Lecture 30
Machine Learning: Introduction to Applications of Deep Learning

MP574: Applications

Sean B. Fain, PhD (sfain@wisc.edu)

Diego Hernando, PhD (dhernando@wisc.edu)

ITK/VTK Applications: Andrew Hahn, PhD (adhahn@wisc.edu)

Learning Objectives

- Training and tuning of deep learning networks continued
- Current approaches to analyzing performance
- Some application examples

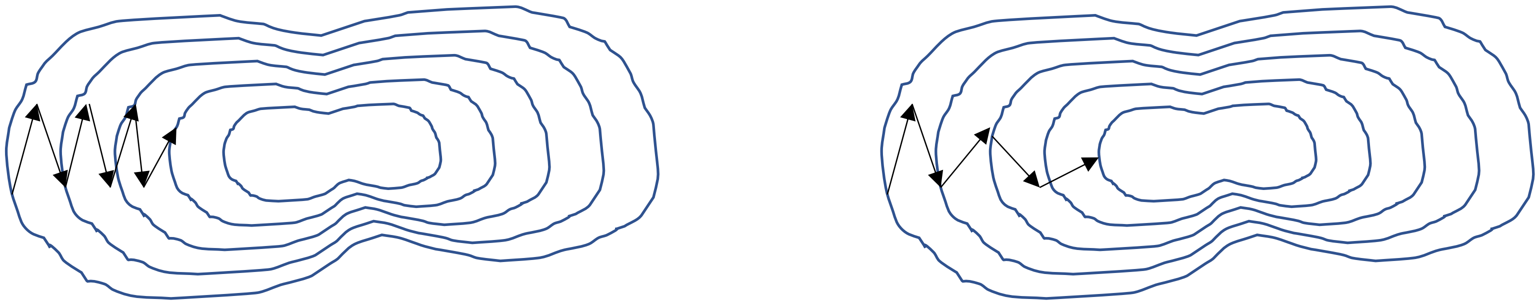
Tuning Training Process

Batch Learning

- Divides input data into equally sized batches B_k
- The training algorithm processes one batch at a time
- After each batch, the combined cost function is calculated and the weights are updated
- Increases learning rate
  - Smaller batches theoretically allow faster learning
  - If batch size is too small gradient directions might be wrong
    - Iterations might cancel each other out
    - Training might be slow or stagnate

Optimizers

- Gradient descent variations
  - Momentum: combines gradient directions with previous iterations



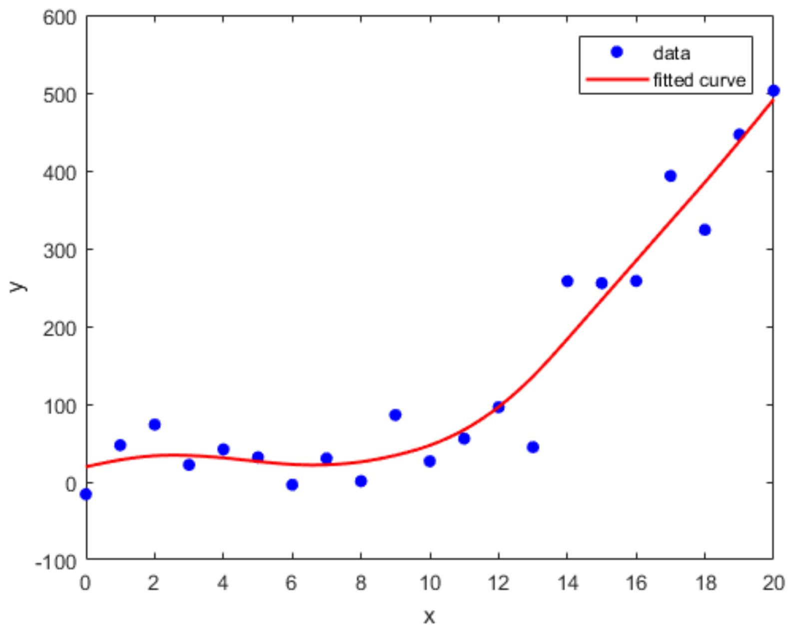
  - AdaGrad: calculates separate learning rate for each parameter (higher learning rates for sparse parameters)
  - Adam: combination of Momentum and AdaGrad optimizers

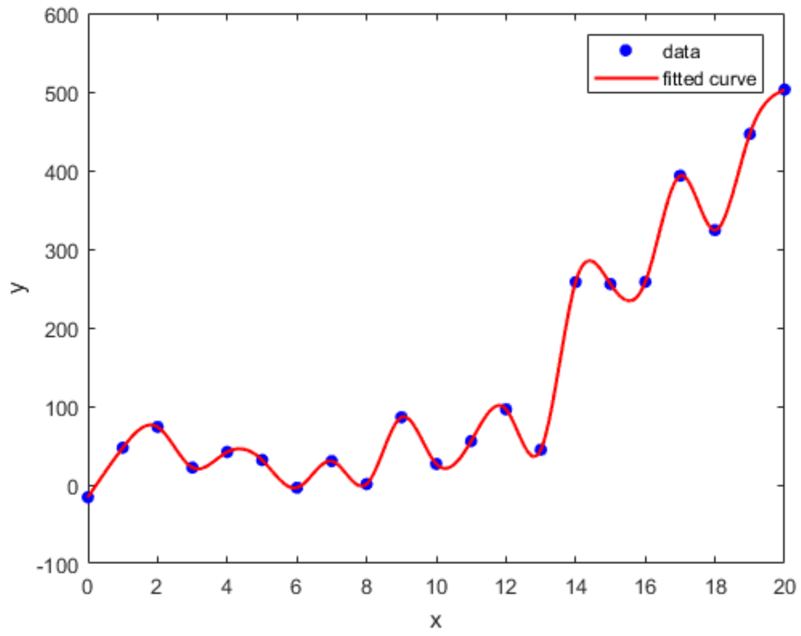
Input Data

- Deep learning requires large amount of training data
- Rule of thumb
  - Training data should be on the order of the number of variables in the model
- More data is always better
- Quality of data matters
  - Has to be representative for future data input
  - Consider possible variables

Generalization and Overfitting

- Just because a trained model works well on the **training data** does not mean it performs well for new data
- Common Problem: Overfitting





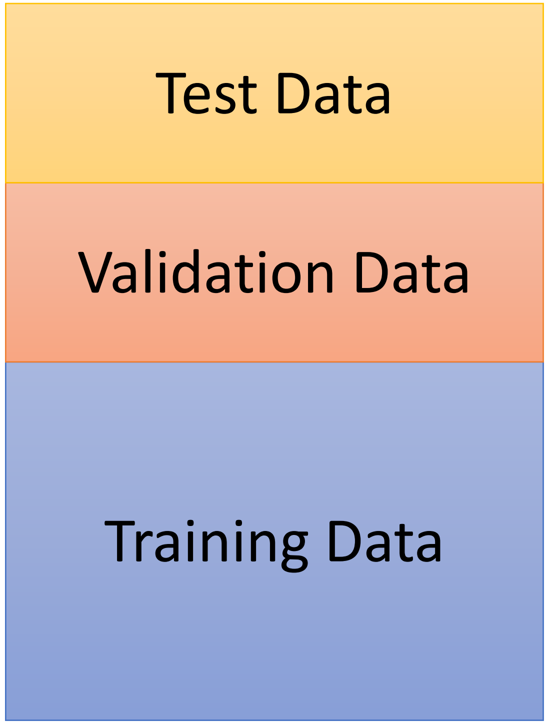
Generalization and Overfitting

- Common causes for overfitting
  - Not enough training data
  - Training data population too narrow
  - Too many variables in the model
  - Too many training iterations

