Deep AutoEncoder

CSE/ECE 577

Beibin Li

Nov/1/2021

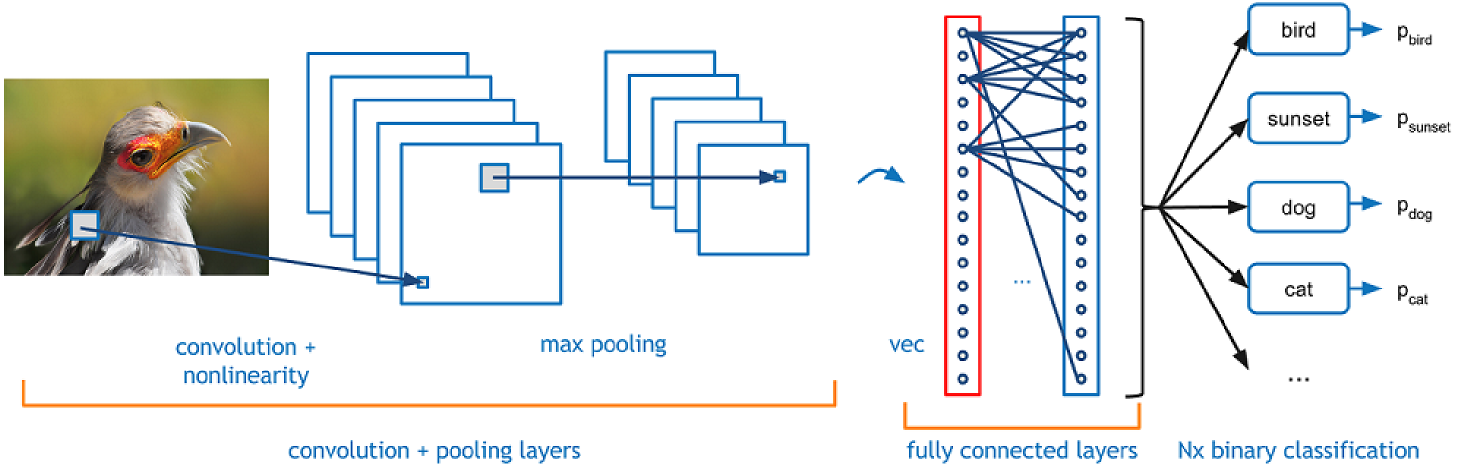
PAUL G. ALLEN SCHOOL

Summary

- Convolutional Neural Network
  - Learning Convolutions
  - CNN for Classification
  - CNN for Segmentation
- Autoencoder (AE)
  - Birth of Autoencoder
  - U-Net
  - PyTorch Implementation in HW 2
- Other Applications for Autoencoder
- Neural Encoding and Decoding with Deep Learning for Dynamic Natural Vision

Convolution Neural Network

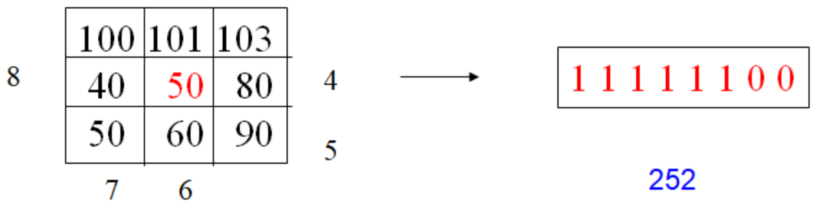
CNN



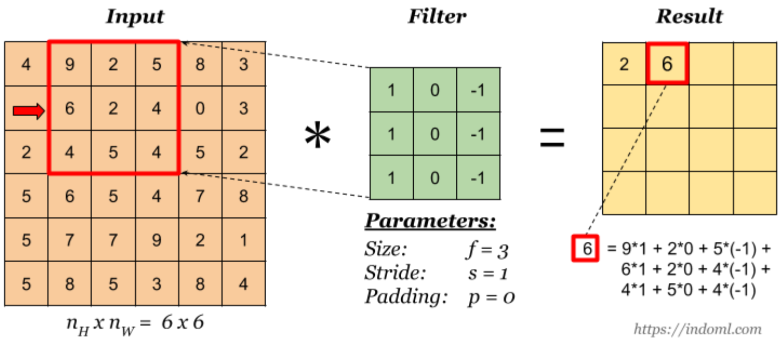
convolution + nonlinearity
max pooling
vec
bird
sunset
dog
cat
...
p bird
p sunset
p dog
p cat
convolution + pooling layers
fully connected layers
Nx binary classification

Convolution: from fixed to learnable

LBP



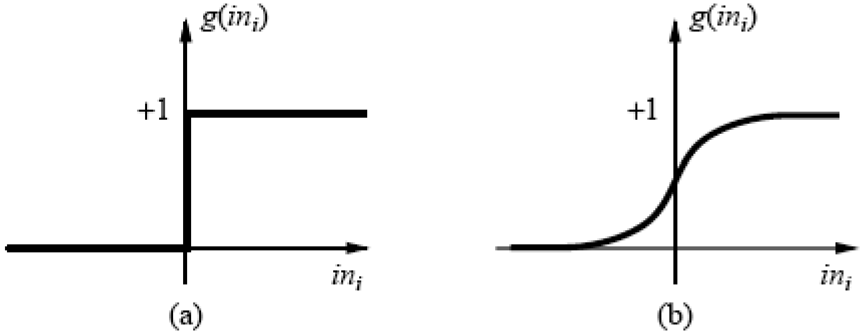
Convolution in CNN



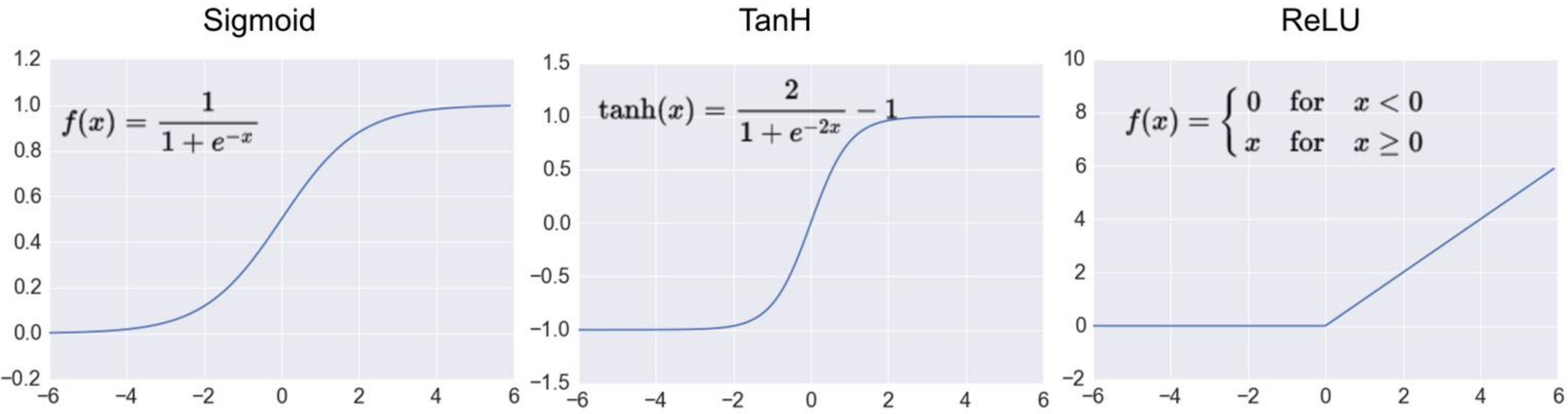
Expert-Designed Convolution: SIFT, HoG, LBP, ...

Learn Flexible Parameters

FC Activation: From Step Function to Sigmoid

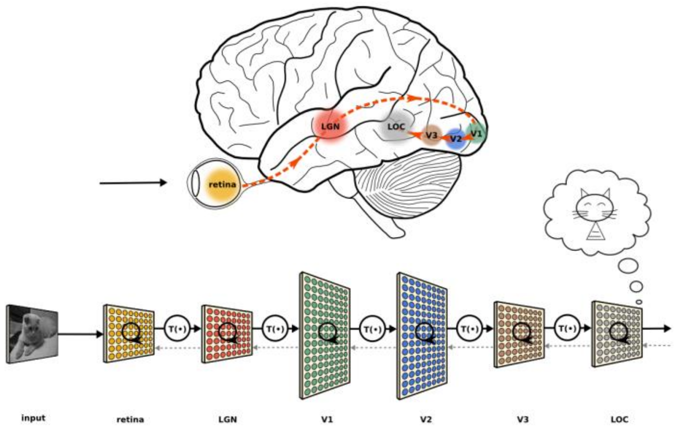


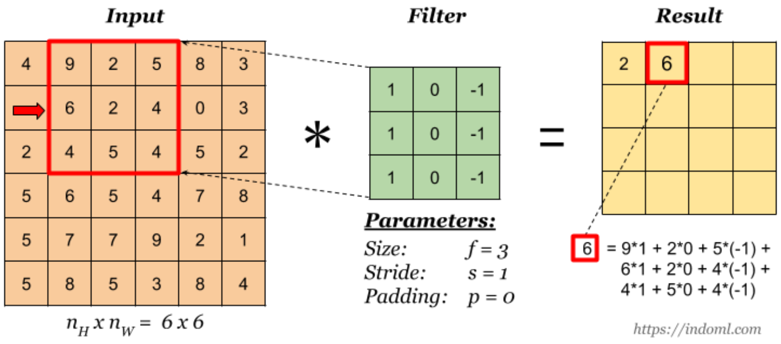
CNN Activation: From Sigmoid to ReLU



Representation

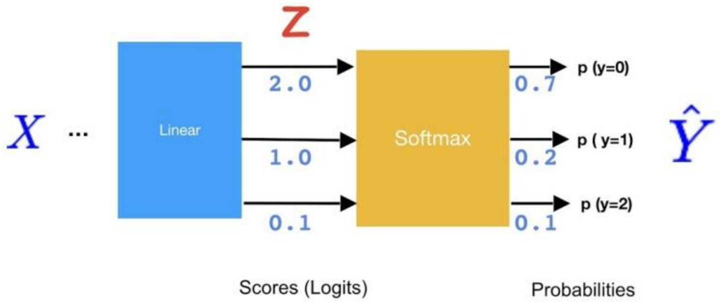
Input = (3, width, height)



