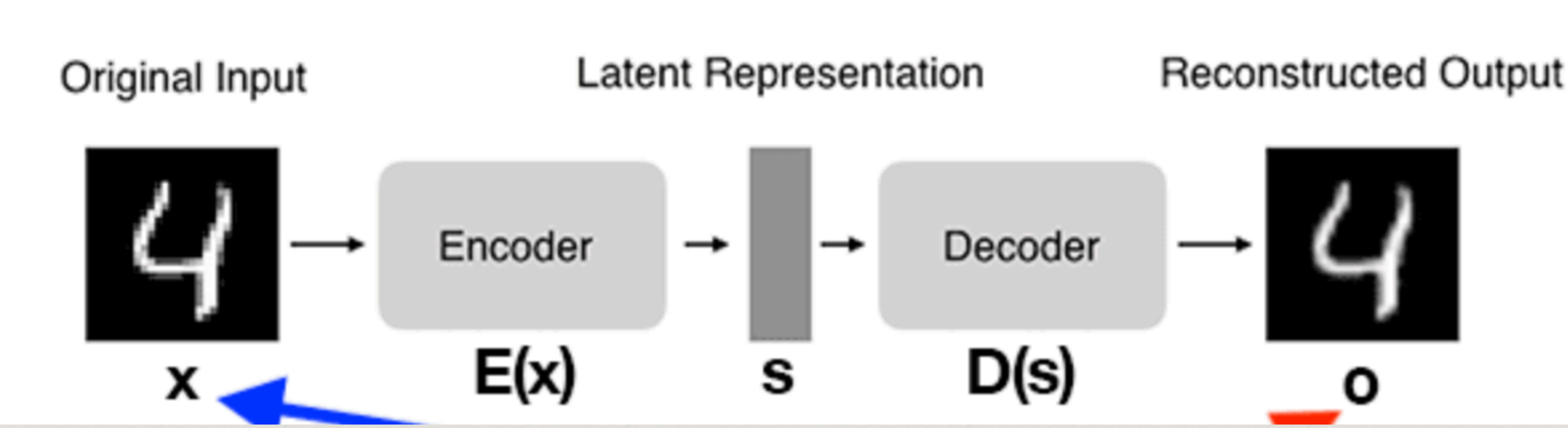


Reconstructed Input Image

Loss function: Minimize the difference between input image and image reconstruction

https://towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798

Autoencoder



https://pyimagesearch.com/2020/02/17/autoencoders-with-keras-tensorflow-and-deep-learning/

Autoencoder Feature Learning

- Through the process of reducing the input data into a lower dimensional space, the auto encoder learns to capture the most important features of the data.
- The bottleneck forces a learned compact representation of these features that is later expanded in the decode layer.

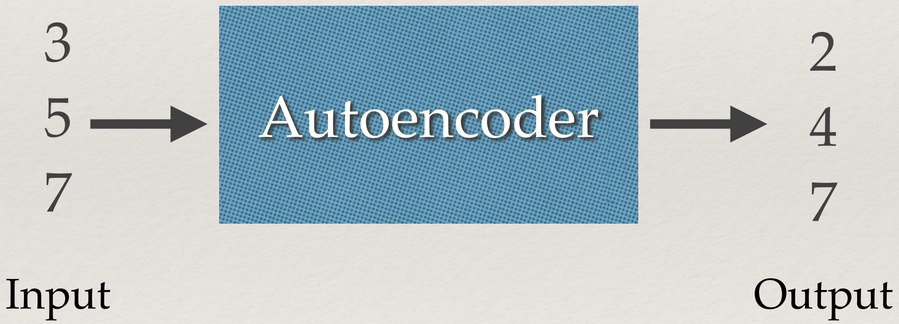
Autoencoder Loss Function

MSE (Mean Squared Error): $\frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x}_i)^2$

Binary Cross Entropy Loss: $-\frac{1}{n}[x_i log(\hat{x}_i) + (1 - x_i) log(1 - \hat{x}_i)]$

