CSE 40171: Artificial Intelligence

Artificial Neural Networks: Neural Network Architectures
Homework #1 has been released
It is due at 11:59PM on 9/16
Architectures
Convolutional Neural Net

Deep (sparse/denoising) Autoencoder

Deep Belief Net

Recurrent Neural Net

Supervised

Unsupervised

Deep

SHALLOW

SVM

Boosting

Perceptron

SP

BayesNP

Slide: M. Ranzato
Long / Short Term Memory

- Unit of a recurrent network
- Used to process time series data
- Each neuron has a memory cell and three gates: input, output and forget
- Gates and an explicitly defined memory address vanishing / exploding gradient problem
Long / Short Term Memory

Legend:

Layer

Pointwise op

Copy

The LSTM Cell © BY-SA 4.0 Guillaume Chevalier
Example Application: Handwriting Recognition

Image Credit: Graves and Schmidhuber NIPS 2009
Autoencoder

- Feed-forward network
- Encode (i.e., compress) information, and then decode (i.e., reconstruct) it from the compressed representation
- Architecture is always symmetric
- Can be built such that encoding weights and decoding weights are the same
Autoencoder Example: MNIST

Image Credit: https://blog.keras.io/building-autoencoders-in-keras.html
Variational Autoencoder

- Feed-forward network
- Have the same architecture as AEs, but are taught something else: approximated probability distribution of input samples
- Take influence into account
  - If one thing happens in one place and something else happens somewhere else, they are not necessarily related
- Rule out influence of some units to other units

Slide Credit: Fjodor van Veen
http://www.asimovinstitute.org/neural-network-zoo/
Variational Autoencoder

[Diagram of a Variational Autoencoder]

Image Credit: Rebecca Vislay Wade
Variational Autoencoder Example

$X_t = [x_{1,t}, x_{2,t}, x_{3,t}, \ldots, x_{n,t}]$

$\hat{X}_t = [\hat{x}_{1,t}, \hat{x}_{2,t}, \hat{x}_{3,t}, \ldots, \hat{x}_{n,t}]$

Image Credit: Deng et al. CVPR 2017
Denoising Autoencoder

- Feed-forward network
- Also the same AE architecture, but we feed the input data with noise
- Error is computed the same way, however
  - Output of the network is compared to the original input without noise
Denoising Autoencoder
Example: MNIST

Image Credit: opendeep.org
Denoising Autoencoder Example: CIFAR10
Hopfield Network

- Recurrent network
- Every neuron is connected to every other neuron
- Each node is input before training, then hidden during training and output afterwards
- Network is trained by setting the value of the neurons to the desired pattern after which the weights can be computed. The weights do not change after this.

Slide Credit: Fjodor van Veen
http://www.asimovinstitute.org/neural-network-zoo/
Energy Landscape of Hopfield Network
Associated Learning Rule

“Let us assume that the persistence or repetition of a reverberatory activity (or "trace") tends to induce lasting cellular changes that add to its stability. […] When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased.”

Hebb. The Organization of Behavior 1949
Hopfield Network’s Associative Memory

- Incorporation of memory vectors
- Two types of operations
  - Auto-association
  - Hetero-association
- Possible to store both types of operations in a single memory matrix

Image Credit: Fjodor van Veen
http://www.asimovinstitute.org/neural-network-zoo/
Reconstruction of noisy input of learned image

Image Credit: http://fourier.eng.hmc.edu/e161/lectures/nn/node5.html
Boltzmann Machine

- Recurrent network
- Very similar to HNs, but some neurons are marked as input and other are hidden
- The input neurons become output neurons at the end of a full network update
- It starts with random weights and learns through back-propagation, or more recently through contrastive divergence (leveraging Markov chains to compute gradients)
- Networks are stochastic

Slide Credit: Fjodor van Veen
http://www.asimovinstitute.org/neural-network-zoo/
Restricted Boltzmann Machine

- In practice, Boltzmann machines are rarely used; RBMs address shortcomings
- Not fully connected, only connect every different group of neurons to every other group
  - No input neurons are directly connected to other input neurons and no hidden to hidden connections exist either
- Training: forward pass the data and then backward pass the data (back to the first layer). After that train with forward-and-back-propagation.
Restricted Boltzmann Machine
Example: MNIST

Filters obtained after 15 epochs

Samples generated from an RBM model after training

Image Credit: http://deeplearning.net/tutorial/rbm.html
Echo State Network

- Recurrent network
- Randomly instantiated architecture
- Training: feed the input, forward it and update the neurons for a while, and observe the output over time
  - Only the connections between the observer and the (soup of) hidden units are changed
- Input layer is used to prime the network and the output layer acts as an observer of the activation patterns over time

Slide Credit: Fjodor van Veen
http://www.asimovinstitute.org/neural-network-zoo/
Instance of Reservoir Computing

Image Credit: Haj Mosa et al. Neurocomputing 2016
Liquid State Machine

- Recurrent network; very similar to ESN
- Major difference: spiking neural network
  - Sigmoid activations are replaced with threshold functions and each neuron is also an accumulating memory cell
- When updating a neuron, the value is added to itself
- Once the threshold is reached, it releases its energy to other neurons

Slide Credit: Fjodor van Veen
http://www.asimovinstitute.org/neural-network-zoo/
Extreme Learning Machine

- ELMs look like ESNs and LSMs, but are not spiking or recurrent
- Are not trained with backpropagation
- Training procedure: start with random weights and train in a single step according to a least-squares fit
- Pro: Fast to train
- Con: Weaker representational capabilities
Neural Turing Machine

- Abstraction of LSTMs
- Attempt to un-black box neural networks
- Memory is explicitly separated
- This type of network is Turing complete
Architectures informed by the brain?
Is building an architecture by hand the best way to do this?
Is optimizing hyperparameters by hand the best way to do this?
Problem: How do we search intractably large spaces?