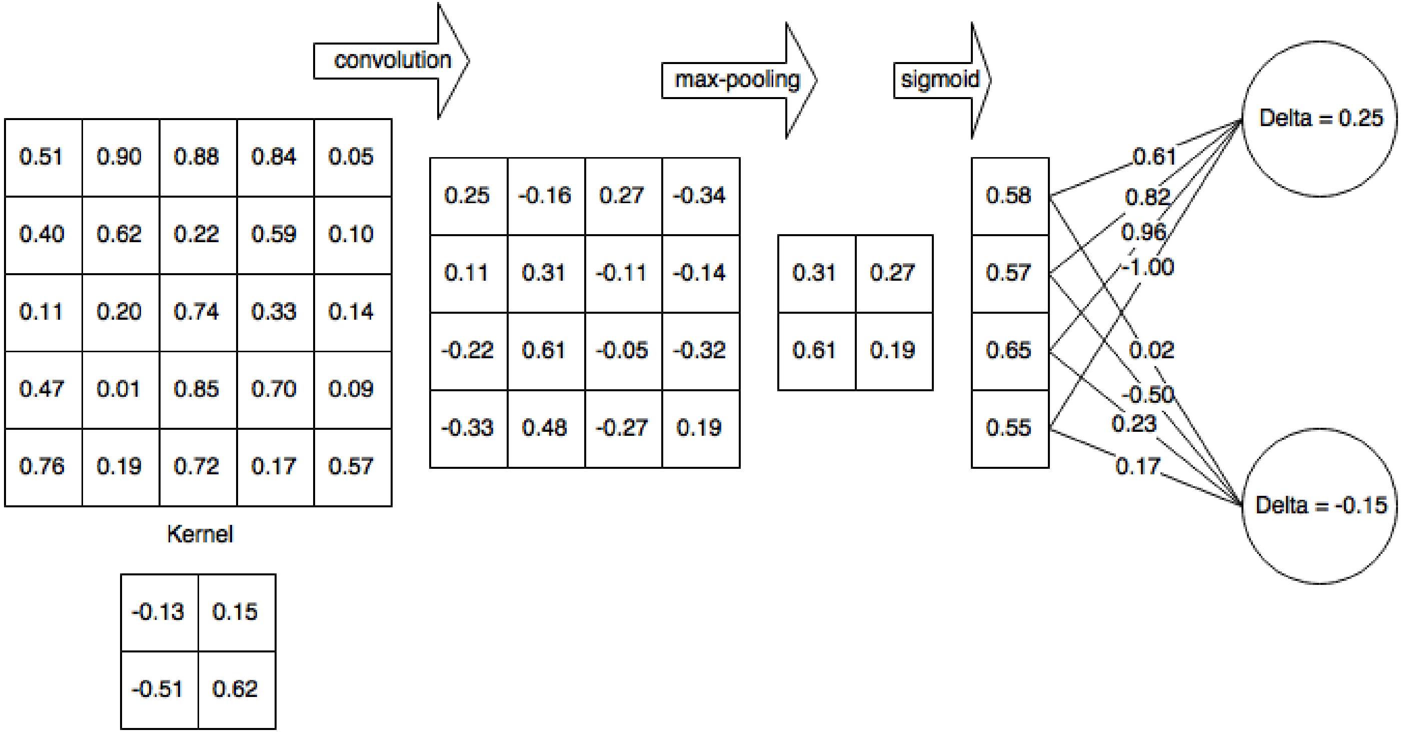


Meet Softmax

$$\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \quad \text{for } j = 1, \ldots, K.$$



Inference



https://datascience.stackexchange.com/questions/27506/back-propagation-in-cnn

Learning

$$\begin{bmatrix} O_{11} & O_{12} \\ O_{21} & O_{22} \end{bmatrix} = \textbf{Convolution}\left(\begin{bmatrix} X_{11} & X_{12} & X_{13} \\ X_{21} & X_{22} & X_{23} \\ X_{31} & X_{32} & X_{33} \end{bmatrix}, \begin{bmatrix} F_{11} & F_{12} \\ F_{21} & F_{22} \end{bmatrix} \right)$$

$$O_{11} = F_{11}X_{11} + F_{12}X_{12} + F_{21}X_{21} + F_{22}X_{22}$$

$$O_{12} = F_{11}X_{12} + F_{12}X_{13} + F_{21}X_{22} + F_{22}X_{23}$$

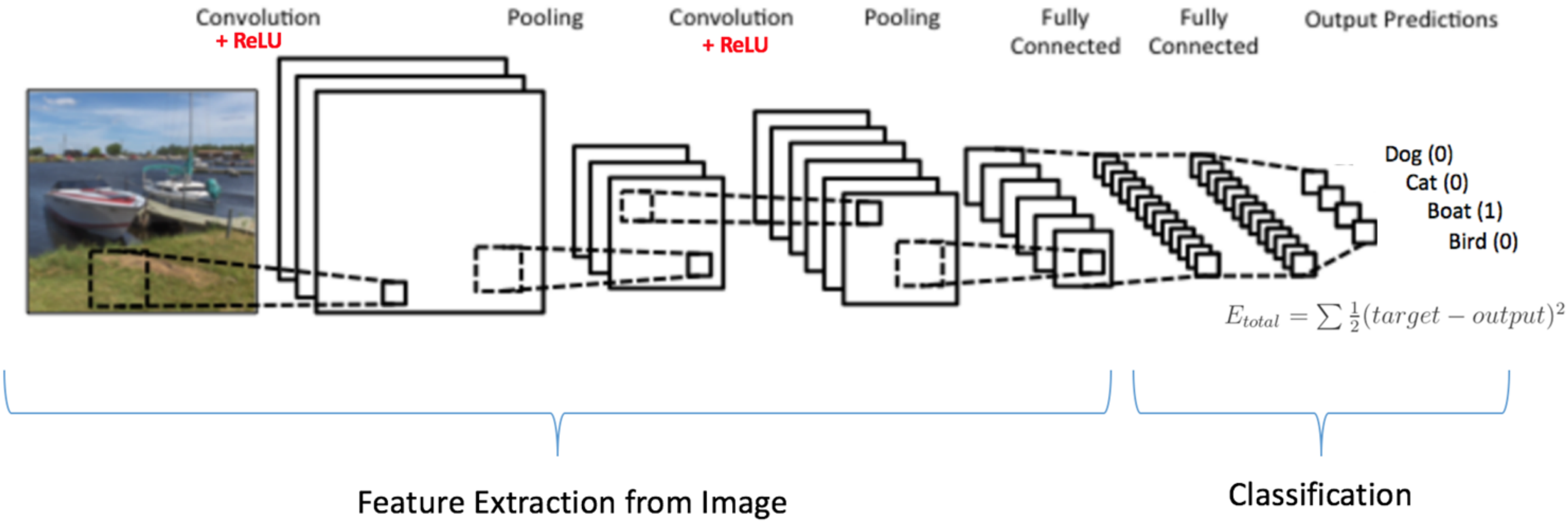
$$O_{21} = F_{11}X_{21} + F_{12}X_{22} + F_{21}X_{31} + F_{22}X_{32}$$

$$O_{22} = F_{11}X_{22} + F_{12}X_{23} + F_{21}X_{32} + F_{22}X_{33}$$

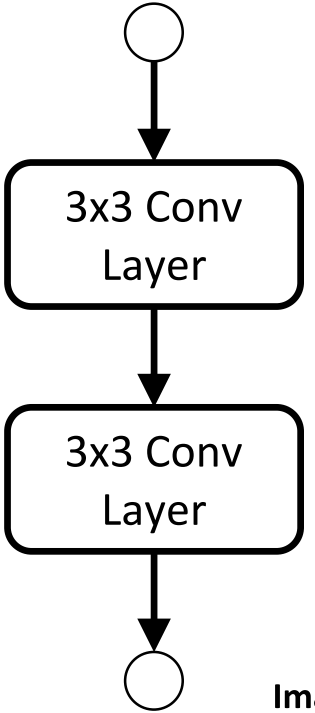
- Details:
- https://www.slideshare.net/EdwinEfranJimnezLepe/example-feedforward-backpropagation
- https://medium.com/@2017csm1006/forward-and-backpropagation-in-convolutional-neural-network-4dfa96d7b37e

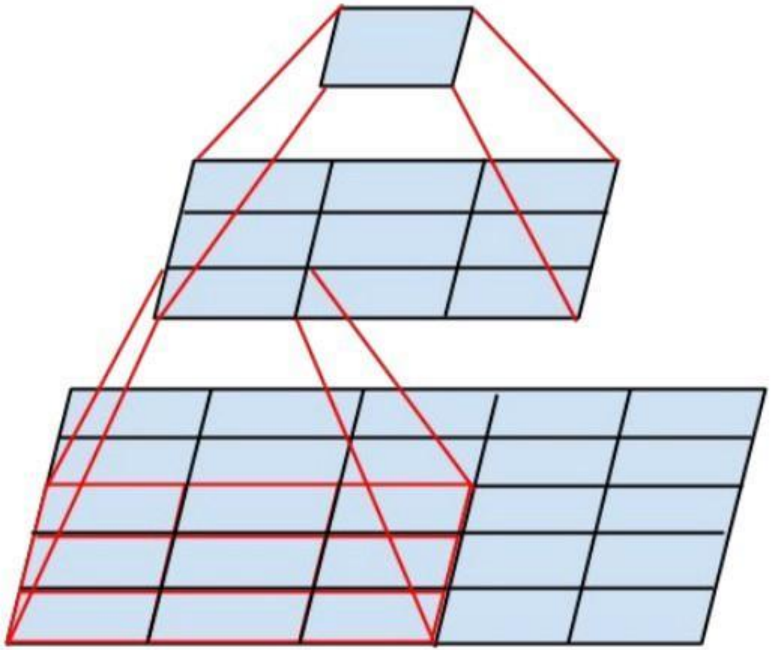
CNN for Image Classification

Vanilla Structure for Image Classification



Convolutional Unit (CU) - VGG





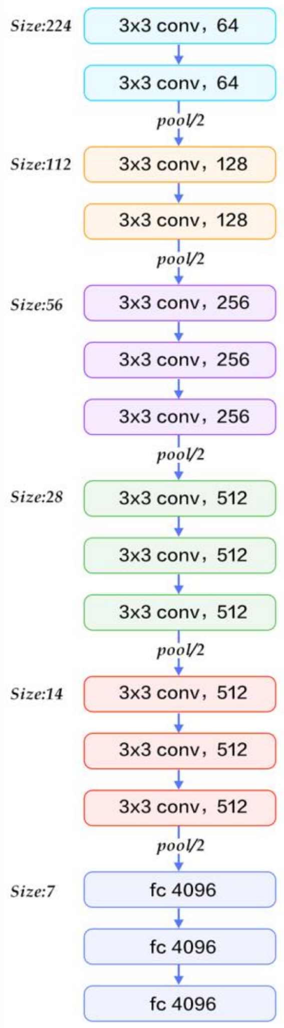
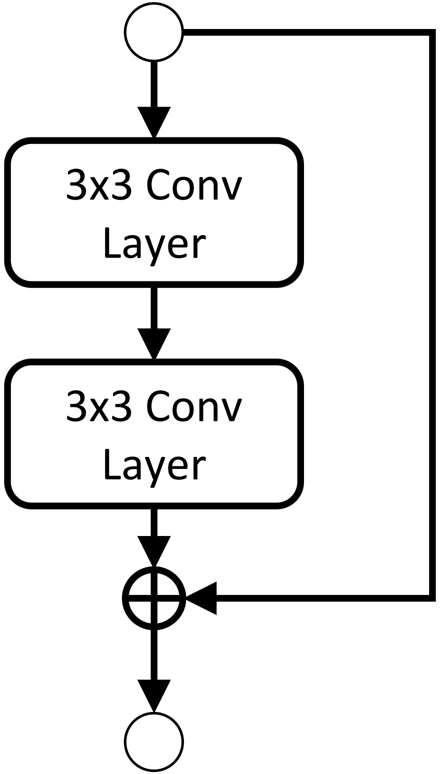