➢ Monitoring training progress with independent data is important

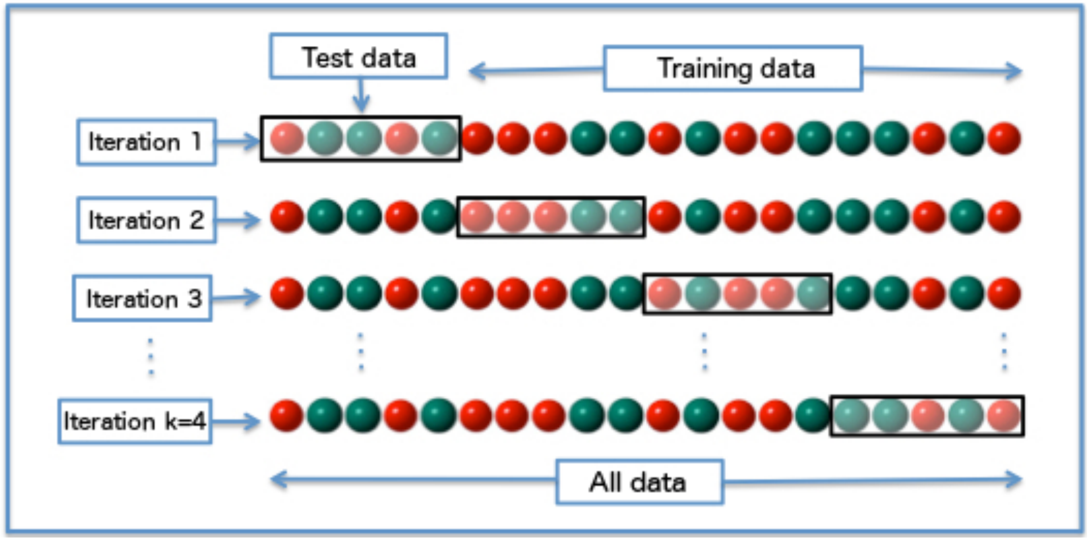
Data (Training, Validation, Test)

- Split data into three groups
  - Training: used to train a neural network (estimate weights)
  - Validation: used to monitor training progress and decide when to stop training process
  - Test: used to test performance and compare different models
- Rule of thumb: 50-25-25 to 60-20-20



Cross-Validation

- Common technique if amount of data is small
- N-fold cross validation
  - Divide data into N groups
  - Perform training with N-1 groups
  - Test trained model with remaining group
  - Repeat process while leaving out a different group for testing
  - Average Results
- Extreme case: leave-one-out



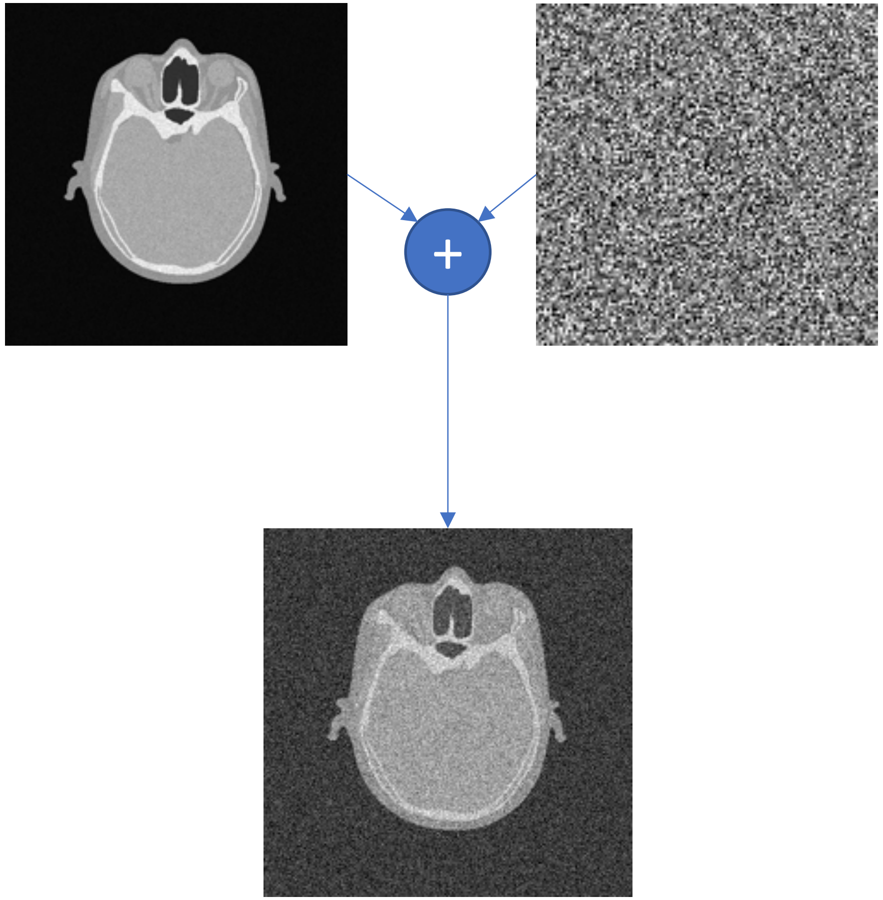
By Fabian Flöck [CC BY-SA 3.0 (https://creativecommons.org/licenses/by-sa/3.0)], from Wikimedia Commons

Regularization

- Parameter norm penalties (sparsifying weights)
  - L2-Norm $\|\boldsymbol{w}\|_2 = \sum_{\forall i} w_i^2$
  - L1-Norm $|\boldsymbol{w}|_1 = \sum_{\forall i} |w_i|$
  - Add regularization term to cost (“Loss”) function e.g. $L = \|\breve{\boldsymbol{y}} - \boldsymbol{y}\|_2 + \alpha\|\boldsymbol{w}\|_2$
- Early Stopping
- Bagging: Train multiple models (networks) and vote on output
- Dropout: Randomly exclude non-output nodes for each training sample
  - Approximates bagging

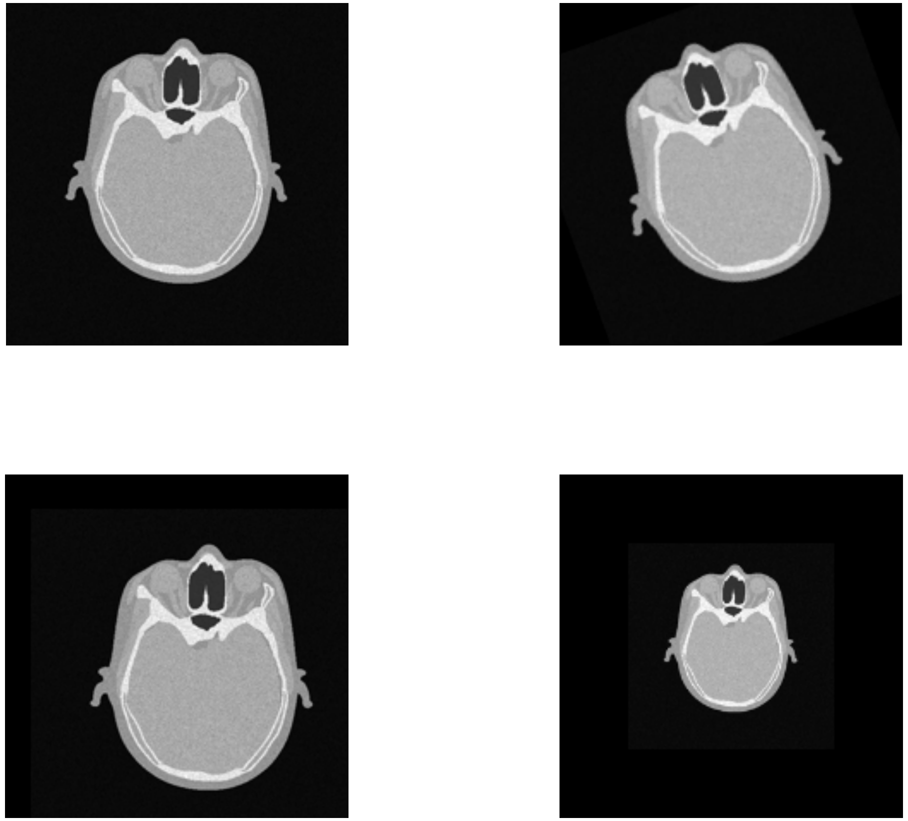
Dataset Augmentation

- Artificially increase amount of data by adding images multiple times with
    - Added noise
    - Geometric transforms (e.g. translation, rotation, scaling)
- Improves robustness of the trained model towards noise and transformations
- Reduces overfitting problem