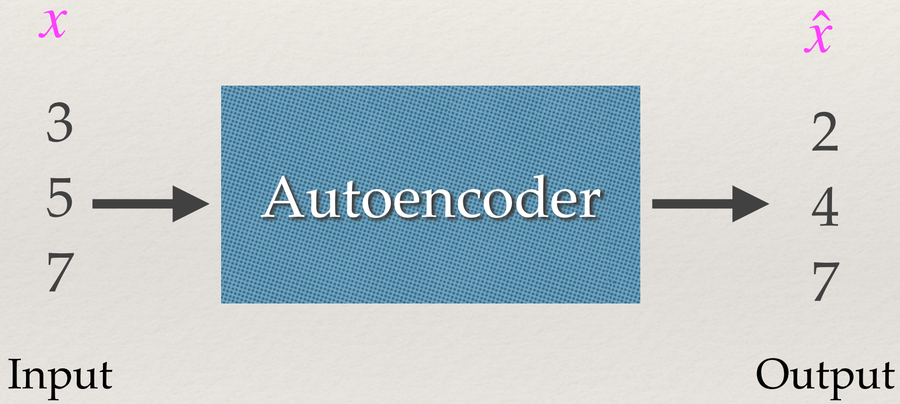
Autoencoder Loss Function

MSE (Mean Squared Error): $\frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x}_i)^2$



Autoencoder Loss Function

MSE (Mean Squared Error): $\frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x}_i)^2$

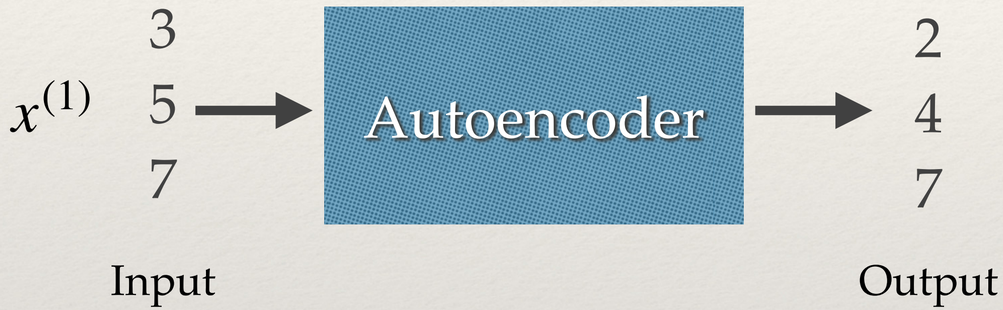


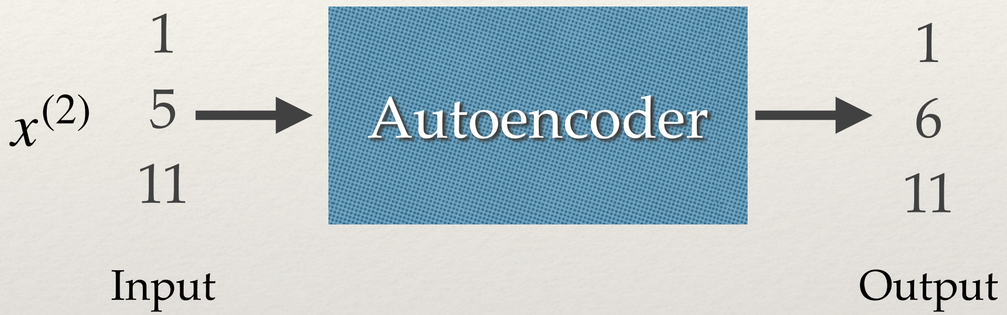
$$\text{Loss} = (x_1 - \hat{x}_1)^2 + (x_2 - \hat{x}_2)^2 + (x_3 - \hat{x}_3)^2$$

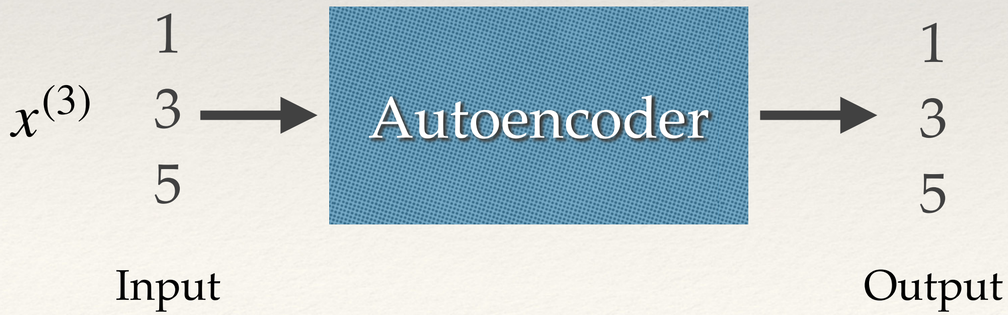
$$\text{Loss} = (3 - 2)^2 + (5 - 4)^2 + (7 - 7)^2$$