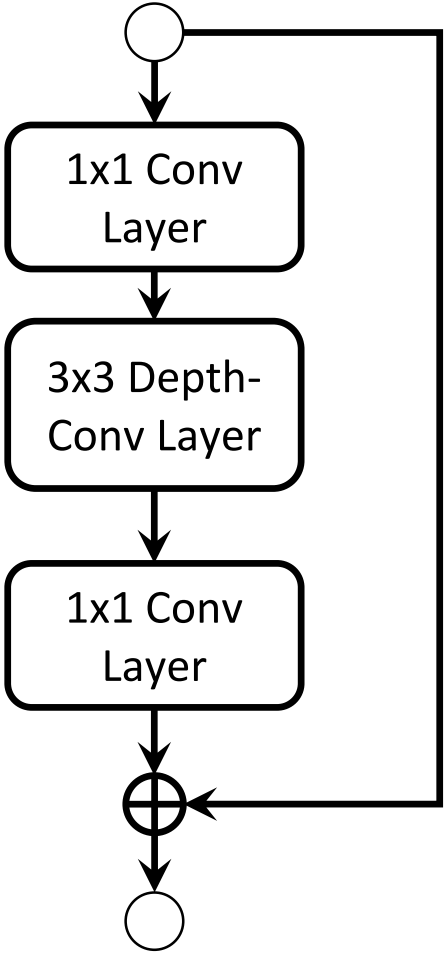
Image Source (Inception): Szegedy, Christian, et al. "Rethinking the inception architecture for computer vision." *CVPR*. 2016.

Basic Block in ResNet



ResNet: He, Kaiming, et al. "Deep residual learning for image recognition." CVPR. 2016.

- Residual Connection
- Element-wise addition of input and output
- Improves gradient flow and accuracy
- In ResNet-18 and ResNet-34
- Still computationally expensive
    - Hard to train very deep networks (> 100 layers)



Bottleneck in ResNet

- Used in ResNet-50, ResNet-101, ResNet-152, etc...
- Computationally Efficient

Influence:

- Bottleneck unit with Depth-wise convs
    - MobileNetv2
    - ShuffleNetv2

- **MobileNetv2:** Sandler, Mark, et al. "Mobilenetv2: Inverted residuals and linear bottlenecks." CVPR, 2018.
- **ShuffleNetv2:** Ma, Ningning, et al. "Shufflenet v2: Practical guidelines for efficient cnn architecture design." ECCV, 2018.

Other Structures and Blocks

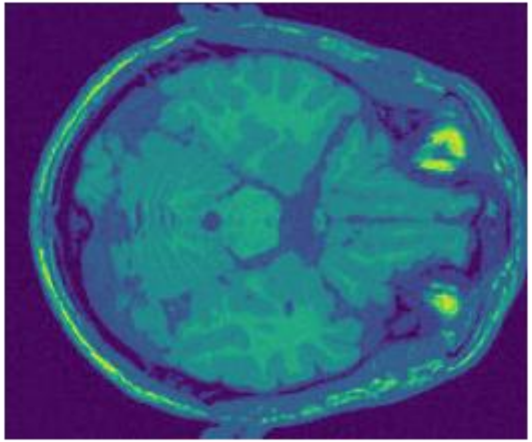
- InceptionNet, 2014, 2016 (v4)
- DenseNet, 2016
- MobileNet, 2017
- ESPNet, 2018
- EfficientNet, 2019
- RegNet, 2021
- ...

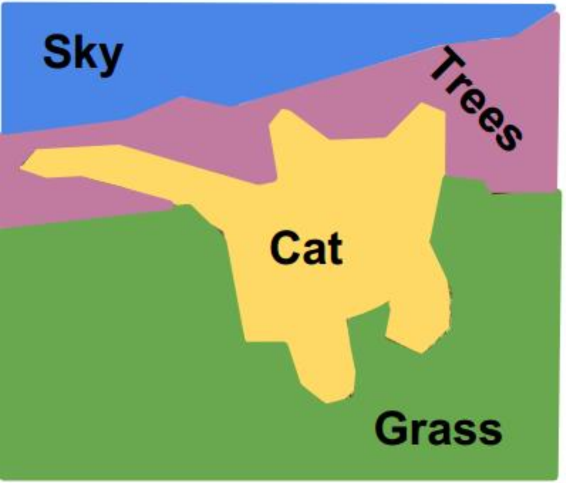
Transformer-based (Not convolutional)