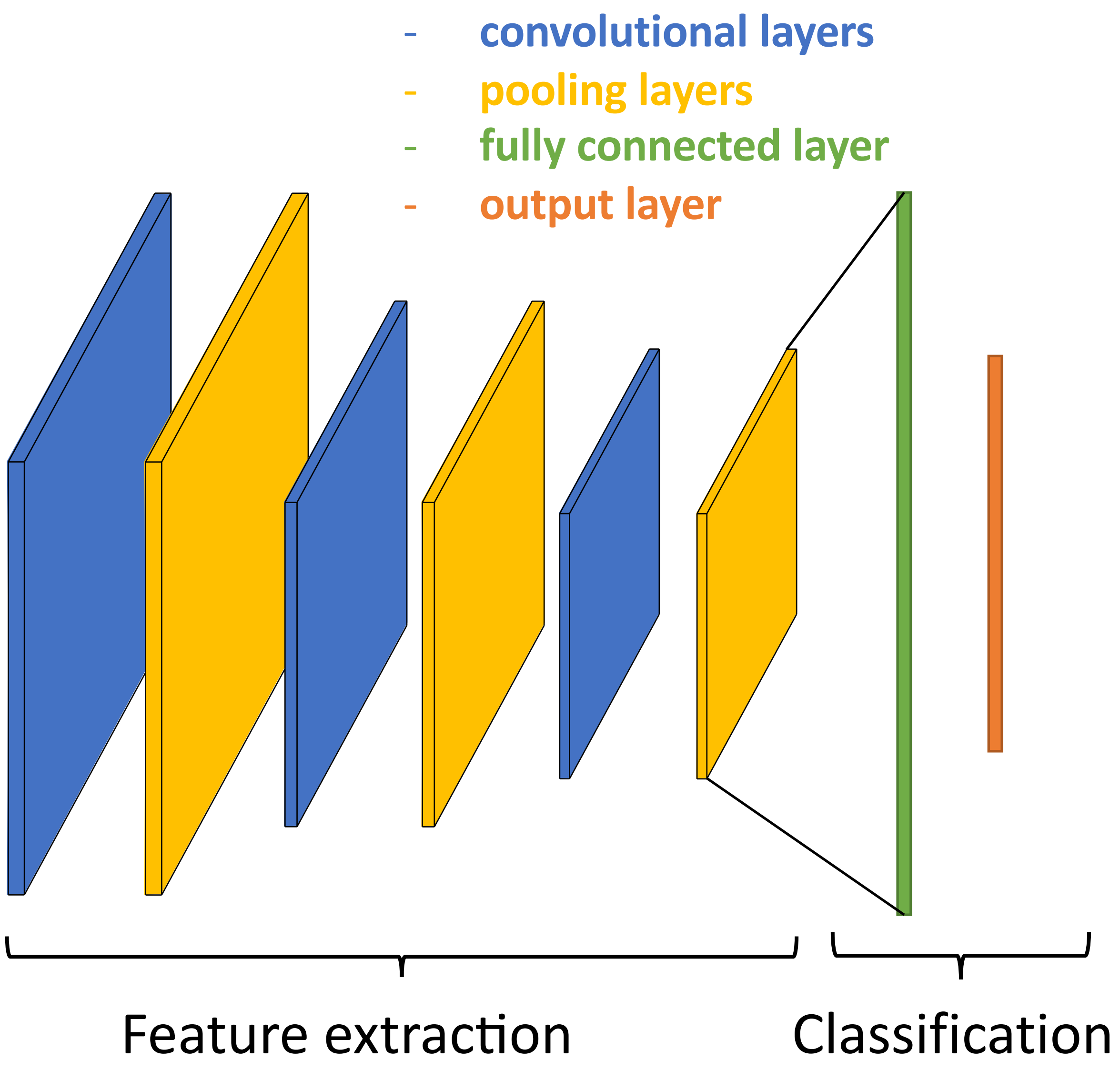
Dataset Augmentation

- Artificially increase amount of data by adding images multiple time with
  - Added noise
  - Geometric transforms (e.g. translation, rotation, scaling)
- Improves robustness of the trained model towards noise and transformations
- Reduces overfitting problem



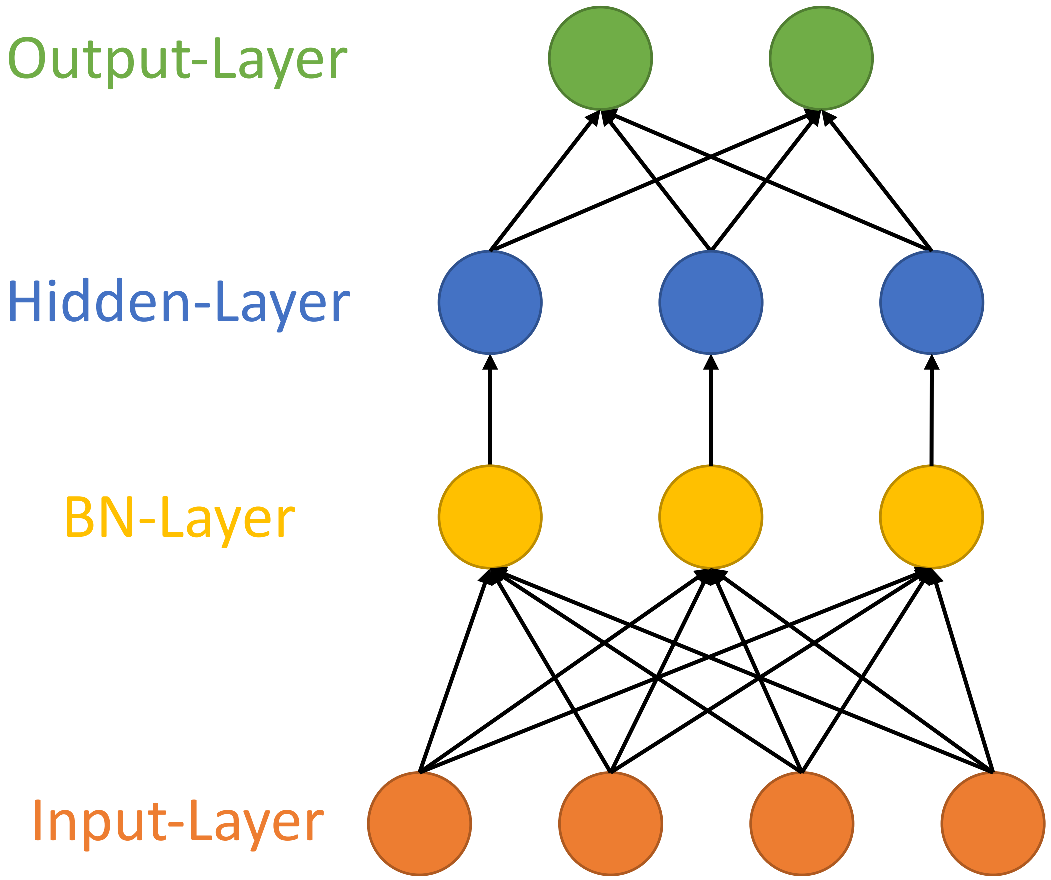
Transfer Learning

- Idea
  - Features for object recognition are similar for different tasks
  - Lower level layers in CNN determine image features
  - Object recognition / classification is performed in the last layer(s)
  - Reuse pretrained neural network
    - Same architecture and weights for lower level layers
    - Replace output layer by a new layer with appropriate size
    - Initialize weights randomly for the output layer
    - Retrain the model for the new task