Autoencoder Loss Function

Calculate the error for all the samples and sum them together



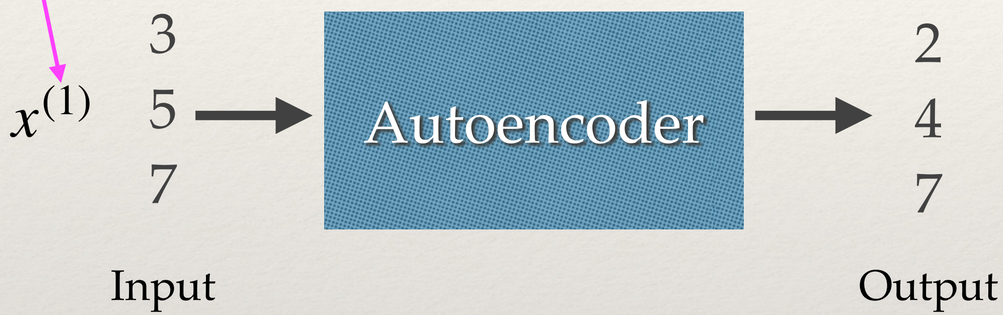


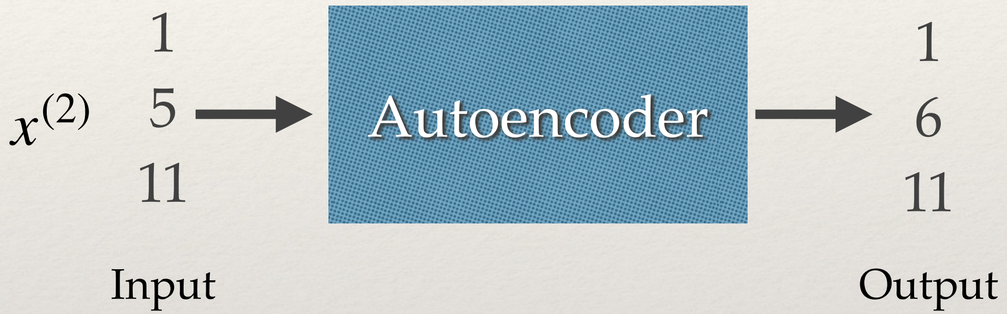


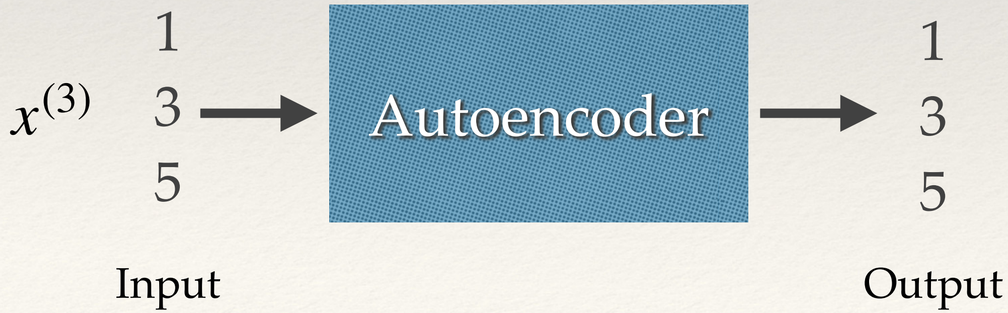
Autoencoder Loss Function

Calculate the error for all the samples and sum them together

The superscript represents the sample number







More complex problems

Images generated by ChatGPT

What do all of these images have in common?

More complex problems

Images generated by ChatGPT

What do all of these images have in common?

Is there a ‘distribution’ that they are being drawn from?