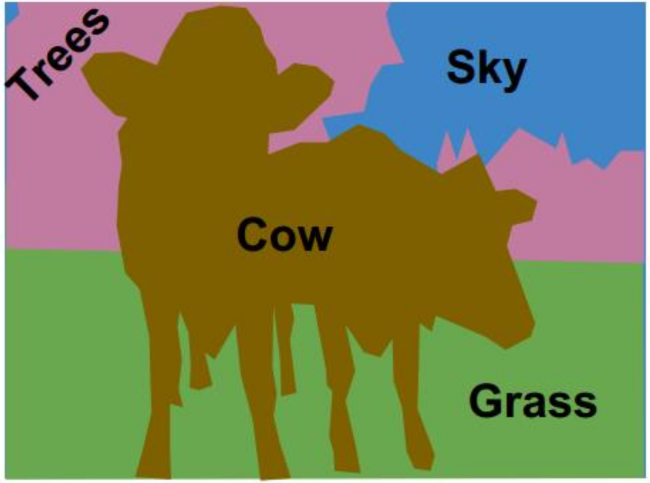
- ViT, 2020
- DERT, 2020

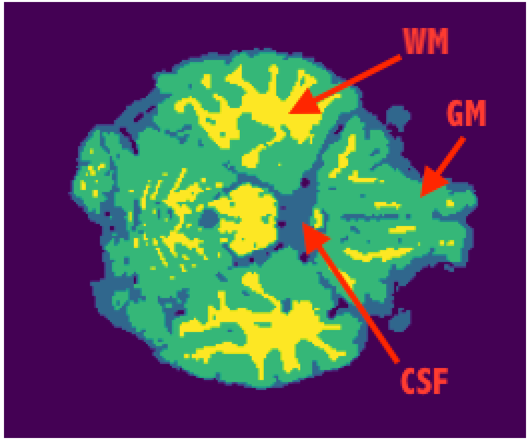
CNN for Semantic Segmentation

HW 2



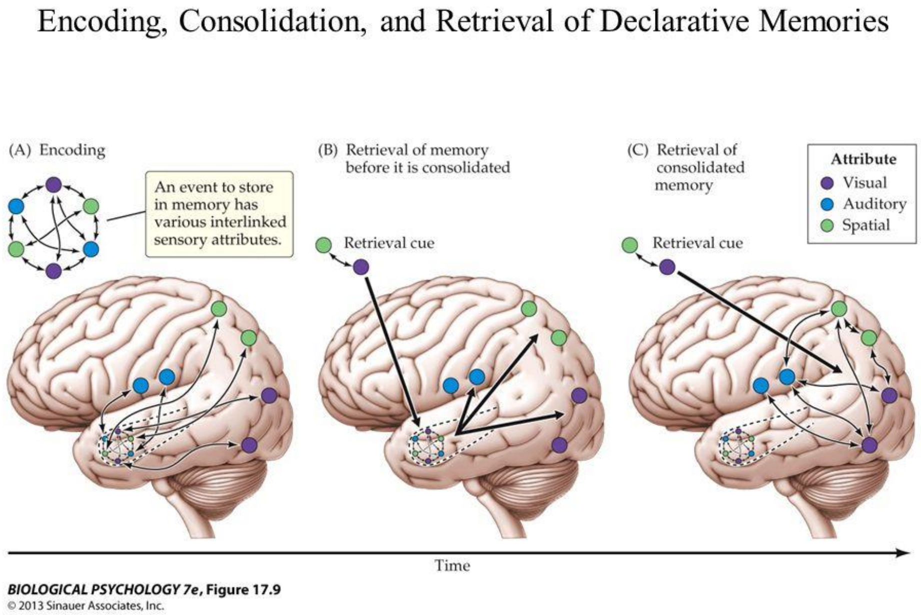






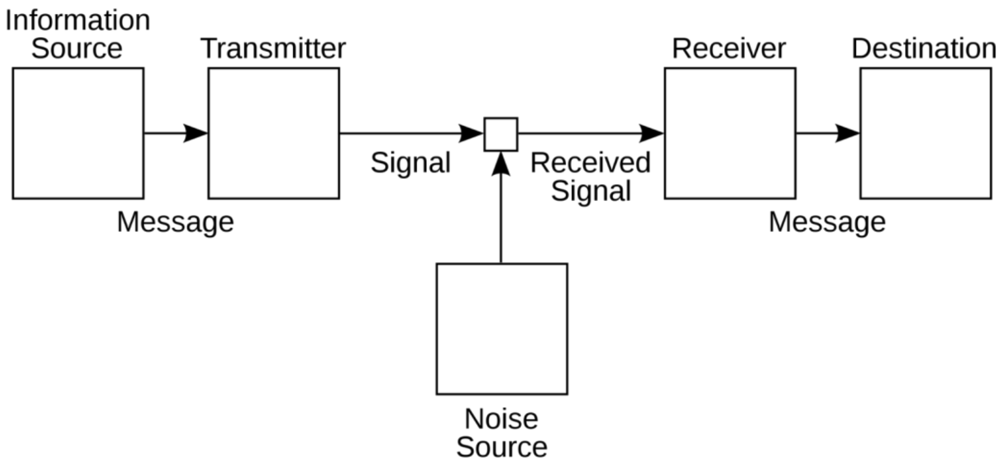
Encoding in Brain

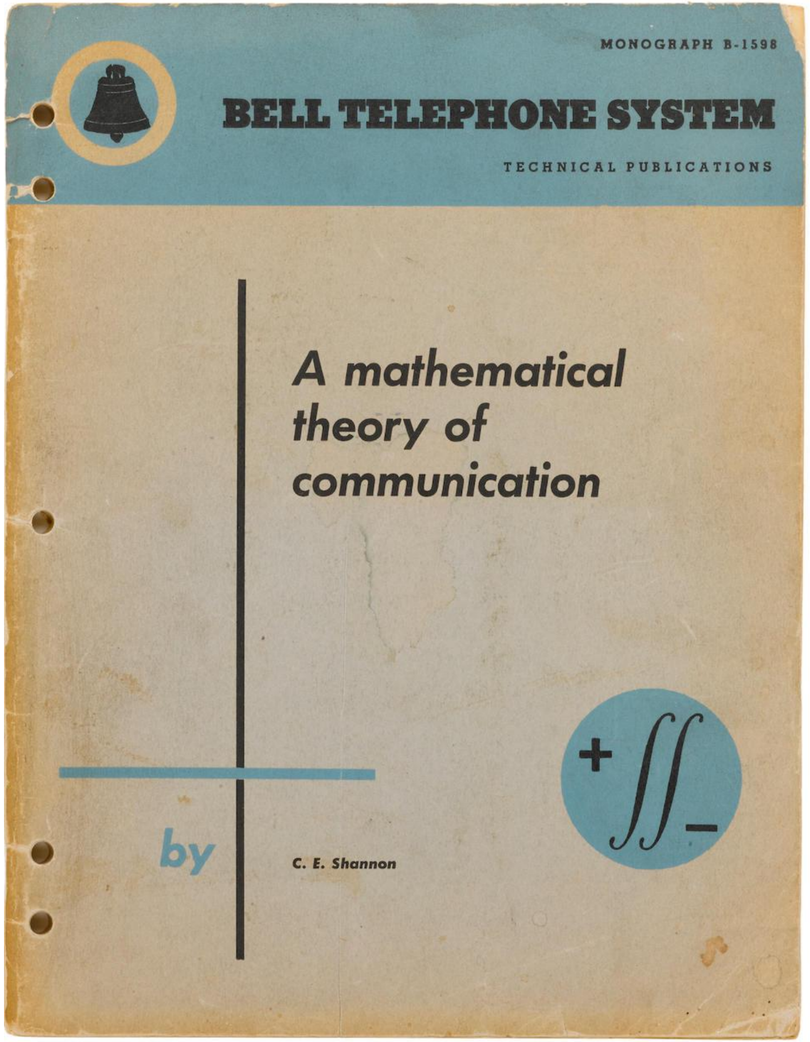
- Brains can:
  - Encode, store and recall information
  - Learn and adapt from previous experiences
  - Build relationships of data
- **Encoding** allows a **perceived item** to be converted into a **construct** that can be stored within the brain and recalled later.
- Studied since 1880s.

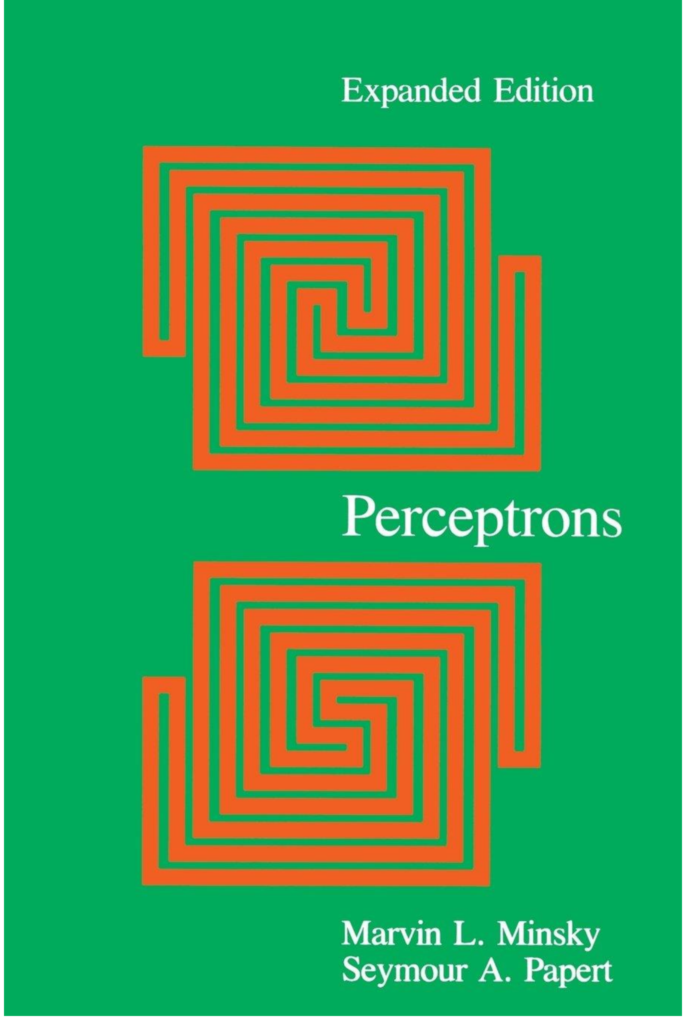


(Wikipedia)

Encoding in Computer







1969

CHAPTER 8

Learning Internal Representations by Error Propagation

D. E. RUMELHART, G. E. HINTON, and R. J. WILLIAMS

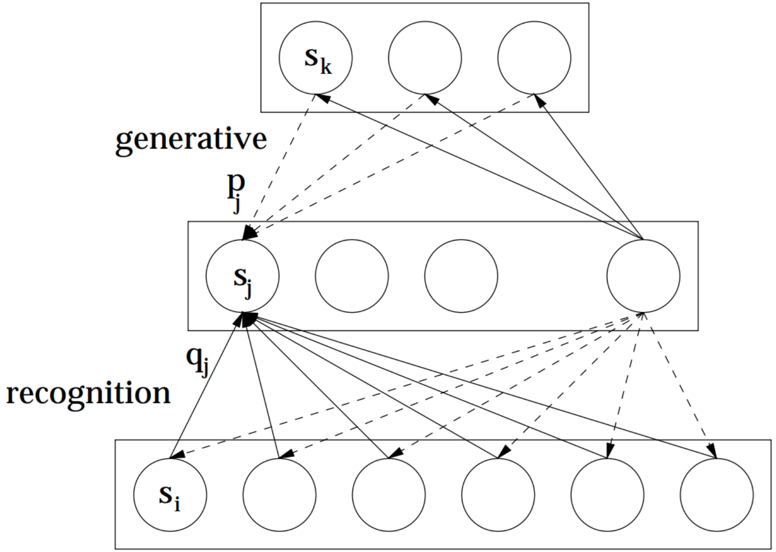
1987

The wake-sleep algorithm for unsupervised neural networks

Geoffrey E Hinton* Peter Dayan Brendan J Frey Radford M Neal

Department of Computer Science
University of Toronto
6 King's College Road
Toronto M5S 1A4, Canada

3rd April 1995



Hinton, Geoffrey E., et al. "The" wake-sleep" algorithm for unsupervised neural networks." *Science* 268.5214 (1995): 1158-1161.

Autoencoder (aka, Encoder-Decoder)

- PCA (SVD)
- Boltzmann machines
- Wake-Sleep Algorithm
- Information Theory
- Unsupervised Learning

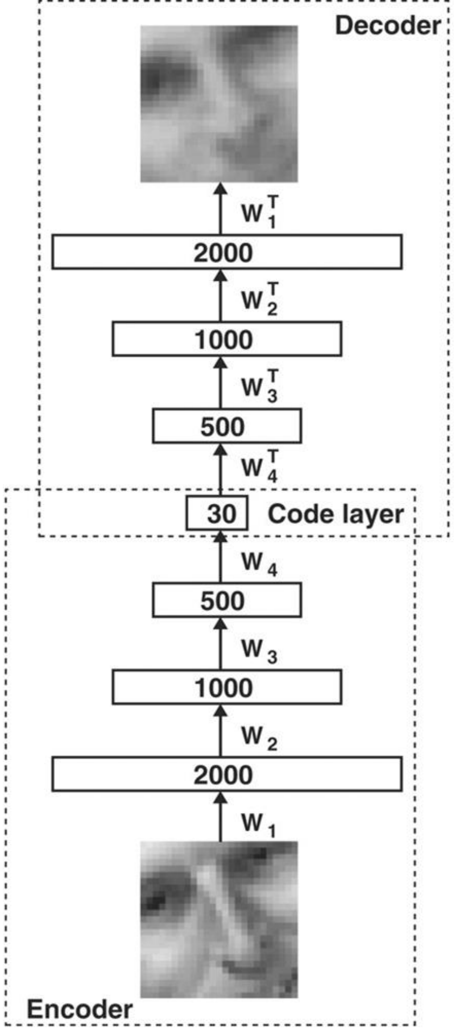
Reducing the Dimensionality of Data with Neural Networks

G. E. Hinton* and R. R. Salakhutdinov

High-dimensional data can be converted to low-dimensional codes by training a multilayer neural network with a small central layer to reconstruct high-dimensional input vectors. Gradient descent can be used for fine-tuning the weights in such "autoencoder" networks, but this works well only if the initial weights are close to a good solution. We describe an effective way of initializing the weights that allows deep autoencoder networks to learn low-dimensional codes that work much better than principal components analysis as a tool to reduce the dimensionality of data.

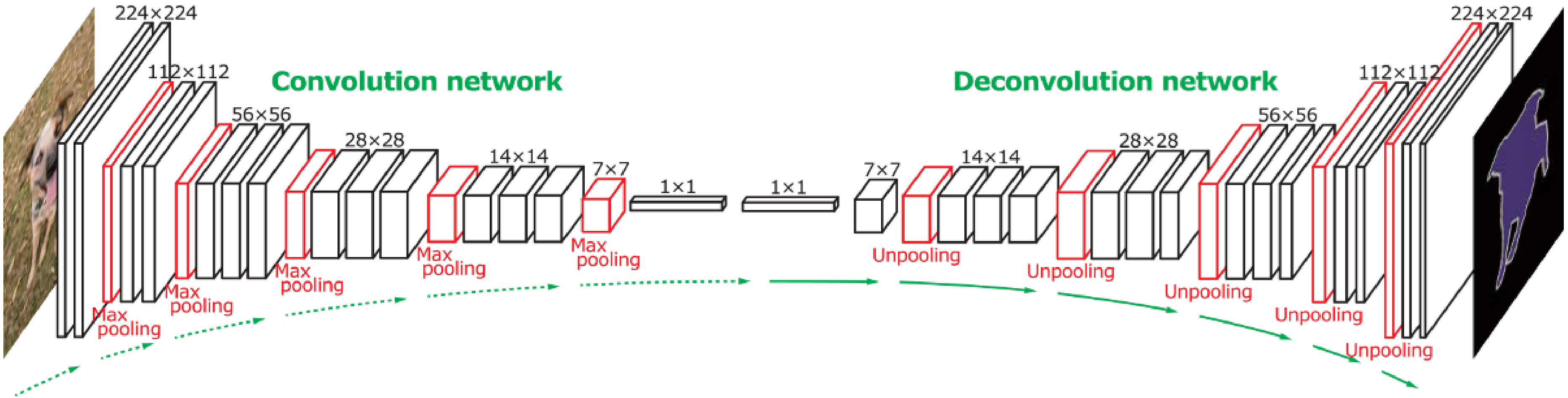
Dimensionality reduction facilitates the classification, visualization, communication, and storage of high-dimensional data. A simple and widely used method is principal components analysis (PCA), which finds the directions of greatest variance in the data set and represents each data point by its coordinates along each of these directions. We describe a nonlinear generalization of PCA that uses an adaptive, multilayer "encoder" network