Data (Batch) Normalization

- Additional layer
- Normalizes the input of each node of a layer by
  - Subtracting the batch mean
  - Dividing by batch standard deviation
- Goal: speed up learning process
  - Reduce internal covariance shift
  - Allows higher learning rates
  - Potentially decreases overfitting



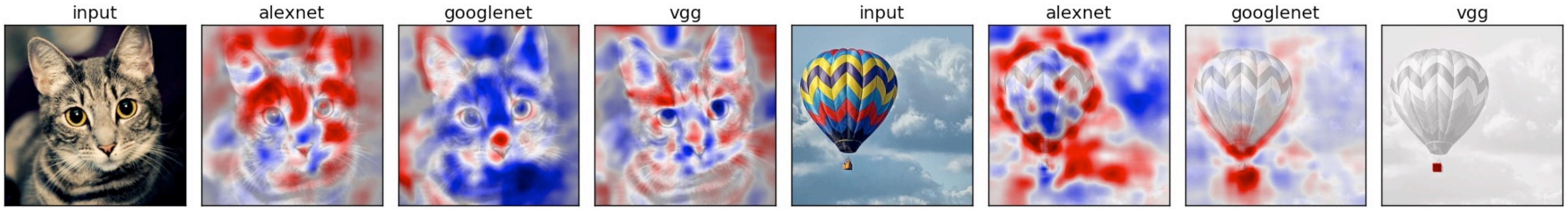
Analyzing Deep Learning Networks

Inside a Convolutional Neural Network

- Common critique: artificial neural networks are **black boxes**
  - Difficult to control what features the network learns
  - May learn inadequate features if input data is not representative of true population
- Understanding how neural networks learn may help
  - Improve the design of networks
  - Improve generalization
  - Make deep learning applications more secure
- Visualizing artificial neural networks is an active field of research

Visualization

- Zintgraf et al. 2017: for a given image detect areas which provide
    - evidence against each class
    - evidence for each class
    - Iteratively removes patches from the input data

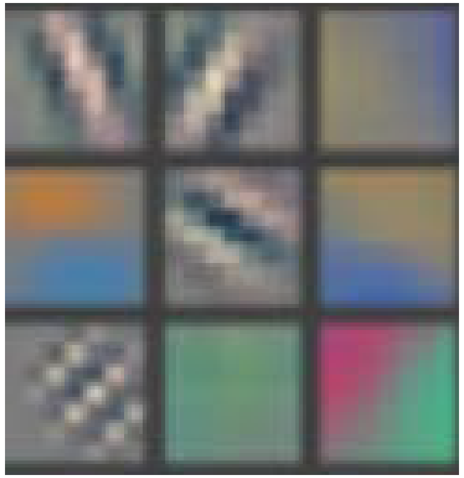


L. M. Zintgraf, T. S. Cohen, T. Adel, and M. Welling, “Visualizing Deep Neural Network Decisions: Prediction Difference Analysis,” Nov. 2016.

Visualization

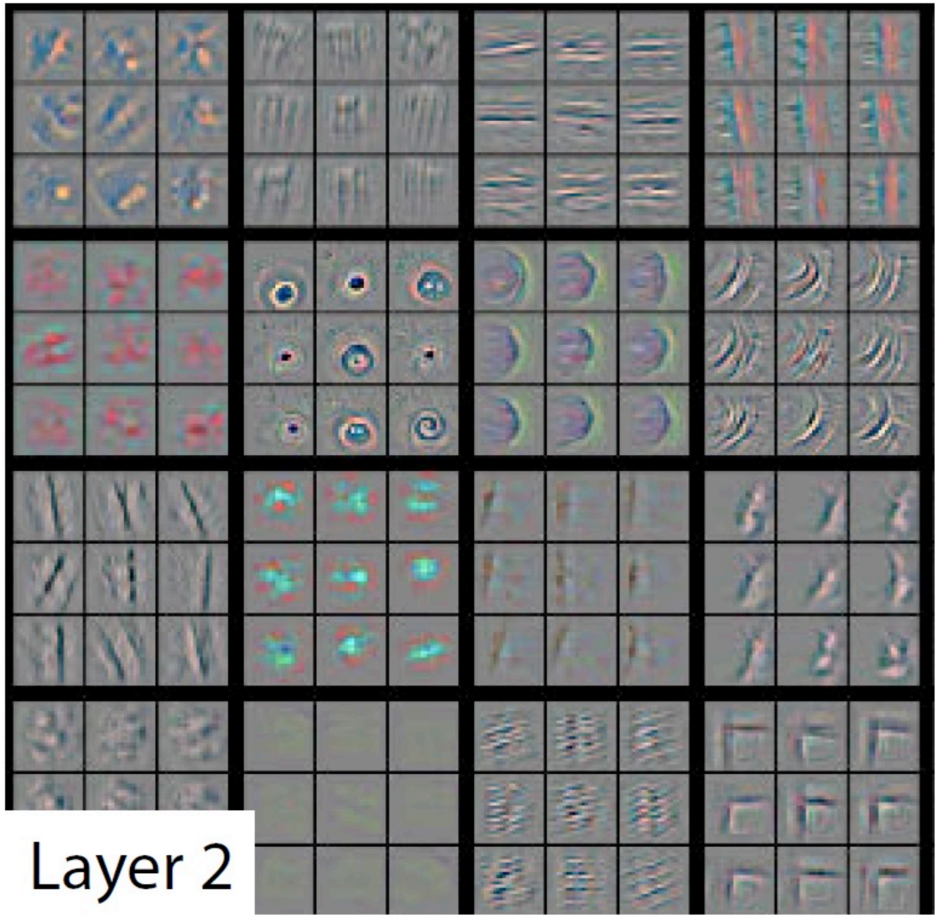
- Zeiler et al.
  - Uses deconvolutional neural network to invert operations
  - Max-pooling operations are not invertible instead algorithm follows location of maximum
  - Visualizes features at each level of the network