More complex problems

Images generated by ChatGPT

Is there a good way to measure the distances between the input and output for images like these?

Cost Functions

- Can measure the distance between probability distributions
  - The input distribution
  - The output distribution
- Kullback-Leibler Divergence measures the difference between the two
- Binary Cross Entropy Loss measures the difference between the pixel values

Cost Functions

- Can measure the distance between probability distributions
  - The input distribution
  - The output distribution
- Kullback-Leibler Divergence measures the difference between the two
- Binary Cross Entropy Loss measures the difference between the pixel values

Binary Cross Entropy Loss

- Binary Cross Entropy Loss is a common cost function to measure the difference between the input and the output image

$$\text{BCE} = -\frac{1}{N}\sum_{i=1}^{N}\left(x_i \cdot \log(\hat{x}_i) + (1 - x_i) \cdot \log(1 - \hat{x}_i)\right)$$

N = total number of features

x_i = original value of the feature

$\hat{x}_i$ = reconstructed value

Binary Cross Entropy Loss

$$\text{BCE} = -\frac{1}{N}\sum_{i=1}^{N}\left(x_i \cdot \log(\hat{x}_i) + (1 - x_i) \cdot \log(1 - \hat{x}_i)\right)$$

Key aspects of binary cross entropy are:

- Penalizes incorrect predictions more heavily, with a higher loss for predictions that are confidently wrong
- For correct predictions, the loss approaches 0 as the predicted probability approaches the actual label

Autoencoder Feature Understandability

- The features learned are ‘machine understandable’ and can be difficult to translate into ‘human understandable’ features (if translated at all)
- Research efforts are being made to help visualize the features that are captured

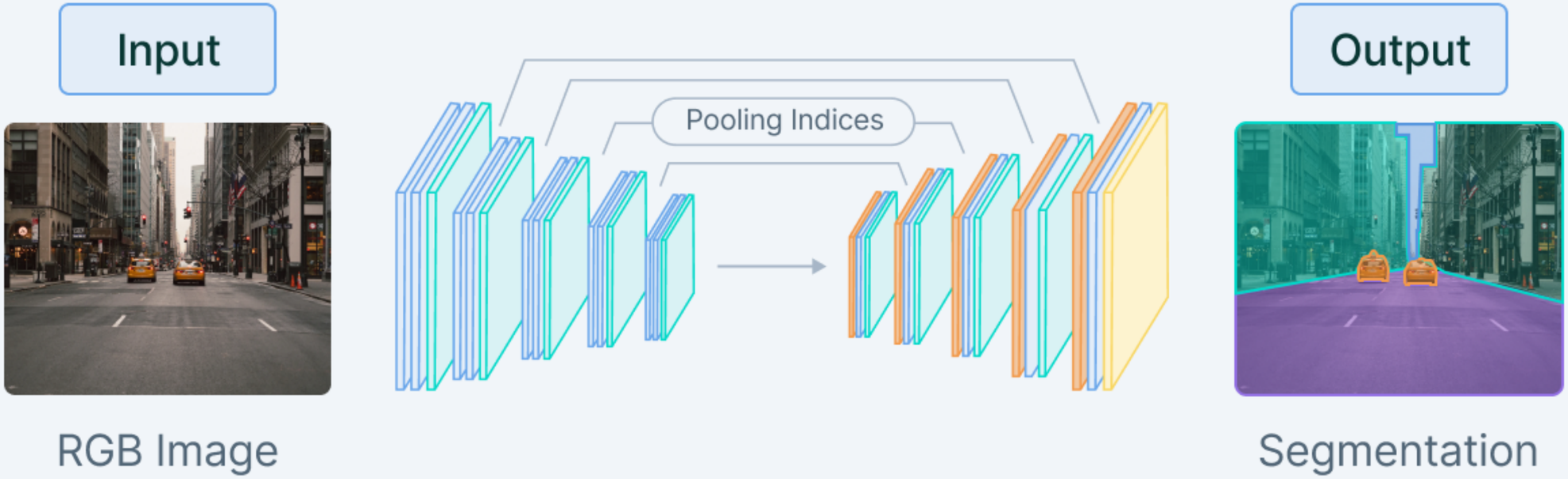
Autoencoder Applications

What applications might these be useful for?