2006 VOL 313 SCIENCE www.sciencemag.org



Unrolling

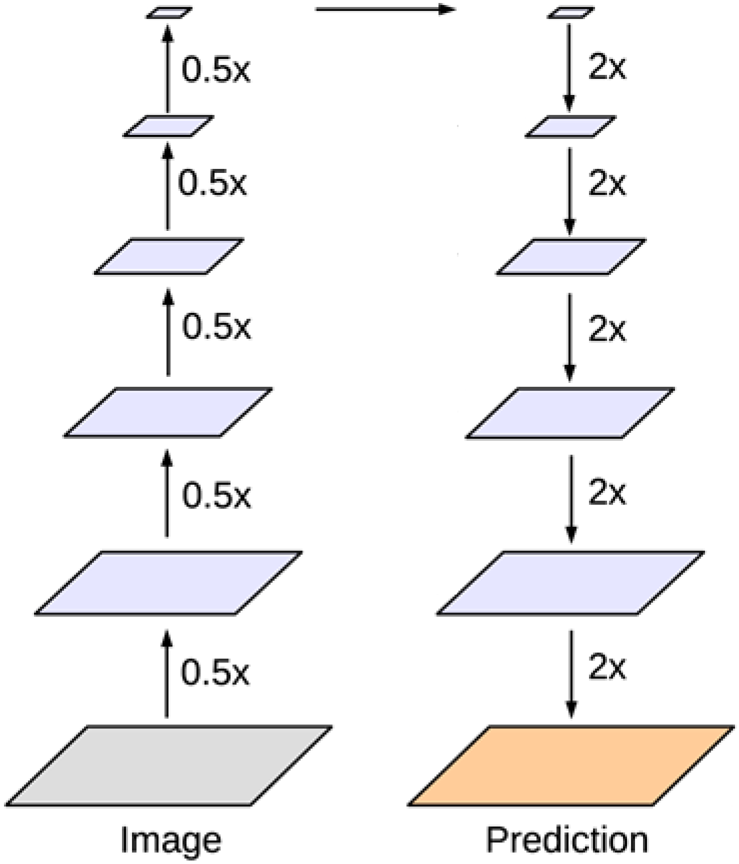
FCN: From MLP to Convolutional



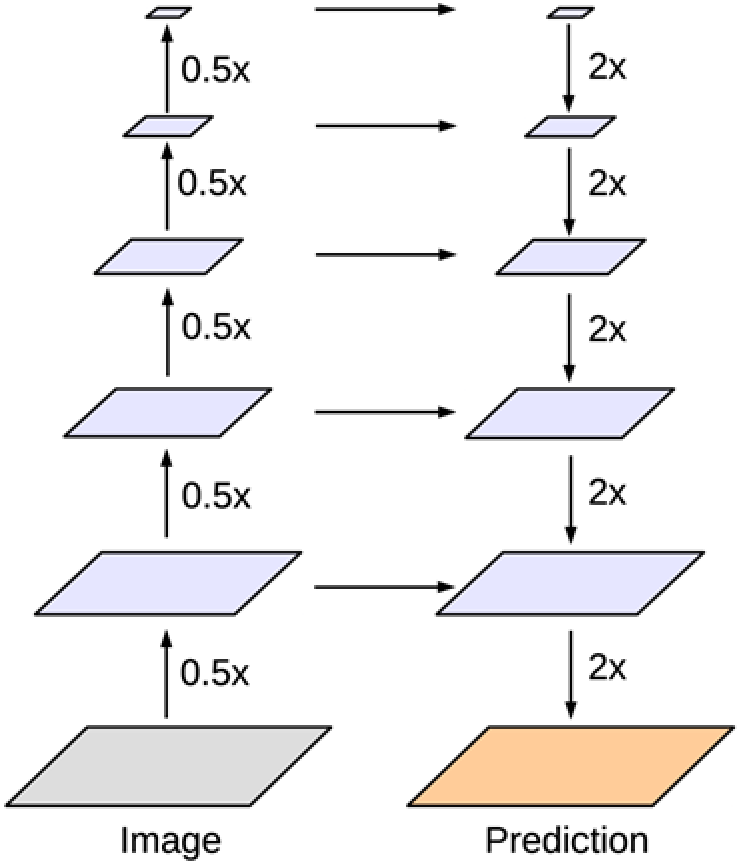
Long, Jonathan, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015.

U-Net

FCN



U-Net

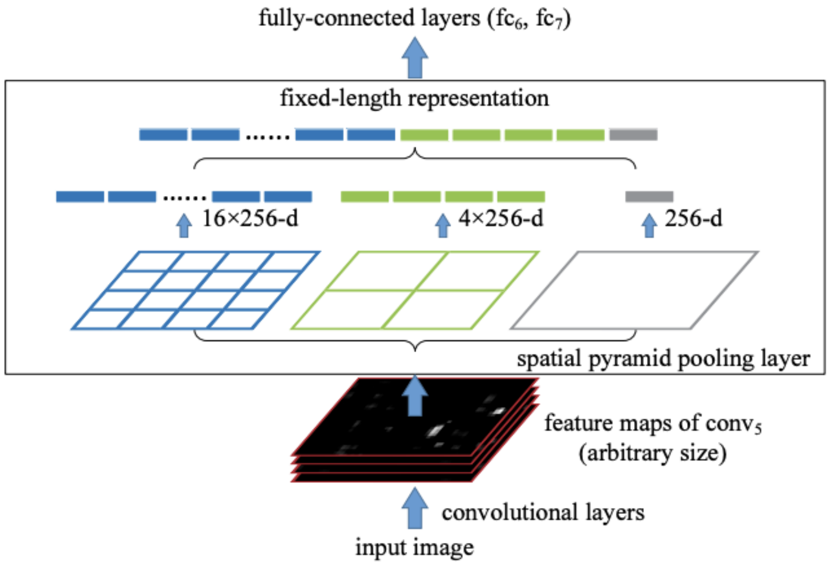


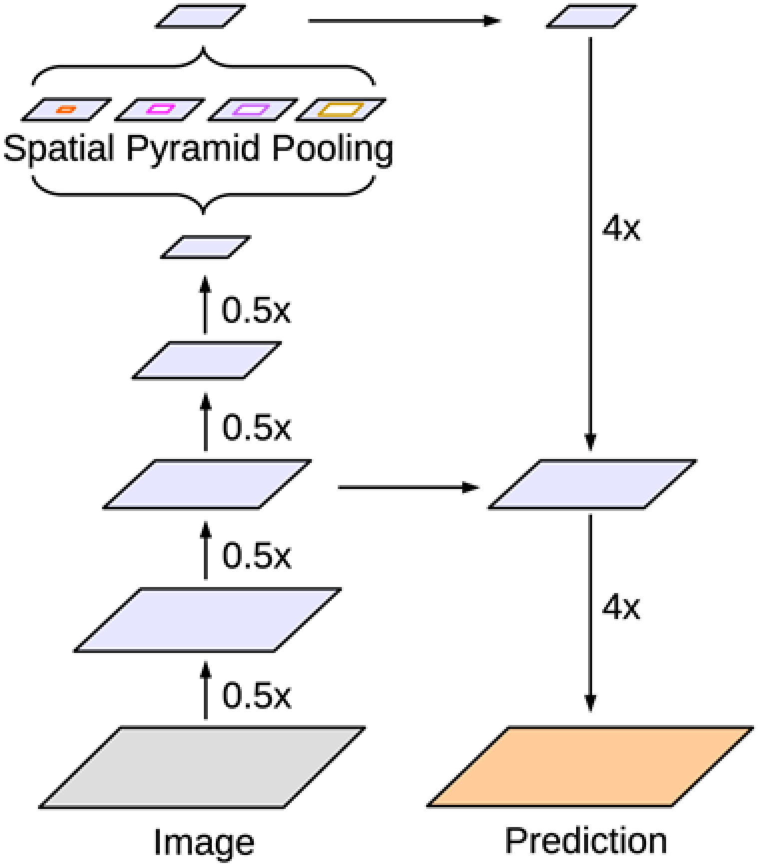
Applications of U-Net

- Nuechterlein, Nicholas, and Sachin Mehta. "3D-ESPNet with pyramidal refinement for volumetric brain tumor image segmentation." ***MICCAI***. Springer, Cham, **2018**.
- Nikolov, Stanislav, et al. "Clinically Applicable Segmentation of Head and Neck Anatomy for Radiotherapy: Deep Learning Algorithm Development and Validation Study." ***Journal of Medical Internet Research*** 23.7 (**2021**): e26151.
- Wilson, Marc, et al. "Validation and Clinical Applicability of Whole-Volume Automated Segmentation of Optical Coherence Tomography in Retinal Disease Using Deep Learning." ***JAMA ophthalmology*** 139.9 (**2021**): 964-973.
- Wang, Shanshan, et al. "Annotation-efficient deep learning for automatic medical image segmentation." ***Nature Communications*** 12.1 (**2021**): 1-13.