Layer 1



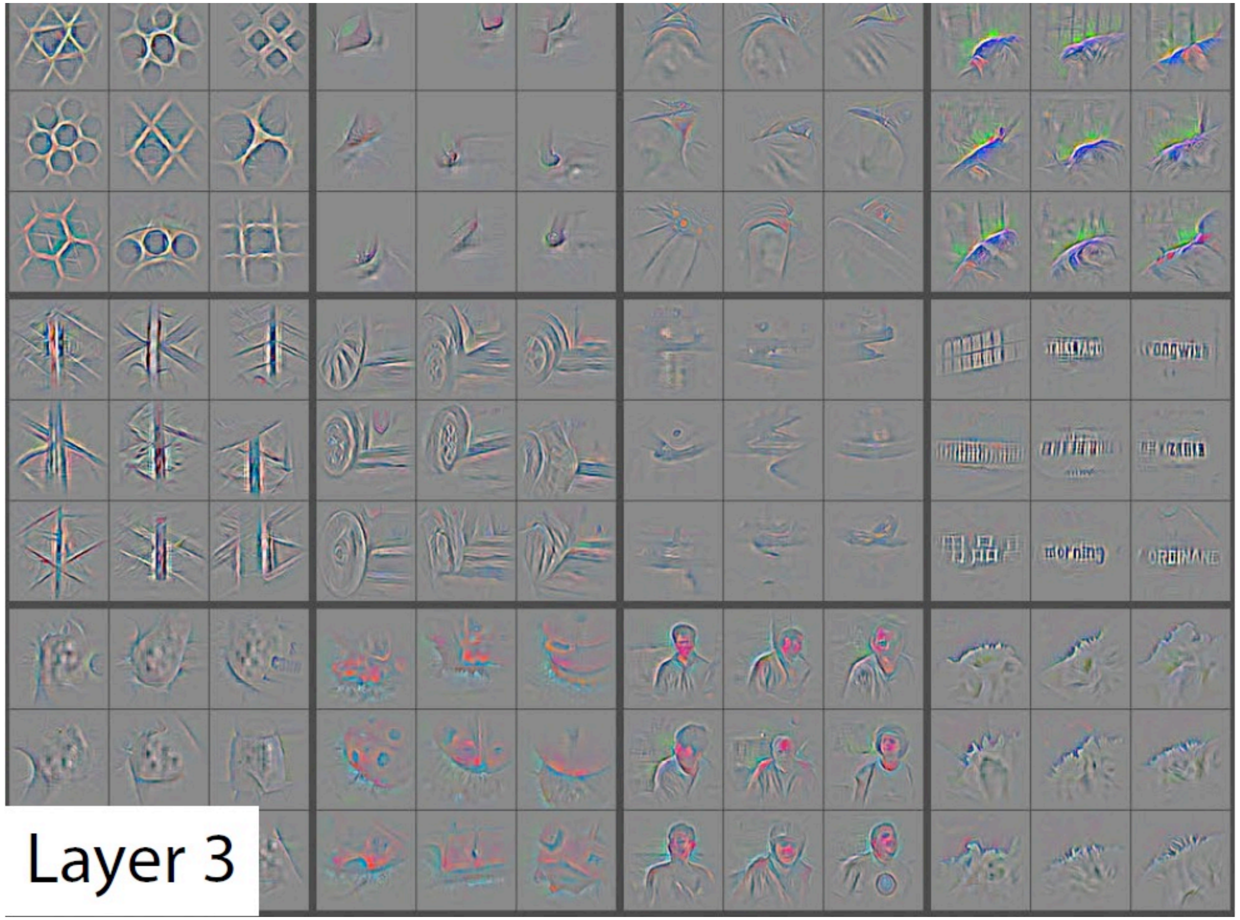
[1] M. D. Zeiler and R. Fergus, “Visualizing and Understanding Convolutional Networks,” in *Computer Vision – ECCV 2014*, vol. 8689, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, pp. 818–833.

Visualization



[1] M. D. Zeiler and R. Fergus, “Visualizing and Understanding Convolutional Networks,” in *Computer Vision – ECCV 2014*, vol. 8689, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, pp. 818–833.

Visualization



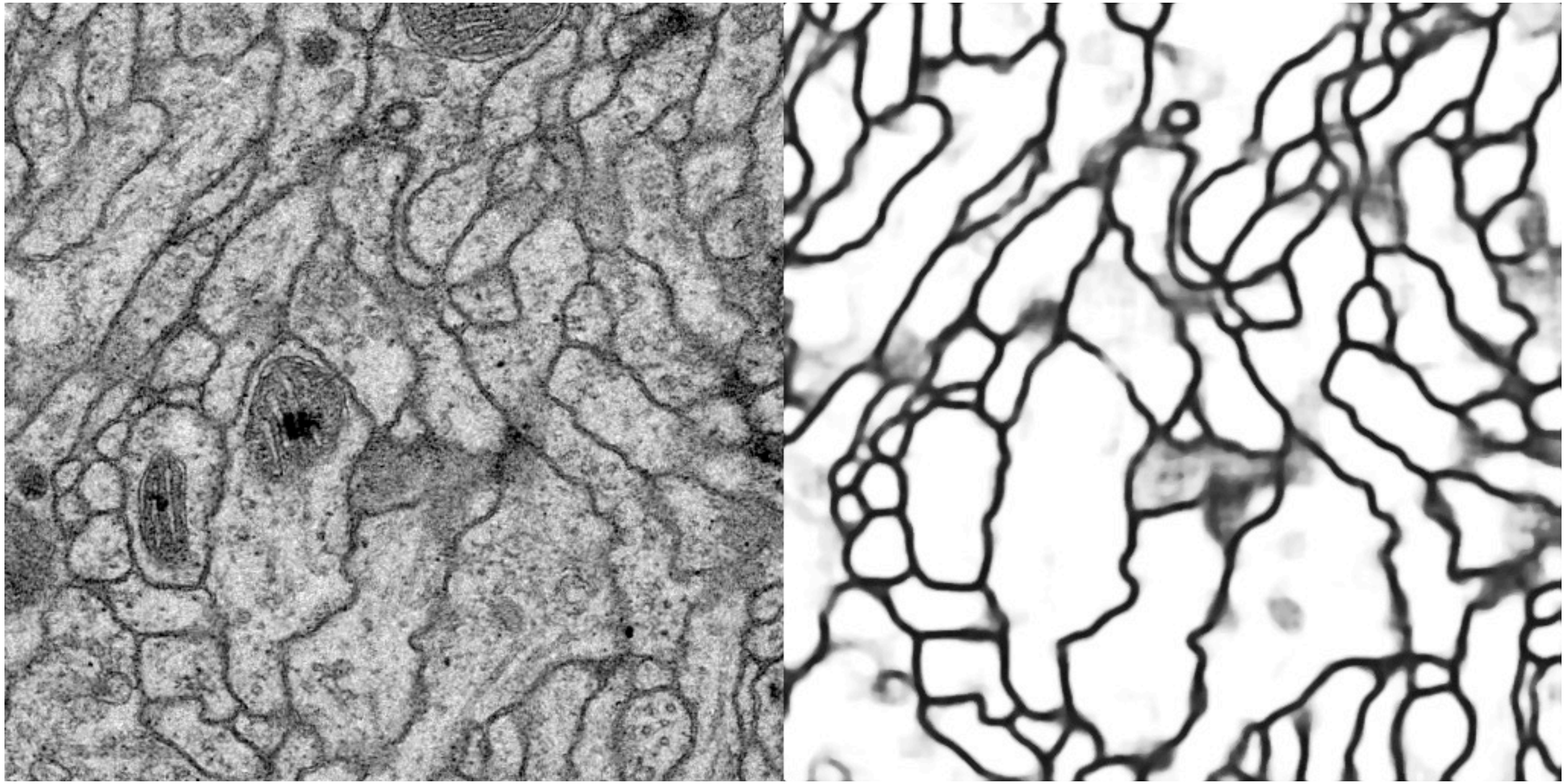
[1] M. D. Zeiler and R. Fergus, “Visualizing and Understanding Convolutional Networks,” in *Computer Vision – ECCV 2014*, vol. 8689, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, pp. 818–833.

Segmentation using Deep Learning

Segmentation: Patch Classification

- Classify each pixel of the image separately
- Use local patch around each pixel as input for deep learning network
- Common application:
  - Digital pathology
  - Microscopy
- Only includes local information
  - ➢Not suitable for semantic segmentation of larger complex objects

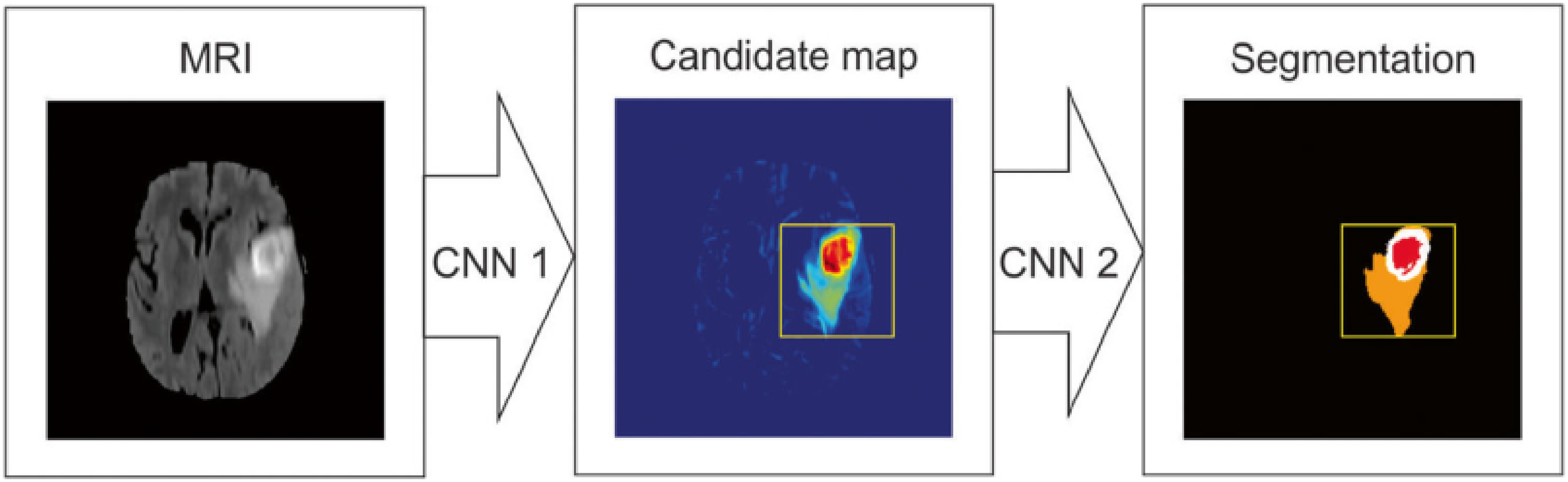
Example: Neuronal Membrane Segmentation



[1] D. C. Ciresan, L. M. Gambardella, and A. Giusti, “Deep Neural Networks Segment Neuronal Membranes in Electron Microscopy Images,” p. 9.

Cascaded CNN Architecture

- First network creates candidate map
- Second network refines segmentation

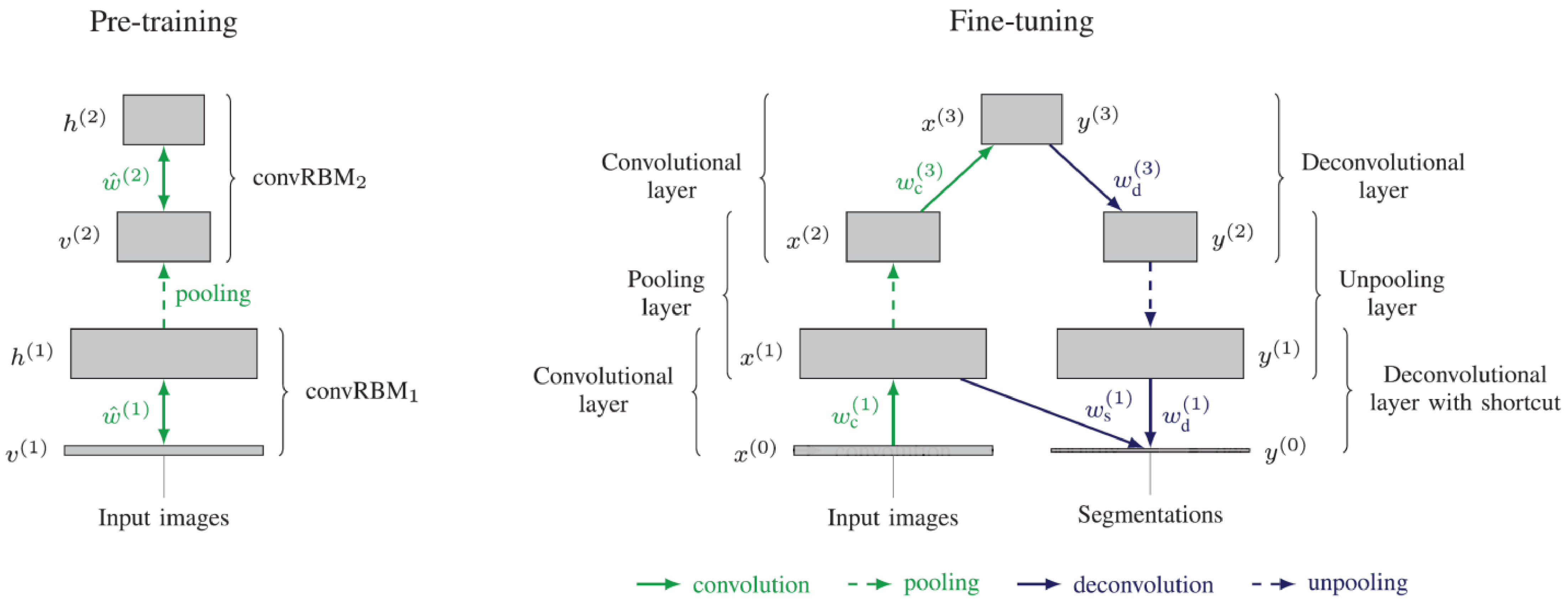


[1] Z. Akkus, A. Galimzianova, A. Hoogi, D. L. Rubin, and B. J. Erickson, “Deep Learning for Brain MRI Segmentation: State of the Art and Future Directions,” *J Digit Imaging*, vol. 30, no. 4, pp. 449–459, Aug. 2017.

Semantic-Wise CNN Architecture

- Deep learning architecture consists of
  - Encoder: convolutional neural network
  - Decoder: deconvolutional neural network
- Output is pixel-wise classification of the input image
- Can learn location dependent information
- Can segment large complex objects

Semantic-Wise CNN Architecture



[1] T. Brosch, L. Y. W. Tang, Y. Yoo, D. K. B. Li, A. Traboulsee, and R. Tam, "Deep 3D Convolutional Encoder Networks With Shortcuts for Multiscale Feature Integration Applied to Multiple Sclerosis Lesion Segmentation," *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1229–1239, May 2016.

Common Preprocessing Steps

- Registration to common anatomical space
- Bone extraction (skull stripping) may help focus segmentation process on soft tissue only
- Bias field correction: correction of image contrast variations due to magnetic field inhomogeneity
- Intensity normalization: similar histogram distributions across different images
- Noise reduction

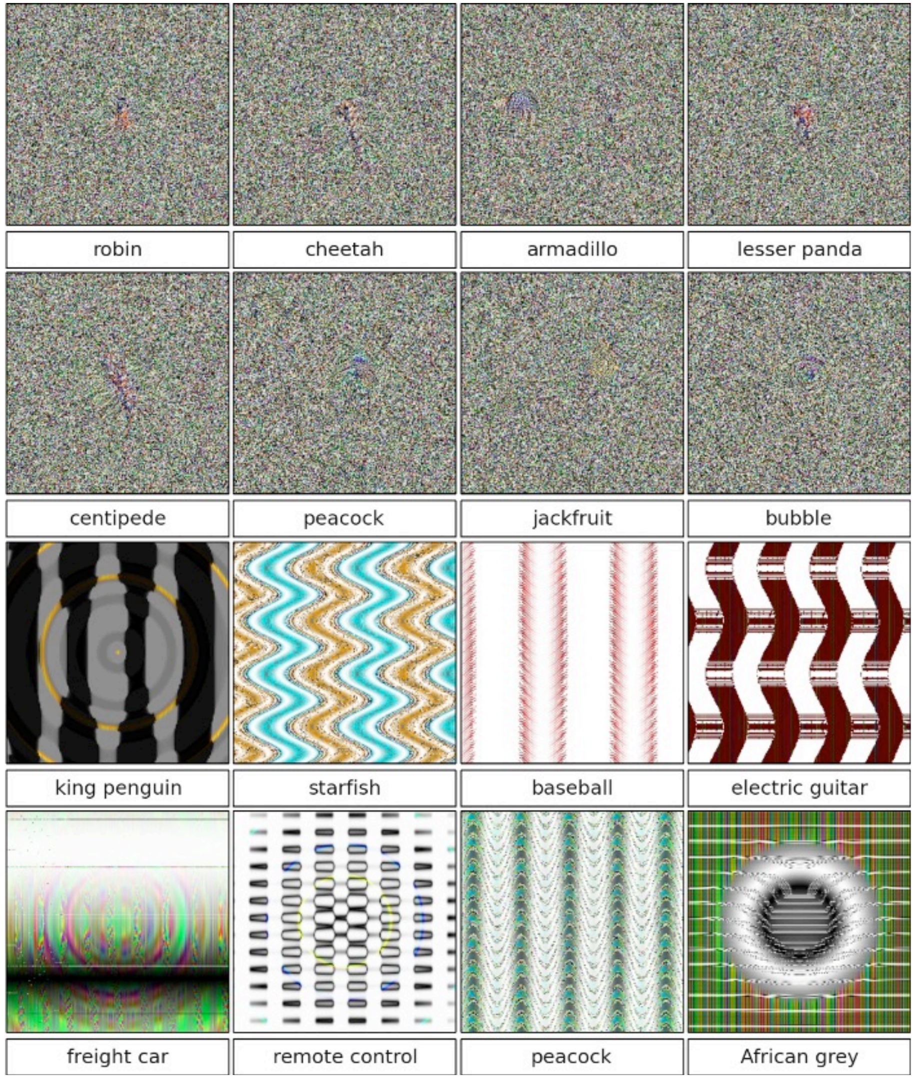
Summary

- Deeper understanding of underlying processes of deep learning
  - Network architecture
  - Training algorithm
  - Mathematics behind deep learning
- Many high-level libraries available
  - Understanding the background helps deciding for
    - Network type and architecture
    - Loss functions
    - Activation functions
  - Understanding the importance of input data

Different Types of Deep Learning Networks

Fooling Deep Learning Networks

- Deep convolutional neural networks can often be fooled by textures are noise
- Security risk for
  - Biometric identification (e.g. face recognition)
  - Spam filters
  - Object detection in security cameras



[1] A. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," 2015, pp. 427–436.

Generative adversarial networks

- Introduced by Ian Goodfellow 2014
- Set of two neural networks
  - Generative model: creates artificial input data
  - Discriminative model: trained to discriminate between true and artificial data
- Discriminative model usually pretrained using real data
- Input for generator may be
  - Noise images
  - Text description of the image contents
  - …

Generative adversarial networks

Salimans, T., Goodfellow, I.J., Zaremba, W., Cheung, V., Radford, A., & Chen, X. (2016). Improved Techniques for Training GANs. *NIPS*.

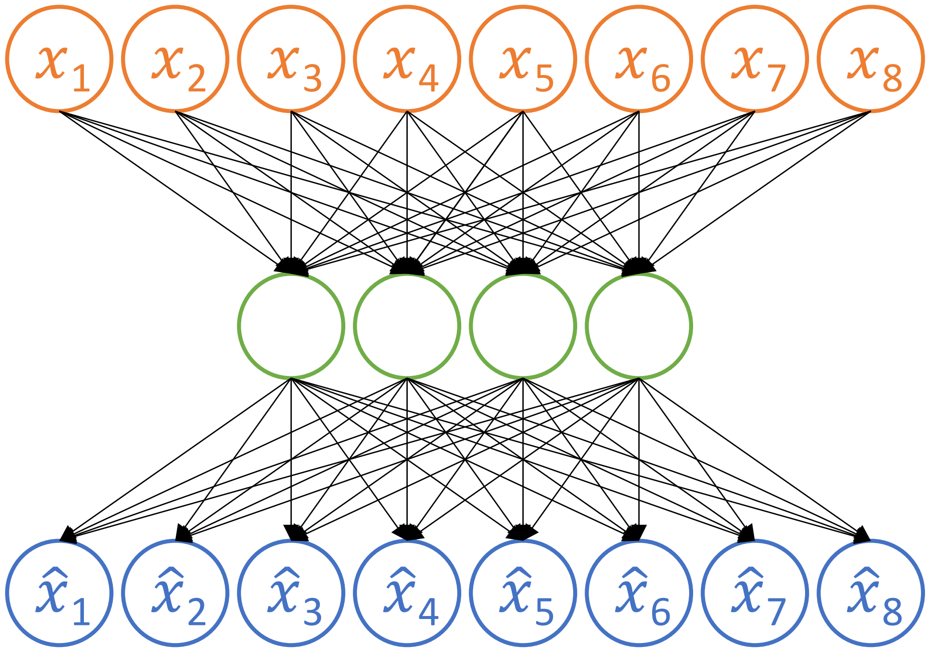
H. Zhang, T. Xu, and H. Li, “StackGAN: Text to Photo-Realistic Image Synthesis with Stacked Generative Adversarial Networks,” in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 5908–5916.

Unsupervised Deep Learning

- Labeling images by hand can be time consuming and expensive
- Unsupervised: no labeled images are available for training
- **Theory**: deep neural networks learn by removing irrelevant information in each layer
- Common examples: clustering, estimation of probability distributions
- Unsupervised deep learning:
  - Train a neural network to learn relevant features to describe a set of images

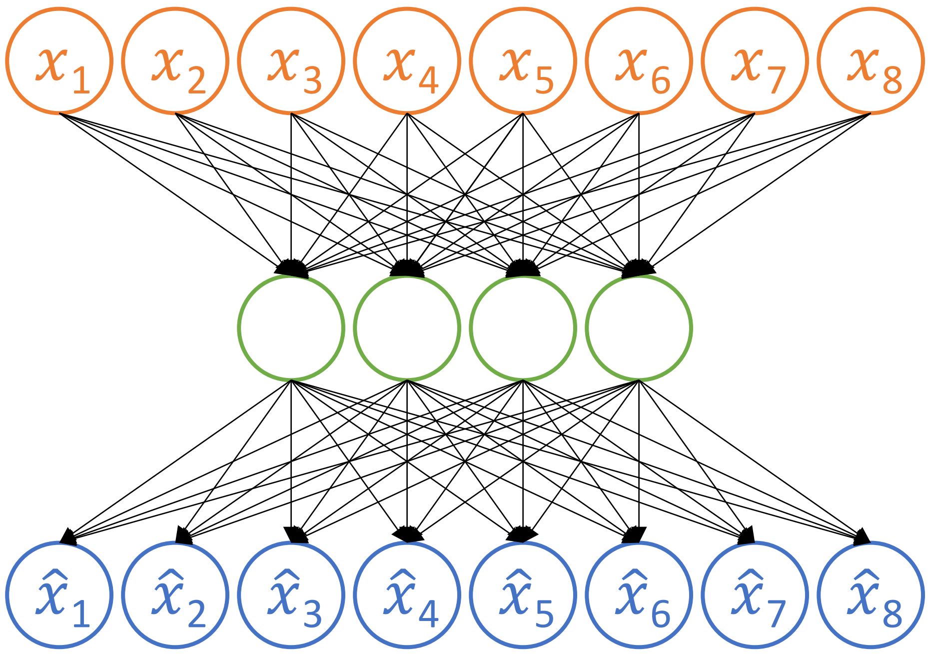
Unsupervised Learning: Autoencoder

- Optimize a network to learn the identity transform
  - Output $\hat{\boldsymbol{x}}$ should be as close as possible to input $\boldsymbol{x}$
- Hidden layer(s) have less nodes than input and output layers
- Forces network to remove irrelevant information
- Learn only relevant features
- Encoder, decoder scheme similar to image compression



Self-Taught Learning

- Use autoencoder to learn relevant features
- Remove decoder part of autoencoder
- Add new set of output layers
- Train the network with a small set of labeled images
- Similar to transfer learning

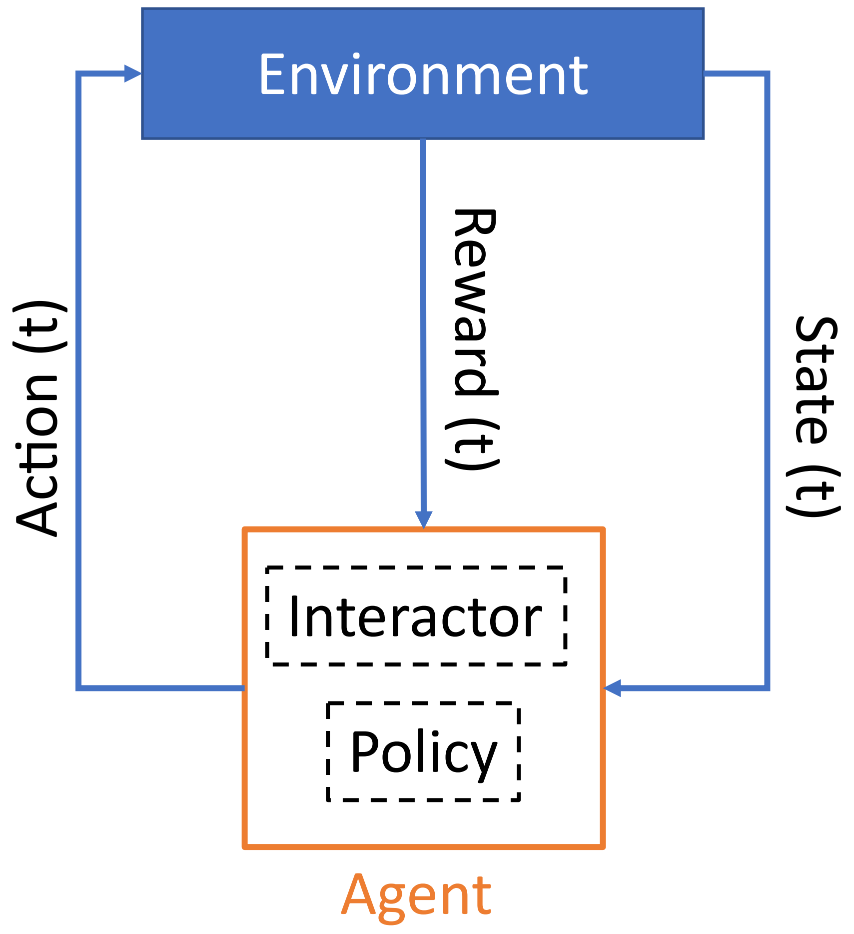


Whitening

- Reduce correlation between adjacent blocks of pixels
  - Uncorrelated (orthogonal) basis
  - Equal variance in all directions
- Principle component analysis
  - $\Sigma = \sum_{\forall i} \boldsymbol{x}_i \boldsymbol{x}_i^\intercal$
  - Eigenvectors of Σ provide reduced orthogonal basis
- ZCA (Mahalanobis transform)
  - Similar to PCA but rotated to be as close as possible to the original data
- Independent component analysis
  - Maximizes statistical separation between basis vectors

Reinforcement Learning

- Learning through interaction
  - Interacts with environment
  - Observes consequences
  - Alters its behavior in response to rewards received
- Has to learn by trail and error
  - perception-action learning loop
  - Markov decision process
- e.g. learn to play video games directly from pixels

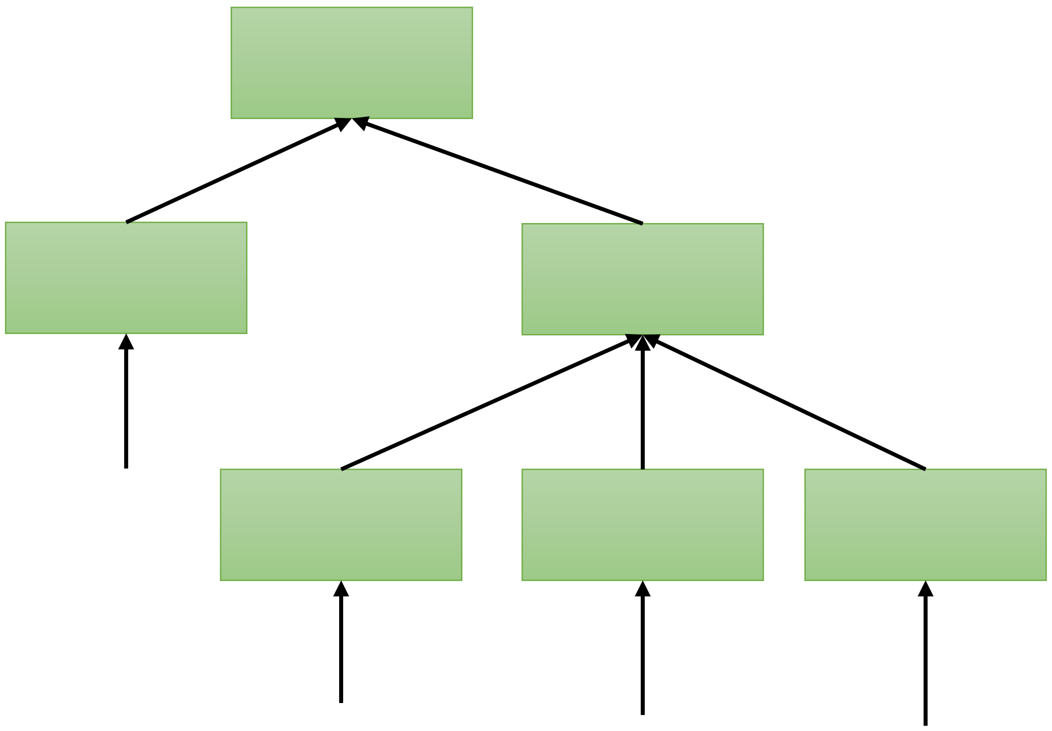


Reinforcement Learning

- Challenges:
  - No ground truth known (only feedback is from rewards)
  - Strong temporal correlations
  - Long range time dependencies
- Value Functions
  - Function that describes the value of a policy given an initial state s
  - The policy which returns the highest value is ‘optimal’
  - Often replaced by Quality function which uses initial action
- Policy Search
  - Aims to find policy directly
- Example: AlphaGo
  - Initial training by supervised learning
  - Policy-gradient reinforcement learning
  - Combined using Monte-Carlo tree search

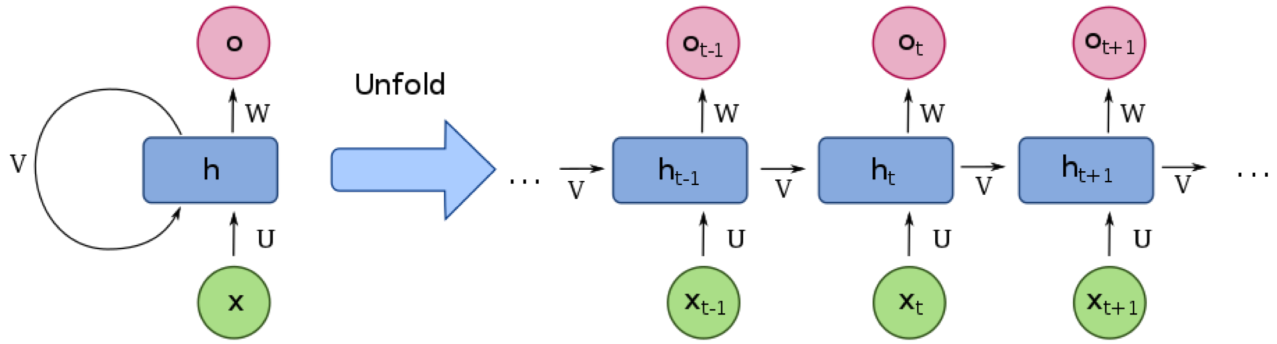
Recursive Neural Networks

- Applies same weights recursively over hierarchical structure
- Used for natural language processing
  - E.g. analyze hierarchical structure of sentences



Recurrent Neural Networks (RNN)

- Special case of recursive neural networks
- Each hidden layer takes
  - Current input
  - Result of previous iteration
- Used for time sequences
  - E.g. natural language processing
- RNNs have a “memory”



Long Short-Term Memory (LSTM)

- Building block for recurrent neural networks (LSTM unit)
- Composed of
  - Cell state
  - Input gate
  - Output gate
  - Forget gate
- Can remember values over arbitrary long time intervals