Autoencoder Applications

- Anomaly Detection
- Data Compression
- Feature Learning
- Dimensionality Reduction
- Noise Removal
- Data/Image Generation (via variational autoencoders)

Convolutional encoder-decoder



https://www.v7labs.com/blog/autoencoders-guide

Autoencoder Challenges

- Difficulty in capturing complex distributions
- Limitation in generating high-quality images (compared to GANs)
- Overfitting when the latent space is too large

https://www.v7labs.com/blog/autoencoders-guide

Autoencoder Implementation

- Keras Library
  - Easy to use autoencoder library
  - Excellent documentation
  - Adopted by the masses

Autoencoder Implementation

- Keras Library
  - Built on Tensorflow, up until now, we've been using PyTorch for most class assignments

Image created from chatGPT4

Keras Documentation

Documentation:
https://keras.io/api/

Examples:
https://keras.io/examples/

Implementing an Autoencoder

```
# This is our input image
input_img = keras.Input(shape=(784,))

# "encoded" is the encoded representation of the input
encoded = layers.Dense(encoding_dim, activation='relu')(input_img)

# "decoded" is the lossy reconstruction of the input
decoded = layers.Dense(784, activation='sigmoid')(encoded)

# This model maps an input to its reconstruction
autoencoder = keras.Model(input_img, decoded)
```

Implementing an Autoencoder

```
# This is our input image
input_img = keras.Input(shape=(784,))
```

Defines the input tensor and that each image will be flattened into a 784 dimensional vector
Input is 28x28 pixels, so a common MNIST example

Implementing an Autoencoder

```
# "encoded" is the encoded representation of the input
encoded = layers.Dense(encoding_dim, activation='relu')(input_img)
```

Encoder layer is responsible for generating the encoded representation of the data.

The above takes input_img as the input, uses the variable 'encoding_dim' to specify what dimension the data will be reduced to in the layer. The activation is relu.

Implementing an Autoencoder

```
# "decoded" is the lossy reconstruction of the input
decoded = layers.Dense(784, activation='sigmoid')(encoded)
```

Decoder layer is responsible for reconstructing the image from the encoded representation.

784 is the output and corresponds to the dimensionality of the input (the 28x28 image flattened)

Activation: sigmoid is used to ensure the values are between 0 and 1 (the normalized pixel value
of the original input images)

Implementing an Autoencoder

```
# This model maps an input to its reconstruction
autoencoder = keras.Model(input_img, decoded)

# This model maps an input to its encoded representation
encoder = keras.Model(input_img, encoded)
```

Creates the autoencoder and encoder model that allows us to call built in functions from the Keras API

Implementing an Autoencoder

```
autoencoder.compile(optimizer='adam', loss='binary_crossentropy')
```

Sets the optimizer to adam (a popular optimization algorithm)
Sets the loss function to binary cross entropy

Implementing an Autoencoder

```
autoencoder.fit(x_train, x_train,
                epochs=50,
                batch_size=256,
                shuffle=True,
                validation_data=(x_test, x_test))
```

Defines the input and output data, in this case they are the same.
Defines the epochs (the number of times it will run through the entire training dataset) to be 50
Batch size is the number of samples that will propagate through the network before the weights will be updated
Shuffle indicates that the data will be shuffled around each epoch so the model won't see the data in the same order each time.
Validation_data is data the model can be evaluated on, but won't be trained on.

Keras Documentation

Building an autoencoder in Keras
https://blog.keras.io/building-autoencoders-in-keras.html

Related Research Articles

Reducing the Dimensionality of
Data with Neural Networks
Hinton/Salakhutdinov

https://www.cs.toronto.edu/~hinton/absps/science.pdf

Related Research Articles

Extracting and Composing Robust Features with Denoising Autoencoders

https://www.cs.toronto.edu/~larocheh/publications/icml-2008-denoising-autoencoders.pdf

AI Current Events

https://www.pcmag.com/news/musks-xai-supercomputer-goes-online-with-100000-nvidia-gpus

AI Current Events

https://www.nbcnews.com/tech/tech-news/openai-considering-restructuring-profit-cto-mira-murati-two-top-resear-rcna172815

End