DeepLab

- Spatial Pyramid Pooling, 2016
  - Learn from different receptive fields
- Nowadays, DeepLab V3 +, 2017





Extra Credits for HW 2

U-Net Model

```
[ ] #@title Create U-Net Model

    # Set random seed for reproduciablity
    torch.manual_seed(577)
    random.seed(577)

    # The `init_features` defines how many channels the hidden layers contain.
    # Usually, people would use 32 init features. Here, because our dataset
    # is small, we use 4 init feautures. So, the channel size is about 32 / 4 = 8
    # times smaller, and the training speed is about 8 x 8 = 64 times faster.
    model = torch.hub.load('mateuszbuda/brain-segmentation-pytorch', 'unet',
        in_channels=1, out_channels=4, init_features=4, pretrained=False)
```

Create U-Net Model

```
Downloading: "https://github.com/mateuszbuda/brain-segmentation-pytorch/archive/master.zip" to /root/.cache/torch/hub/master.zip
```

+ Code + Text

Create Data Loader

```
[ ] #@title TODO: create data loader

    # TODO: create a data loader with batch size 8 and shuffling.
    # Note 1: check `torch.utils.data.dataloader` .
    # Note 2: You will need the `dataset` variable from the above block.
    dataloader = None
```

Gradient Descent (Adam)

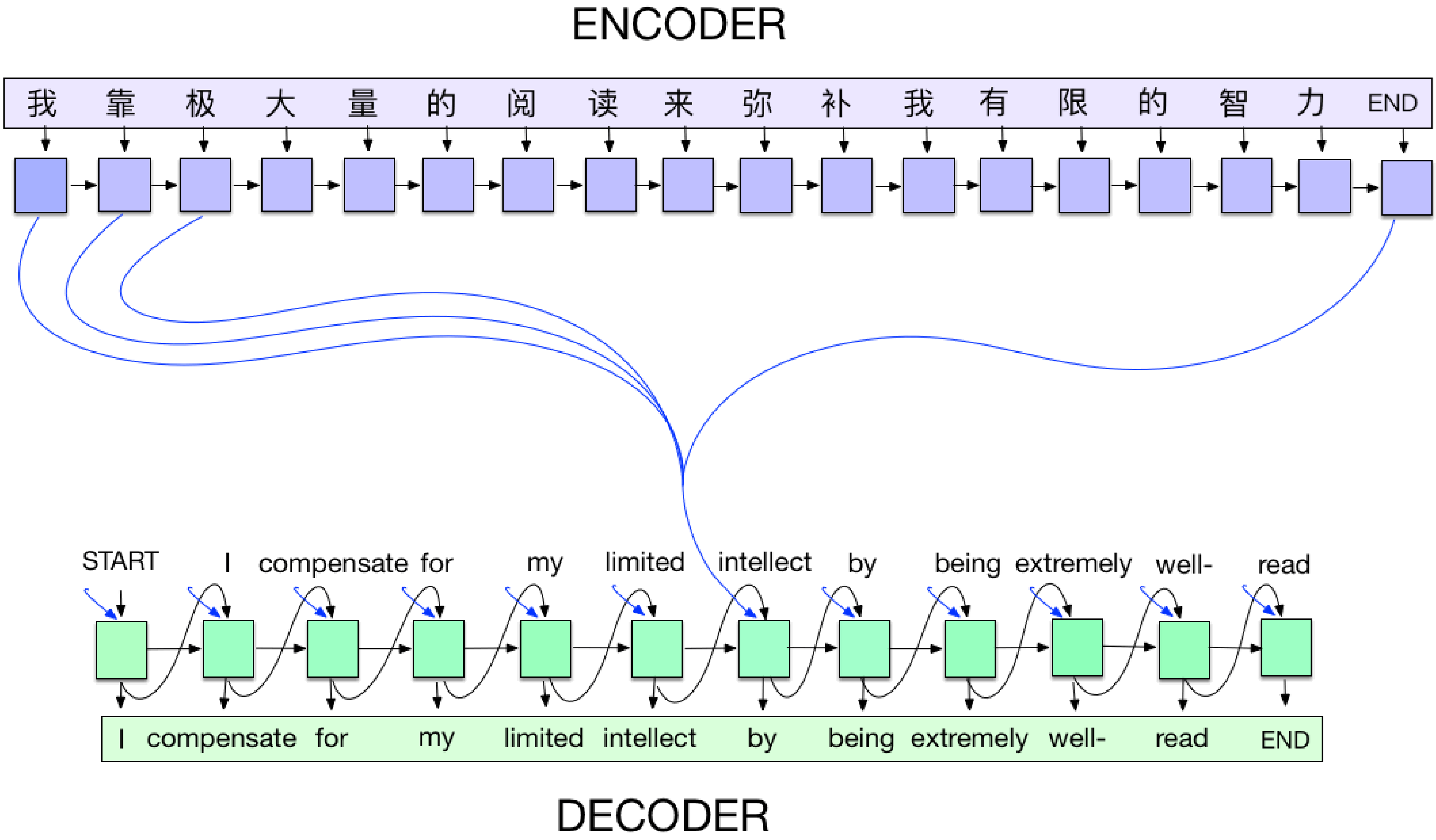
```
[ ] #@title Define loss and optimizer
    criterion = torch.nn.CrossEntropyLoss()
```

```
[ ] #@title TODO: create an optimizer


    # TODO: create an Adam optimizer with learning rate 0.001.
    # Paper Reference: https://arxiv.org/abs/1412.6980
    # Note: check out `torch.optim.Adam`
    optimizer = None
```

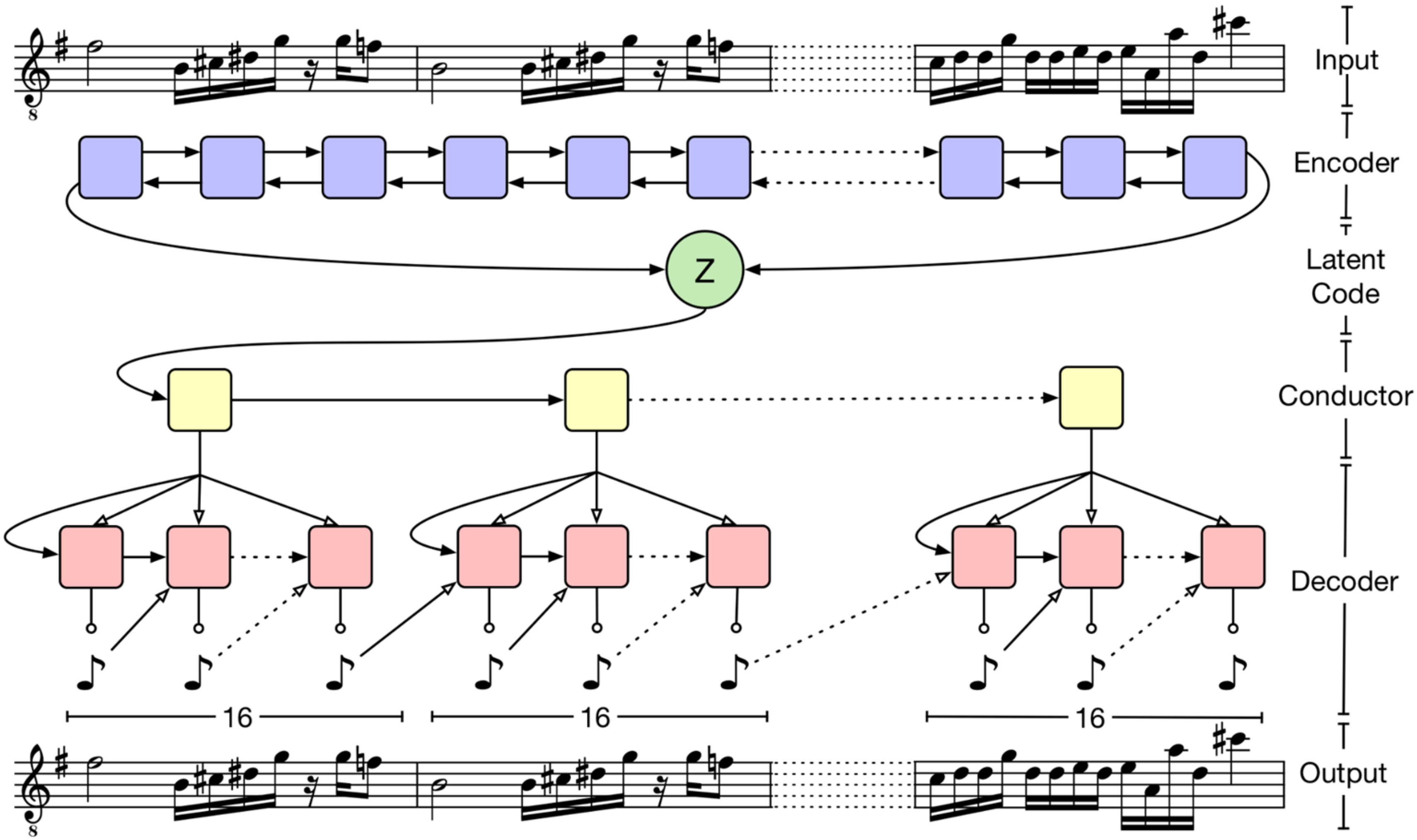
Training

```
[ ] #@title Training
    # For simplicity, we only train a few epochs. Good enough~
    # Note that you might need 20 minutes to run this block.

    # If you have enough time (or machine power), you can increase
    # the number of epochs. It can improve your results.
    # Don't turn off your monitor while running the code: if the
    # machine falls asleep, the Colab will stop working.
    num_epochs = 5


    use_cuda = torch.cuda.is_available()
    if use_cuda:
      model = model.cuda()

    for epoch in range(num_epochs):
      losses = []
      if dataloader is None or optimizer is None:
        break  # NotImplementedError
      for x, y in tqdm.tqdm(dataloader):
        if use_cuda:
          x, y = x.cuda(), y.cuda()
        pred = model(x.float())
        loss = criterion(pred, y.long())

        loss.backward()
        optimizer.step()
        optimizer.zero_grad()

        losses.append(loss.item())

      print("Epoch:", epoch, "Mean Loss:", np.mean(losses))
```

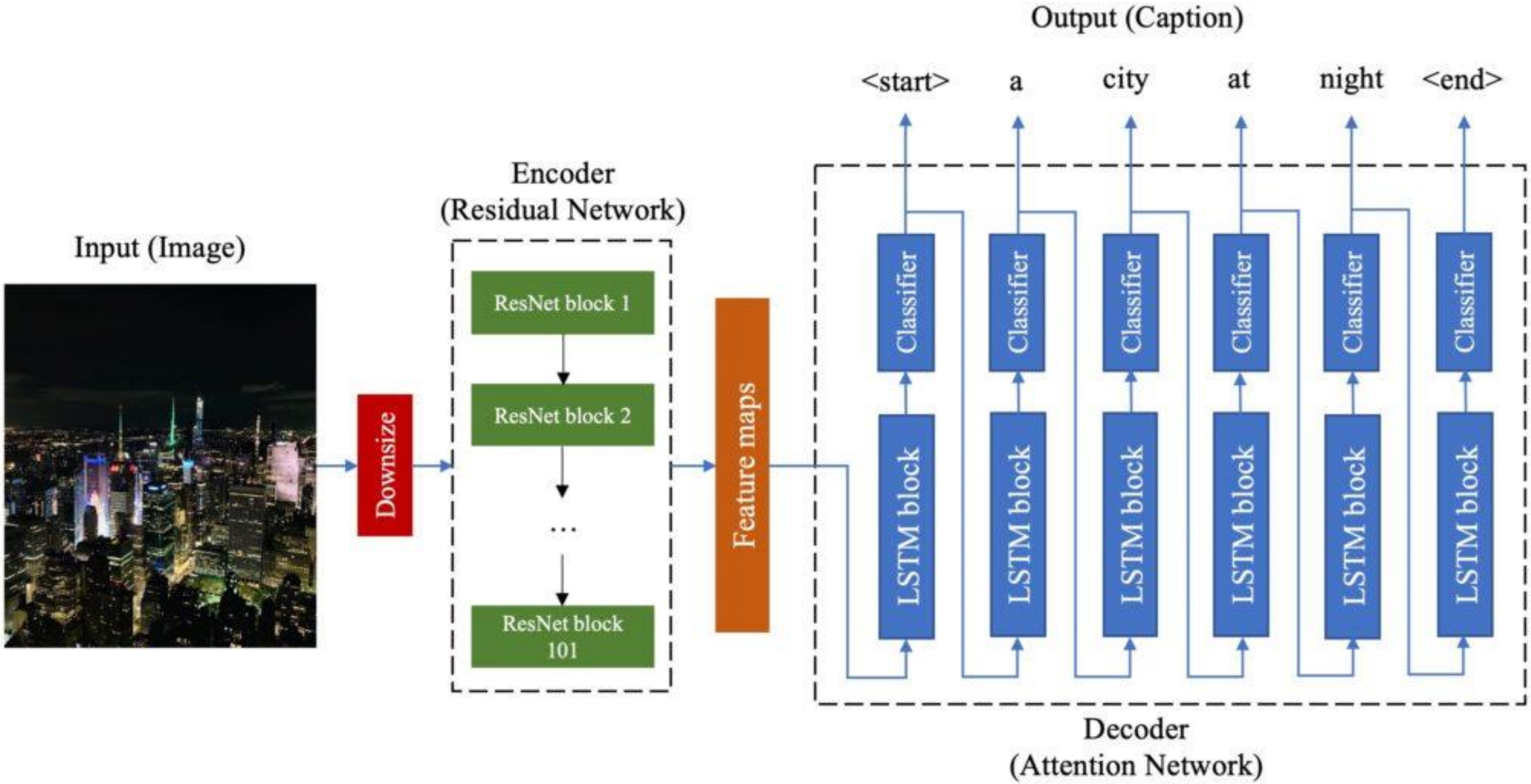
Other Applications of Autoencoder



ENCODER
我 靠 极 大 量 的 阅 读 来 弥 补 我 有 限 的 智 力 END
START I compensate for my limited intellect by being extremely well- read
I compensate for my limited intellect by being extremely well- read END
DECODER

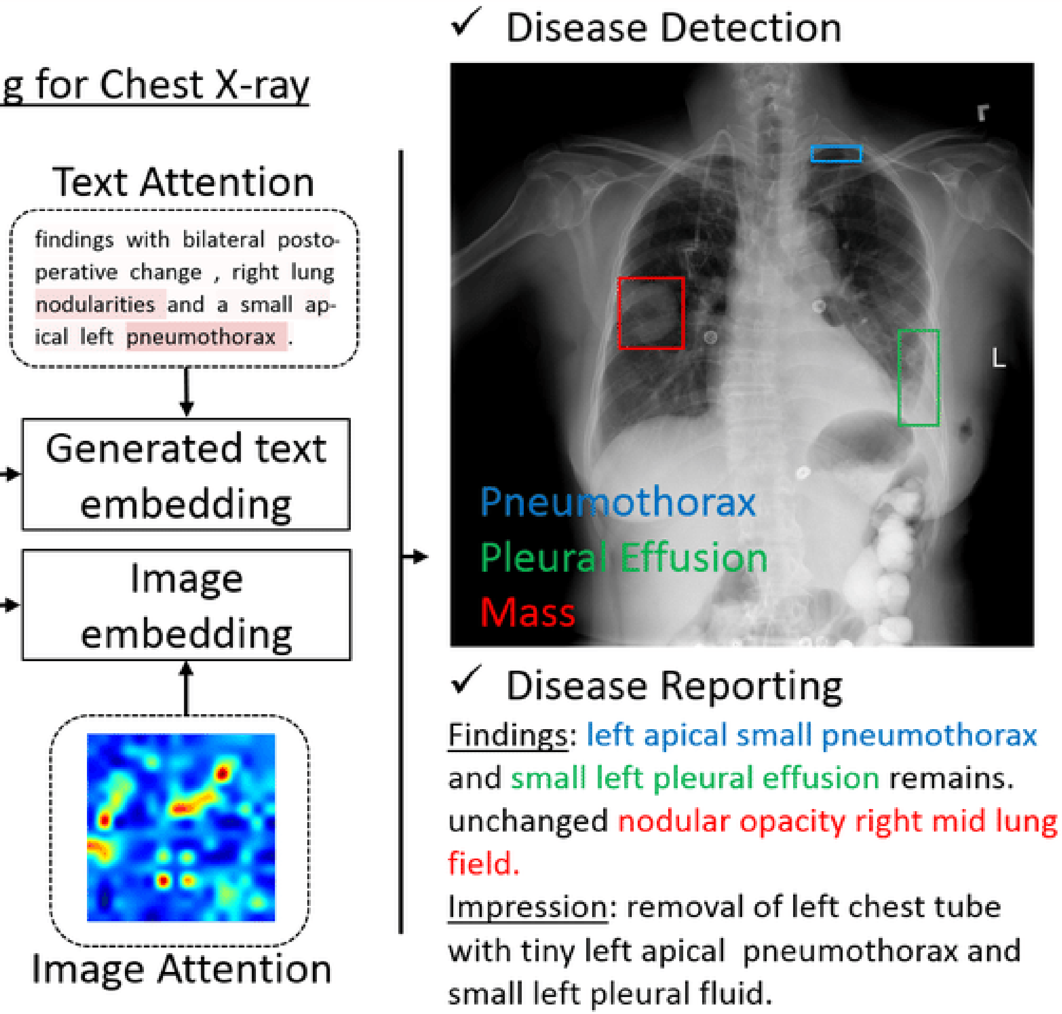


Input
Encoder
Z
Latent
Code
Conductor
Decoder
16
16
16
Output

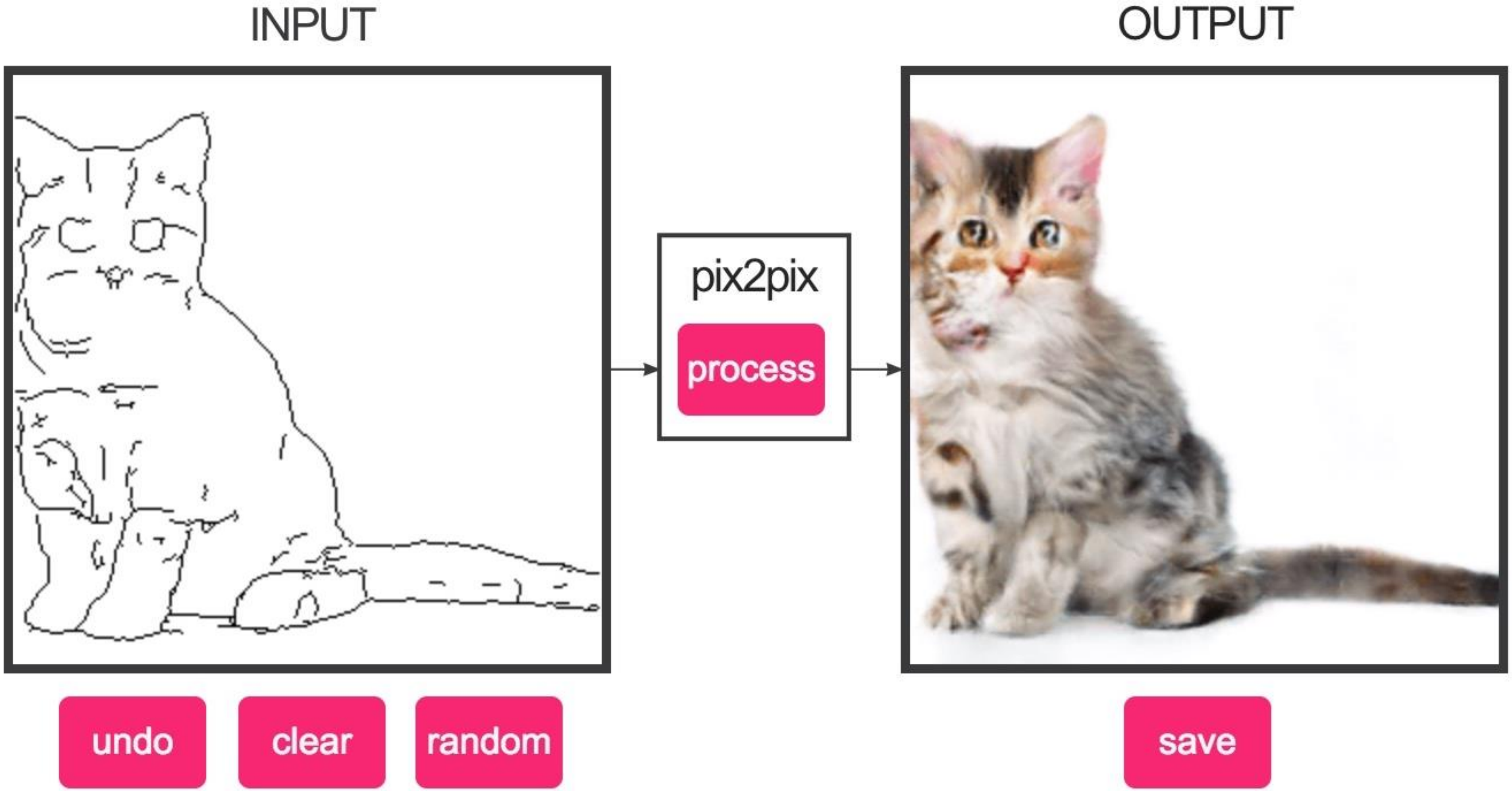


https://blog.filestack.com/api/filestack-image-captioning-an-image-describer-using-attention-networks/

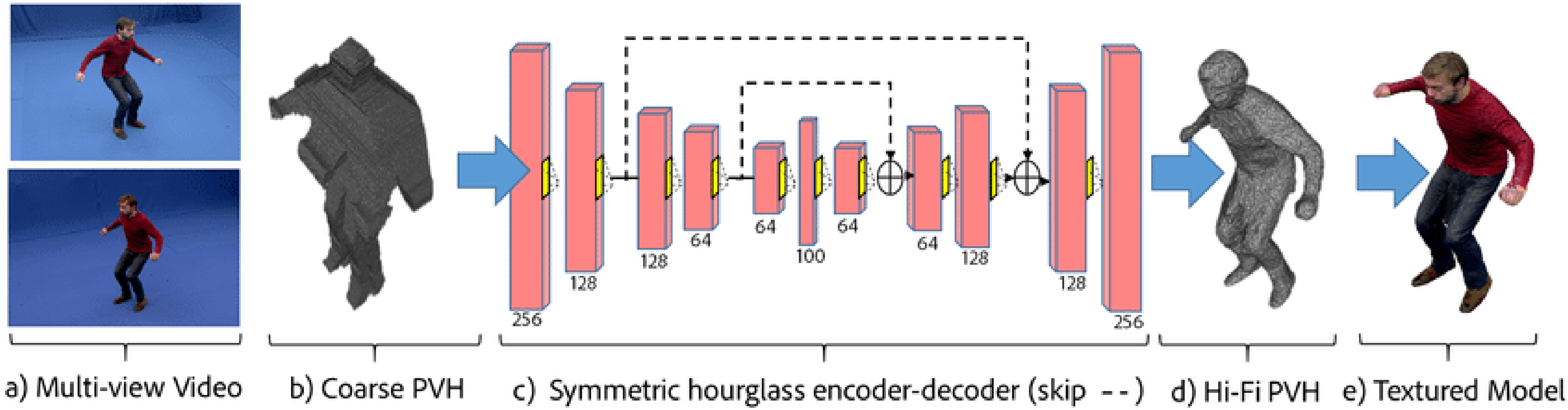
Towards Automated Reporting for Chest X-ray



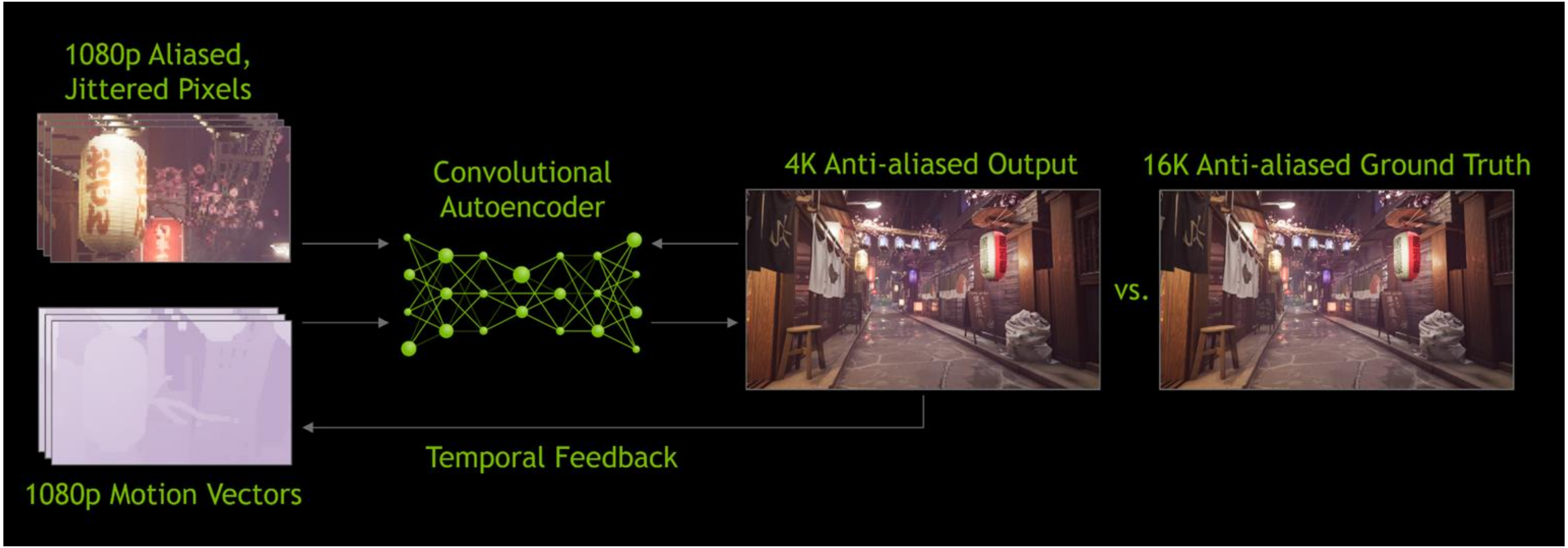
Wang, Xiaosong, et al. "Tienet: Text-image embedding network for common thorax disease classification and reporting in chest x-rays." *IEEE CVPR*. 2018.



Isola, Phillip, et al. "Image-to-image translation with conditional adversarial networks." *Proceedings of the IEEE CVPR*. 2017.



Gilbert, Andrew & Volino, Marco & Collomosse, John & Hilton, Adrian. (2018). Volumetric performance capture from minimal camera viewpoints.



1080p Aliased,
Jittered Pixels
Convolutional
Autoencoder
4K Anti-aliased Output
16K Anti-aliased Ground Truth
vs.
Temporal Feedback
1080p Motion Vectors

Applications... Multi-Modality

The encode data and decode data can be...

- Image
- Video
- 3D Image
- Audio
- Text
- Signal
- More...

Neural Encoding and Decoding with Deep Learning for Dynamic Natural Vision

ORIGINAL ARTICLE

Neural Encoding and Decoding with Deep Learning for Dynamic Natural Vision

Haiguang Wen[1,2], Junxing Shi[1,2], Yizhen Zhang[1,2], Kun-Han Lu[1,2], Jiayue Cao[2,3] and Zhongming Liu[1,2,3]

[1]School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47906, USA, [2]Purdue Institute for Integrative Neuroscience, Purdue University, West Lafayette, IN 47906, USA and [3]Weldon School of Biomedical Engineering, Purdue University, West Lafayette, IN 47906, USA

Address correspondence to Zhongming Liu, Assistant Professor of Biomedical Engineering, Assistant Professor of Electrical and Computer Engineering, College of Engineering, Purdue University, 206 S. Martin Jischke Dr, West Lafayette, IN 47907, USA. Email: zmliu@purdue.edu

Overview

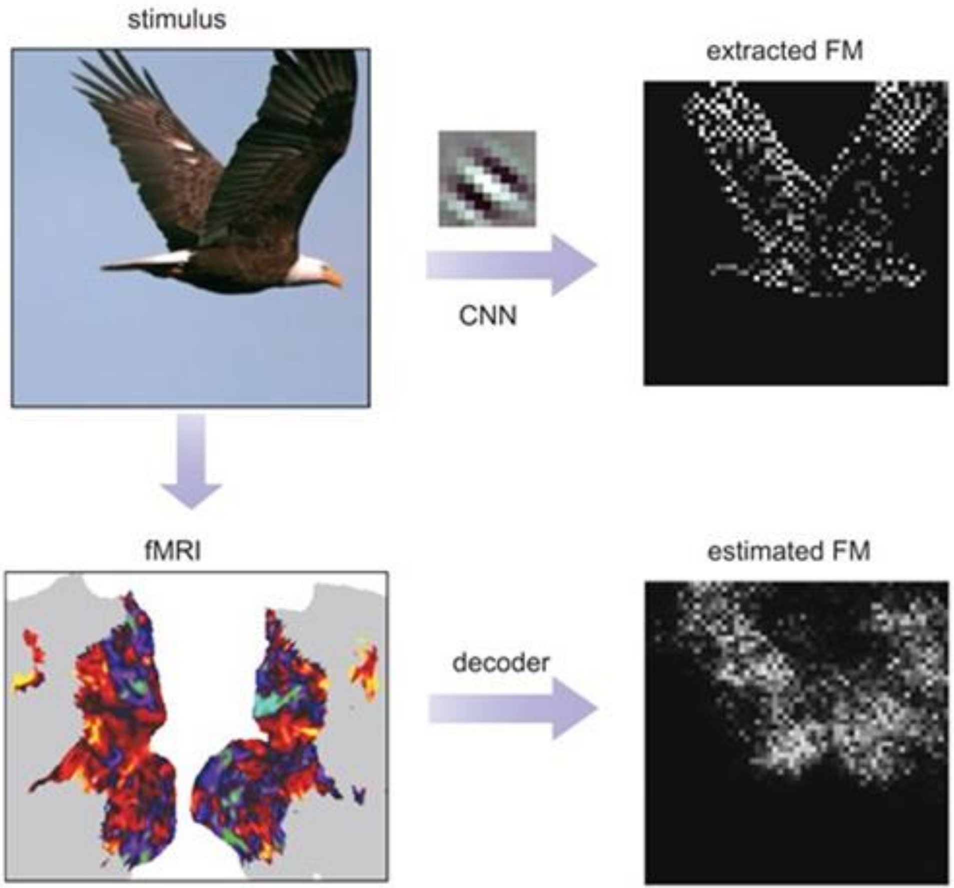
- Subjects were in an MRI machine watching colorful videos that contained various object categories.
- fMRI showed activated areas of the brain as the subjects watched the videos.
- Encoding and decoding models were developed and evaluated for describing the bi-directional relationships between the CNN and the brain.

Details

- Studied how the brain decodes visual information.
- Acquired 11.5 hours of fMRI data from each of 3 human subjects watching 972 different video clips including diverse scenes and actions.
- Used AlexNet to extract hierarchical visual features from the movie stimuli.
- Five convolutional layers and three fully connected layers.
- Reduced the original 1000 categories to 15.
- Compared the outputs of CNN units to the fMRI signals.
- Trained encoding and decoding models for describing the relationship between the brain and the CNN.

Figure 7.

(a)



FM = Feature Map

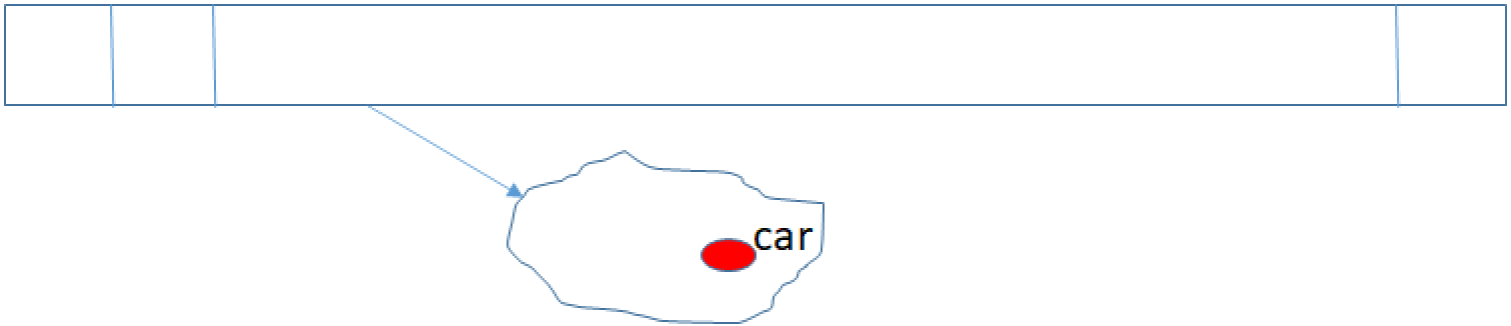
Figure 8.

Open in new tab Download slide

Reconstruction of a dynamic visual experience. For each row, the top shows the example movie frames seen by 1 subject; the bottom shows the reconstruction of those frames based on the subject's cortical fMRI responses to the movie. See Movie 1 for the reconstructed movie.

Findings

- The CNN recognizes the same object as the subject recognizes in the video from features in its penultimate layer.
- They correlate those features with brain voxels for each object.
- Then they can reconstruct the movie from the fMRI using a decoder.
- And also determine which object category from the fMRI.



Discussions:

Do the results mean human brains and artificial neural networks work in a similar way?

Discussions:

Do the results mean autoencoder can map brain activity to stimulus?

Pros vs Cons of This Study

Pros:

Cons: