CS 5043: Advanced Machine Learning

Andrew H. Fagg

Symbiotic Computing Laboratory

School of Computer Science

What is Machine Learning?

Classes of Machine Learning Problems

Classes of Machine Learning Problems

Supervised learning:

- Training set contains only input / output (labels) pairs
- Outputs could be continuous, probabilistic or categorical

Classes of Machine Learning Problems

Unsupervised learning:

- The training set contains only inputs
- Fundamental question: what is the structure of these inputs?
  - A common case: algorithm assigns categorical labels to each sample
  - But we can also ask continuous questions. For example: are there linear or nonlinear manifolds that the data live on?

Classes of Machine Learning Problems

Reinforcement learning:

- Different than direct prediction or classification: RL is about taking sequences of actions in some environment
- At each step:
  - In response to an input, the model (agent) produces some action
  - The feedback signal is an evaluation of the results of this and previous actions

Classes of Machine Learning Problems

Reinforcement learning:

- A common case: a single evaluation can be a function of the sequence of outputs that is generated
  - How much time did it take to solve a task?
  - How much energy did you use while solving the task?

- Learning problem: for a given input, what is the output that maximizes the expected reinforcement over time?

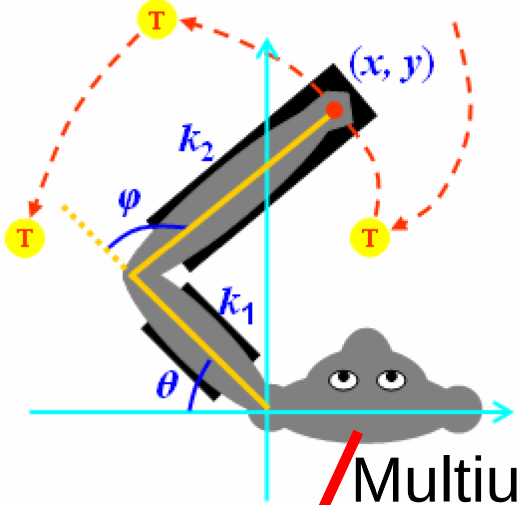
What is Machine Learning?

- Fundamentally: using data to automatically construct models
- The models must be predictive!
  - I.E.: to be useful, it must produce meaningful output given novel situations.

Brain-Machine Interfaces

Command prosthetic arm

Estimate of intended movement



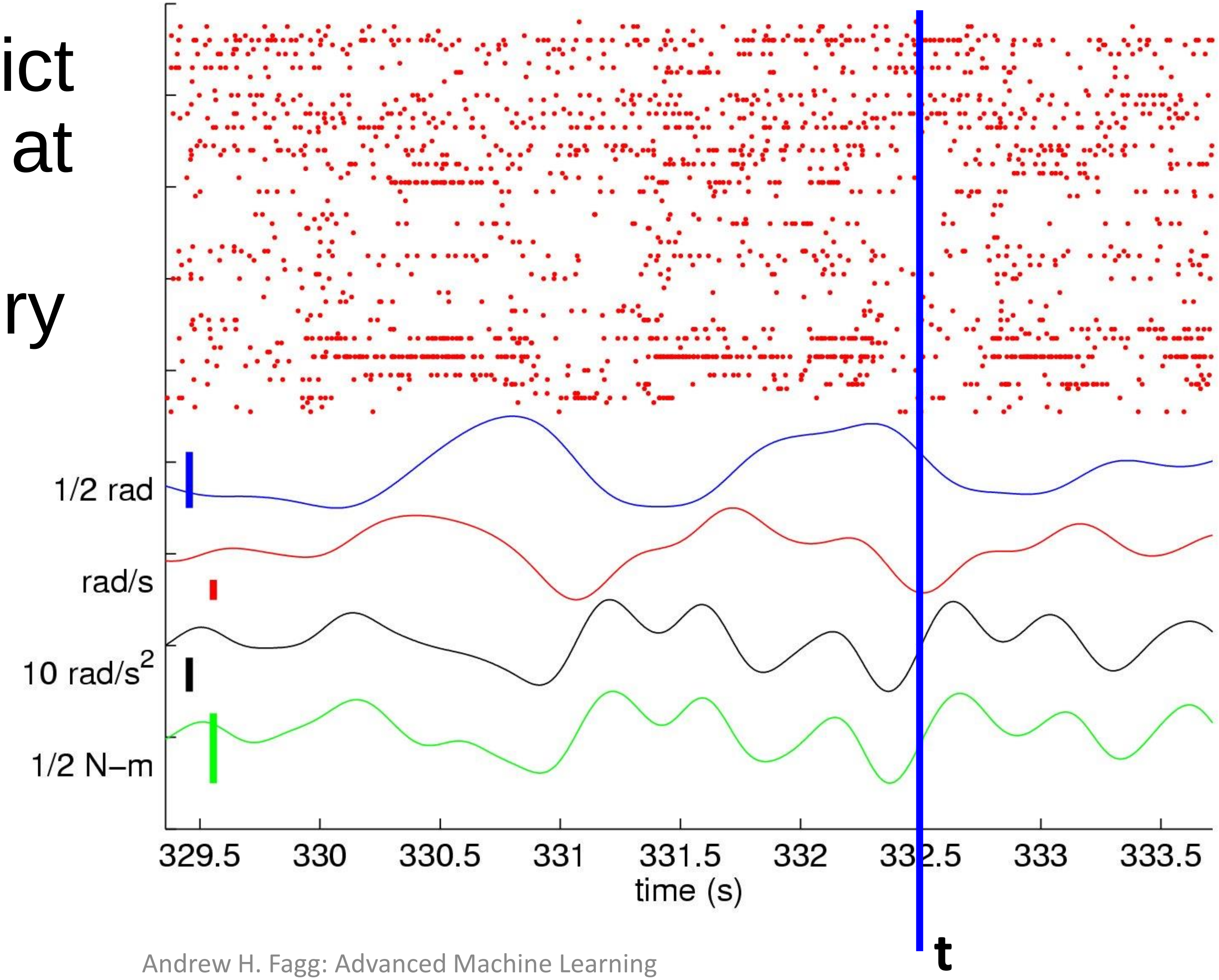
Multiunit recording

Predictive model

In collaboration with Nicholas G. Hatsopoulos and Lee E. Miller

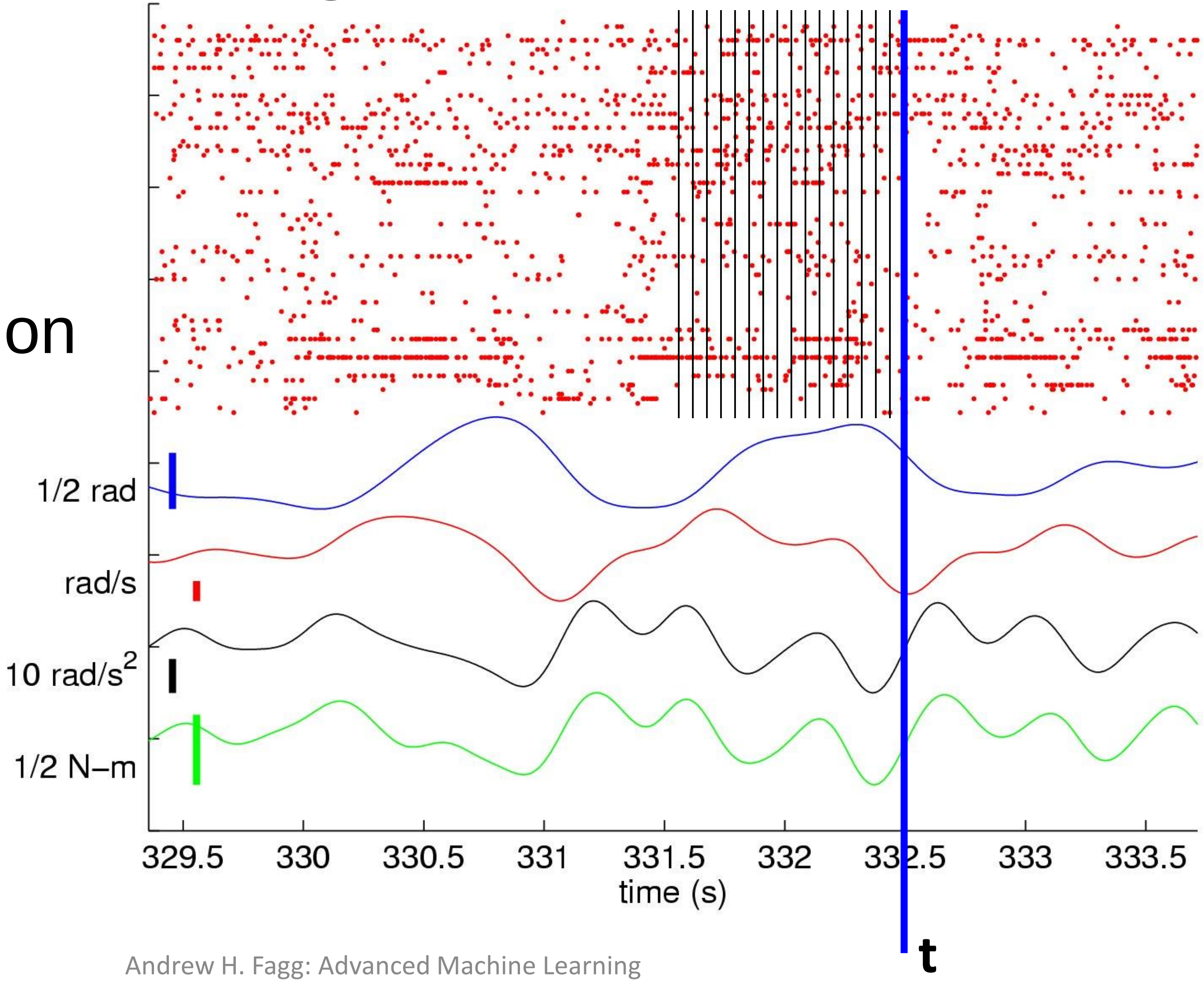
Decoding Arm State

Want to predict arm motion at time t given recent history of spiking behavior



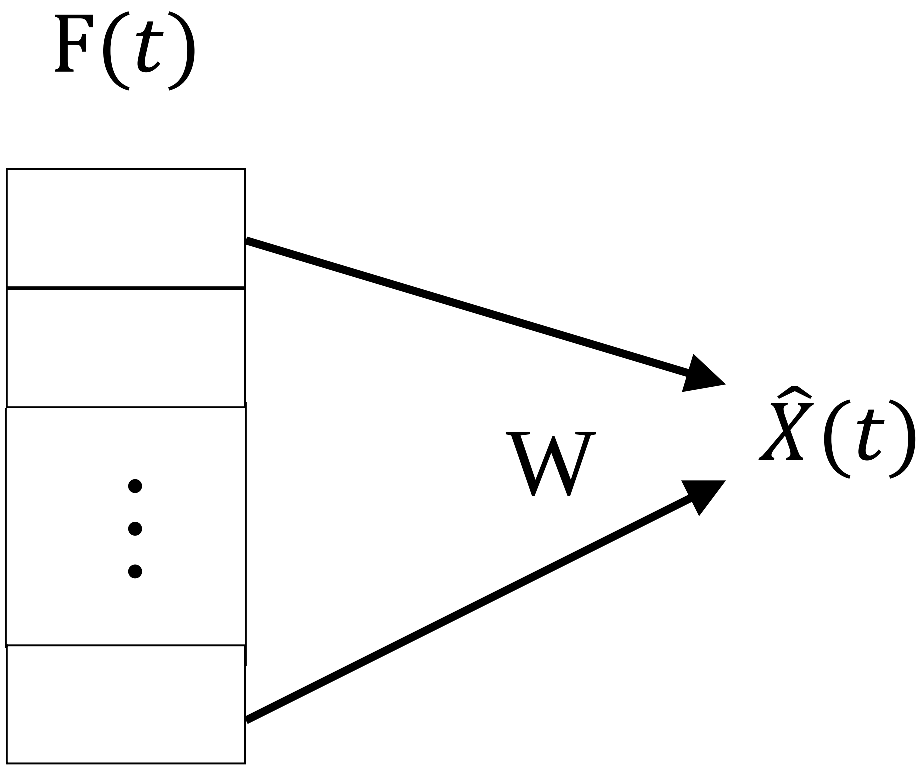
Decoding Arm State

50ms bins: 20 descriptors of neural activation for each cell



Wiener Filter

Each feature (F_i) is a count of spikes by a neuron for a 50 ms bin



$$\hat{X} = g_W\big(F(t)\big) = W^T F(t)$$

Column vector encoding spike counts for N cells at T taps up to time t

Computing a “Good” Model

Must define what we mean by good

- Common for this case:
  - Compute the sum of the squared prediction errors
  - Choose the parameters so as to minimize this error metric

    Least Mean Squared (LMS) error

Computing a “Good” Model

- For BMI problem, we typically have:
  - ~1000 parameters
  - ~1000-20,000 examples
- Easy for LMS to overfit these models
  - Great performance on the training data
  - But … poor performance on independent data
- One approach: modify the error function to punish large magnitude parameters
  - Regularization!

Even Harder Problems…

Let’s consider higher-dimensional inputs (e.g., 10K, 100K variables) …

Image Understanding

How do we make sense of natural scenes?

Schulter, et al. (2018)

Image Understanding

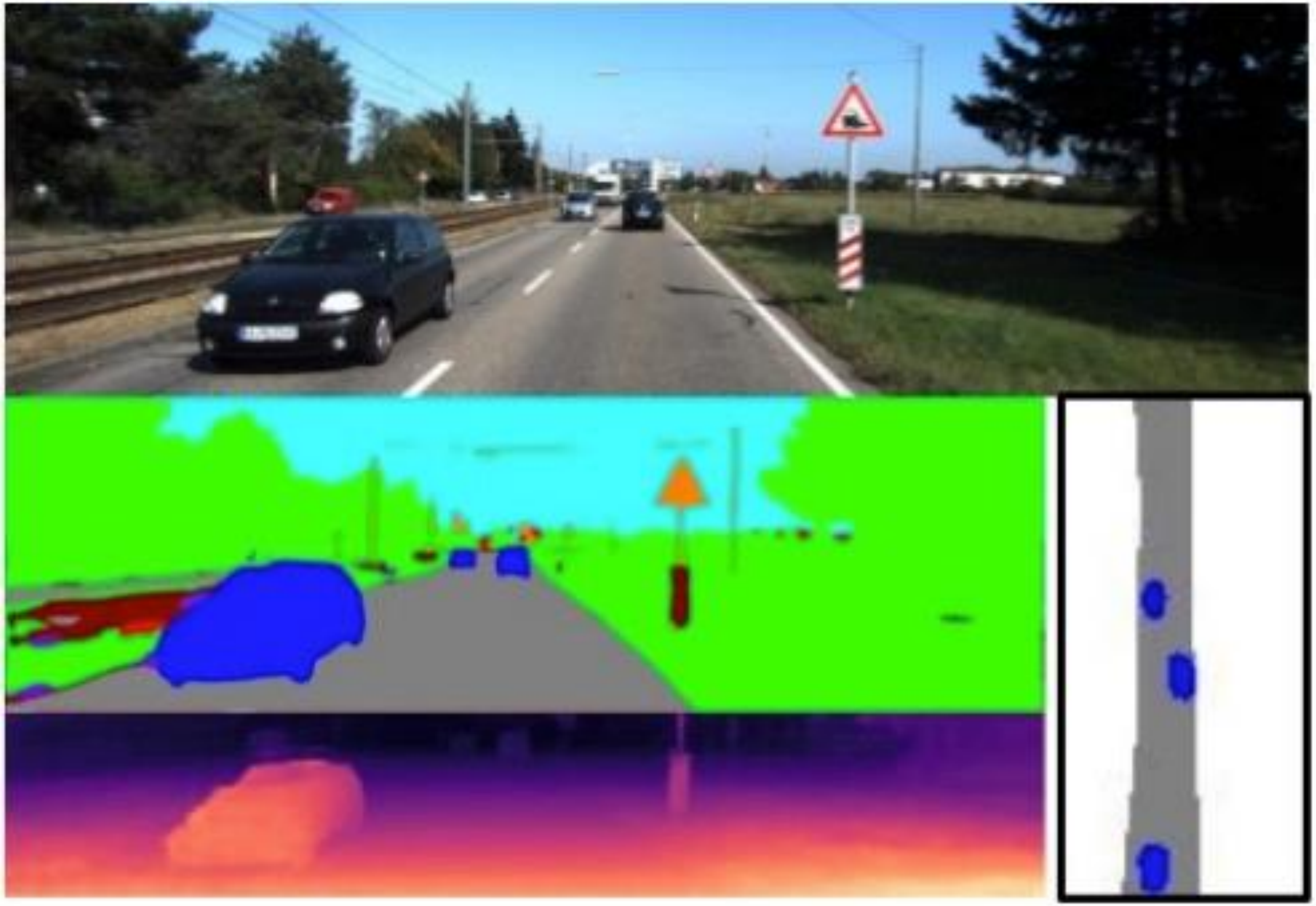


Image Understanding

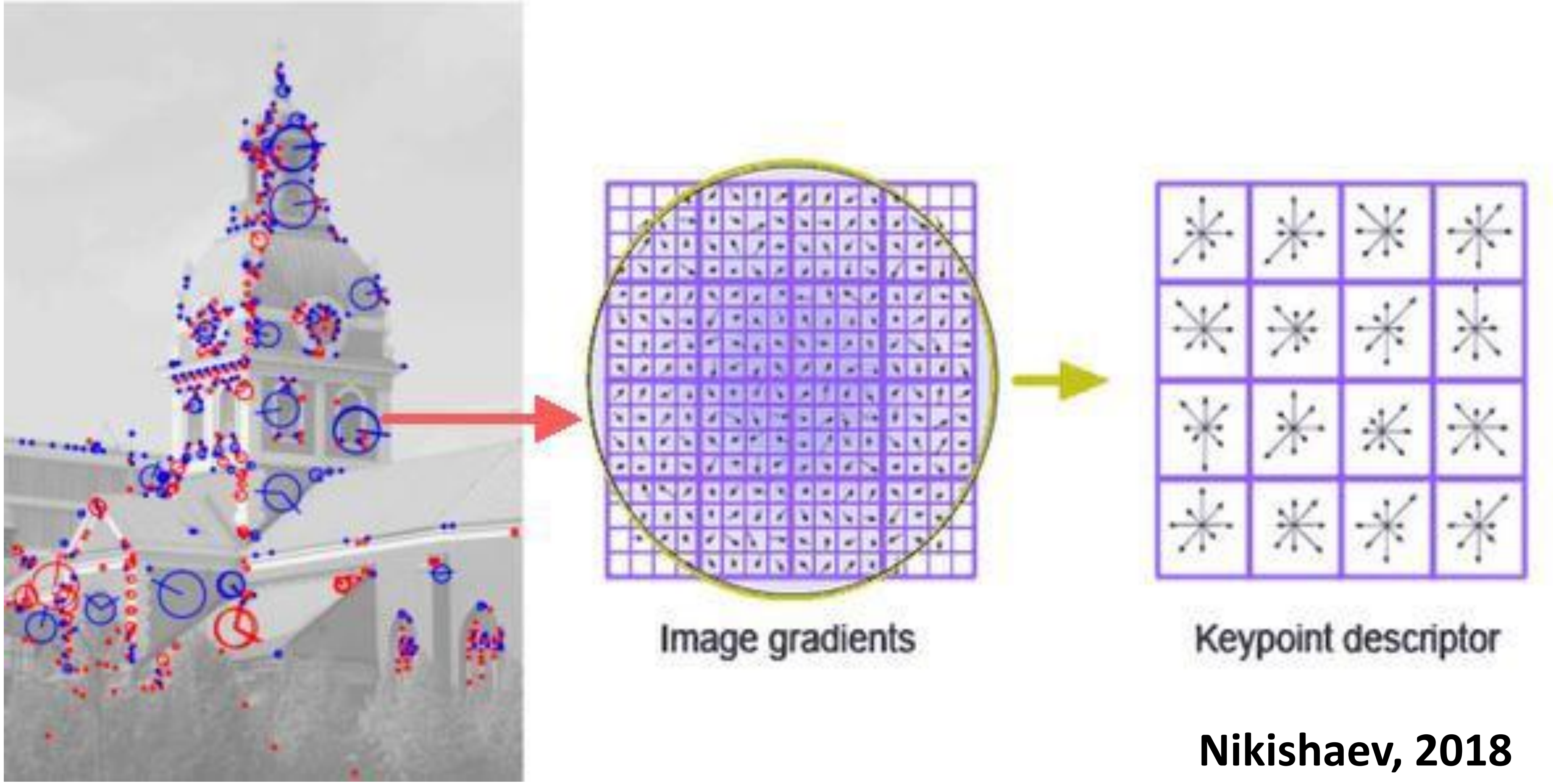
- High-dimensional inputs
- Many different cases
- Even a linear model would require a large number of parameters
- Big risk of overfitting

Computer Vision

Much of computer vision has been about hand-crafting feature detectors that allow us to extract just the right information….

- Edges, corners, blobs, …
- Many have been hand-tuned
- Identified features allow us to summarize an image with a smaller number of degrees-of-freedom
- It is then feasible to use ML to effectively learn the parameters of the simpler model

Computer Vision: SIFT Features



Nikishaev, 2018

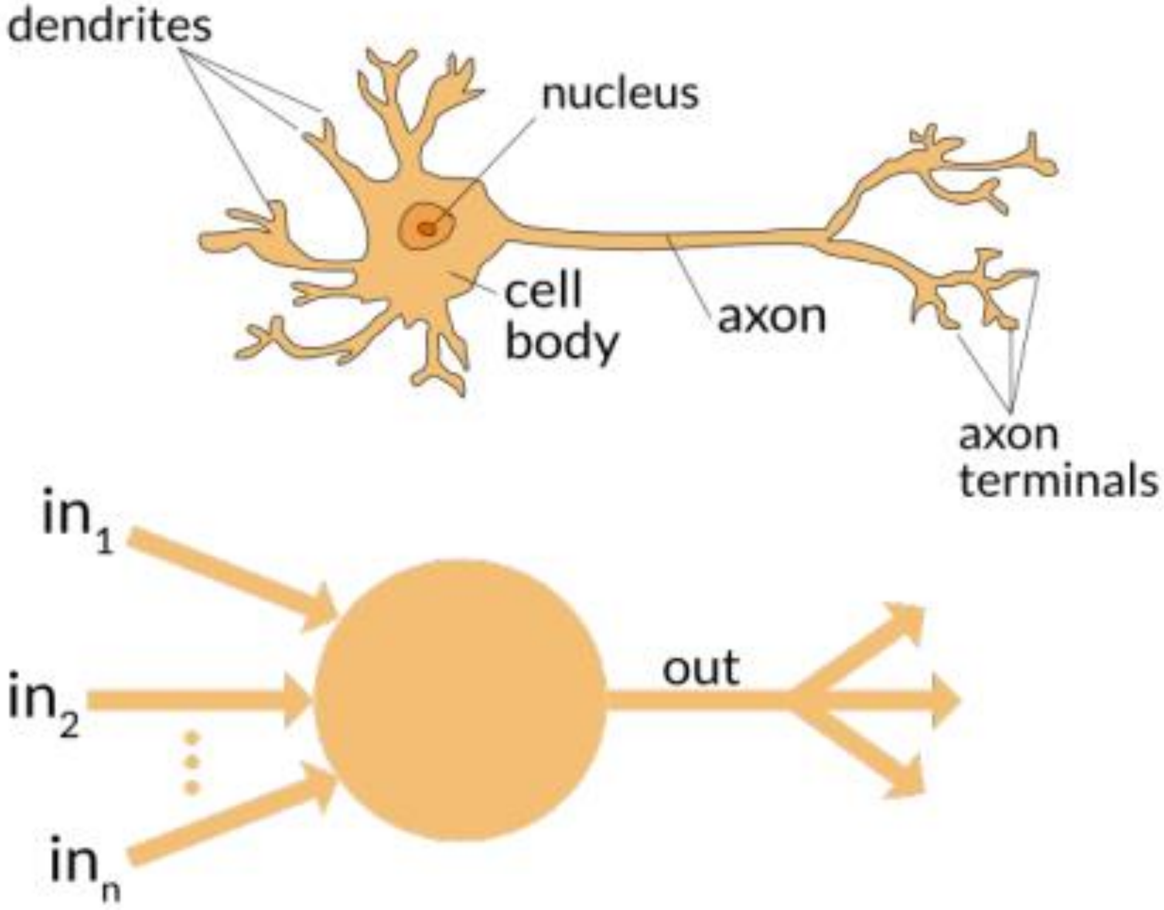
Computer Vision: SURF Features

Kleliboux et al. (2010)

But …

- What are the right features to be paying attention to?
- Could we also learn an appropriate set of operators from the data?
- This takes us back to more complex models

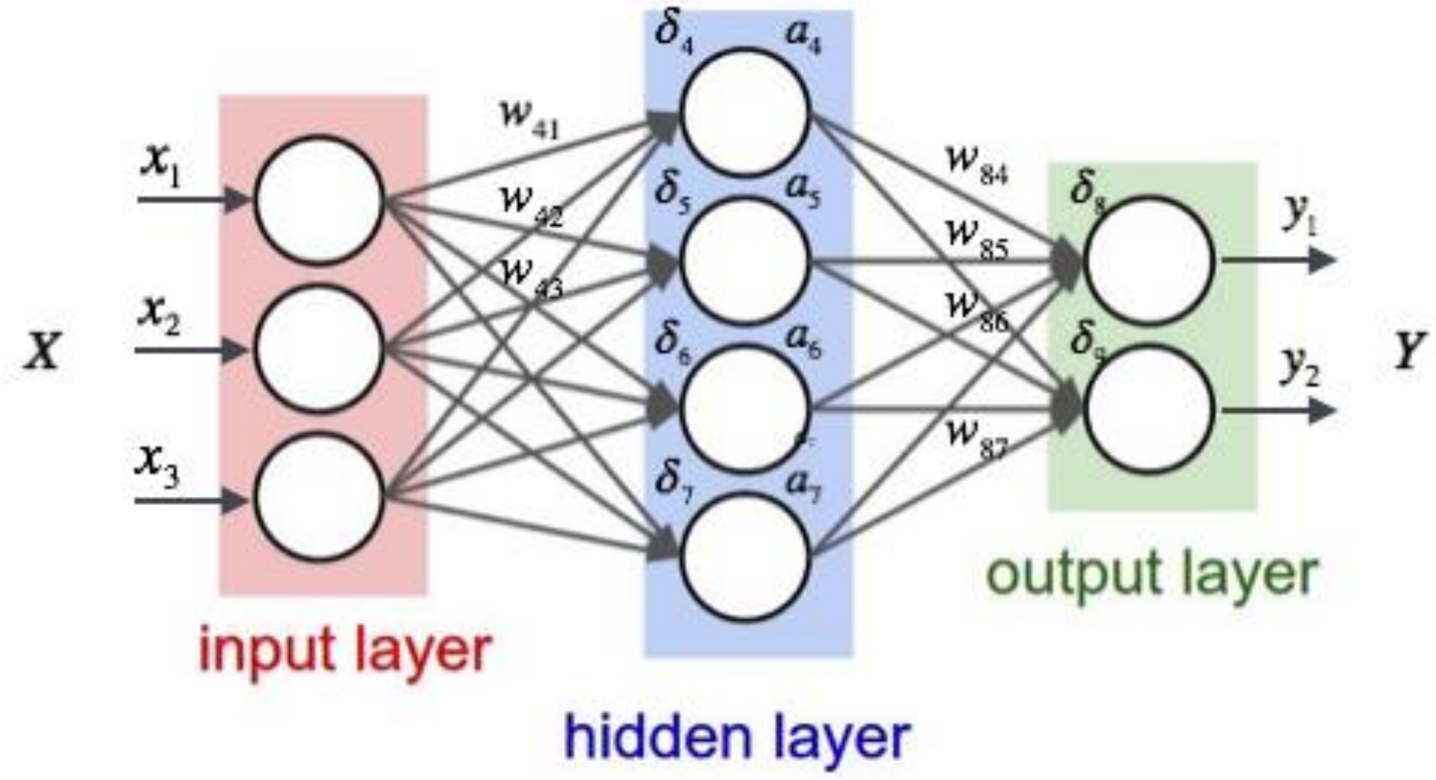
Neurons



Wu (UCLA)

Networks of Neurons

- Parameters are tunable for both the hidden and output layers
- Hidden layer becomes our feature detectors



Valkov (2017)

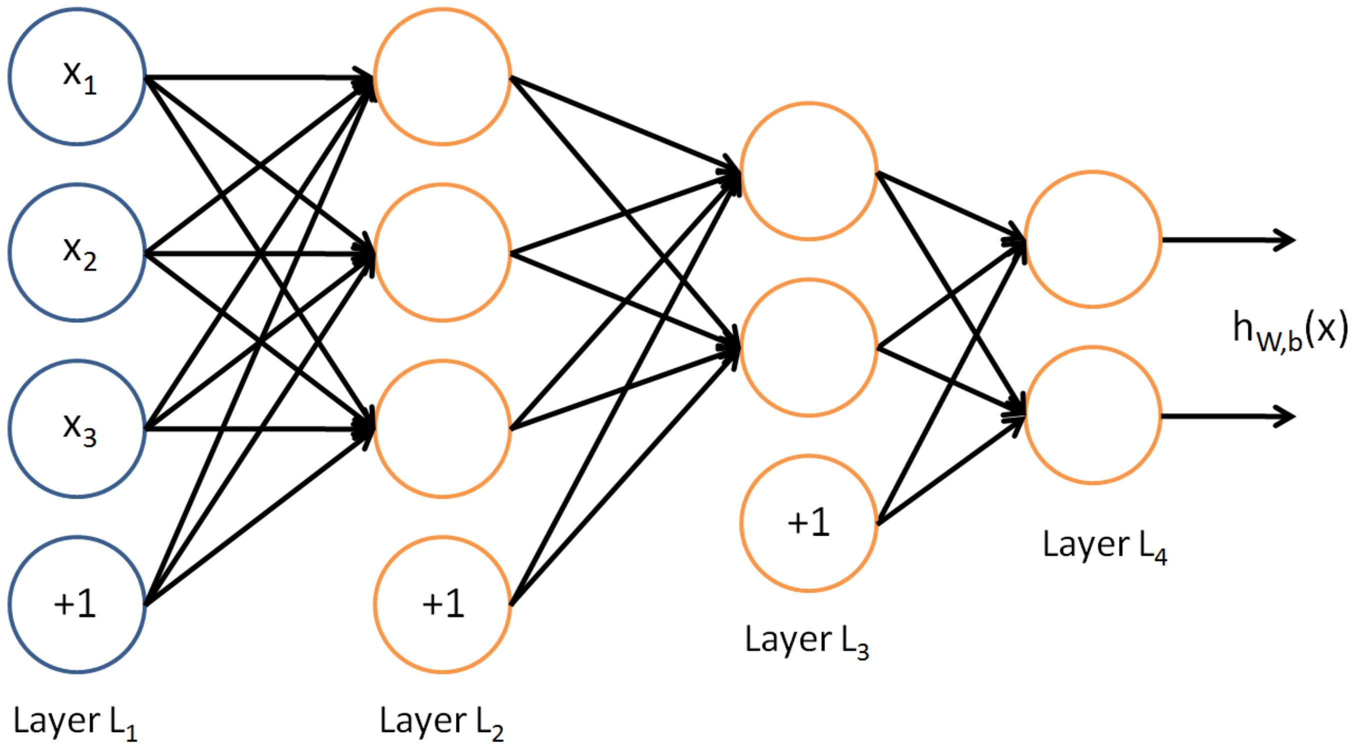
Networks of Neurons

Features are abstractions

- True if these are hand coded or if they are learned!
- Does it make sense to construct abstractions of abstractions?

Deeper Networks

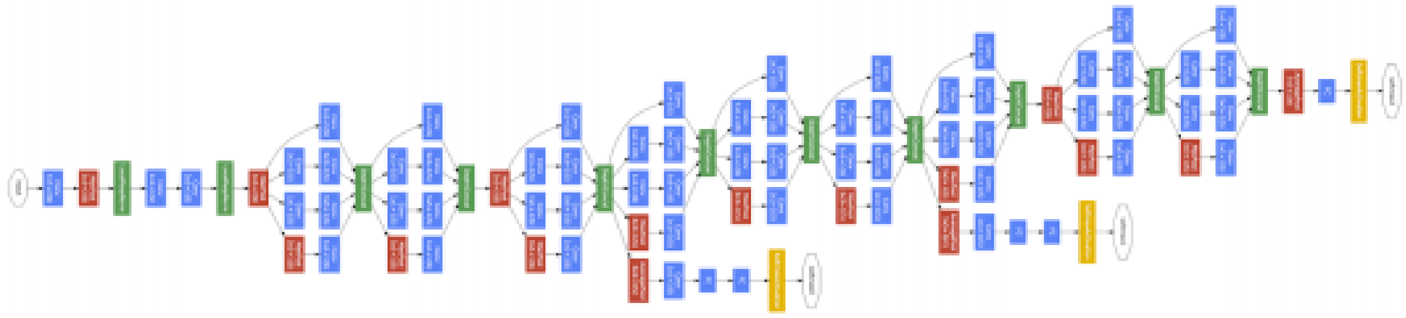
Abstractions of abstractions: multiple hidden layers



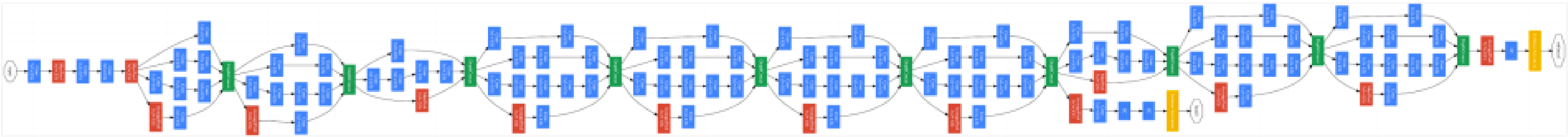
http://deeplearning.stanford.edu/

Even Deeper …

Jou (2019)



[1]Inception 5 (GoogLeNet)



Inception 7a

[1]Going Deeper with Convolutions, [C. Szegedy et al, CVPR 2015]

Agents Acting in the World

Given the current state of the world (or an agent's view of it), what is the next best action to take?

- How do we encode state/situation?
- How do we measure "best"?

Agents Acting in the World

Measuring the outcome of an action:

- Sometimes, we know immediately what the outcome is and can evaluate this outcome
- For many interesting problems, the outcome is dependent on a long history of actions
- This requires an agent to attempt many different sequences of actions to infer what the “best” is

Agents Acting in the World

worldchesspieces.com

analyticsindiamag.com

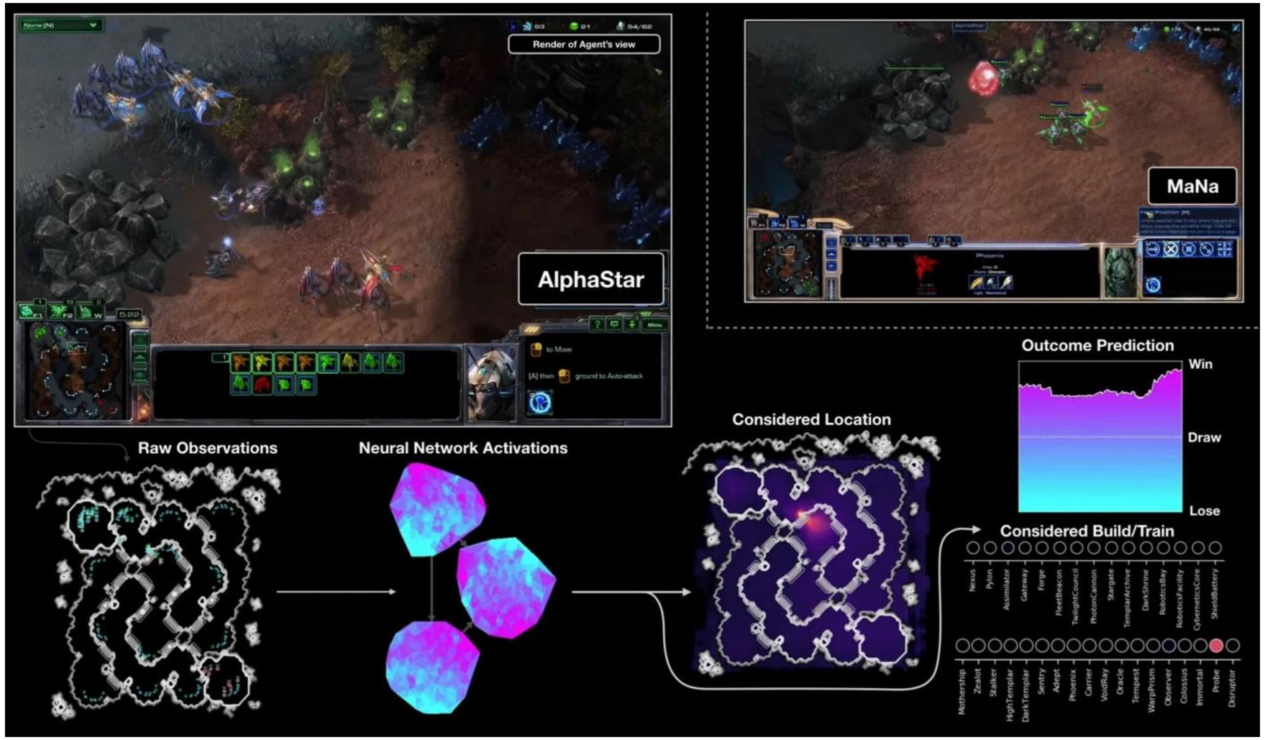
Agents Acting in the World

- Taking the history of states and actions into account in making action decisions is just as complex (perhaps more so) as taking all of the pixels into account in an image classification problem
- Getting a handle on this from a learning perspective requires:
  - Lots of data
  - Plenty of abstraction

Agents Acting in the World

Combining image processing with acting: even more complex problem


ten Pas (2018)



techcrunch.com

(Some) Challenges of Deep Learning

- Having enough data
  - And being able to store it!
- Having the right data
  - Sampling from the true distribution
  - Stationarity of the underlying distribution
- Vanishing/exploding gradient problems
- Being able to explain what a learned model is doing

Recent Advances for DL

Confluence of:

- Availability of **a lot** data
- Computational and data handling hardware
- Easy-to-use computational tools (e.g., python, tensorflow, keras…)
  - Automatic computation of gradients!
- Key algorithmic insights, e.g.:
  - Addressing vanishing gradient issues
  - Parameter sharing
  - Transfer learning

Our Topics

- Backpropagation
- Model Evaluation Process: metrics, cross-validation, statistics, addressing the multiple comparisons problem
- Tools: TensorFlow and Keras
- Convolutional Neural Networks
- Recurrent Neural Networks
- Timeseries Processing
- Deep Reinforcement Learning
- Generative Models
- Generative Adversarial Networks
- Explainability

What I am assuming about you…

- Statistics and hypothesis testing
- Experience with machine learning
  - Including: Multi-layer neural networks and backpropagation
- Programming skills
- Able to jump into Python, including the “Object-Orientedness” of it
- Know or can learn unix command-line tools

Resources

- Course web page: `http://www.cs.ou.edu/~fagg/classes/aml`
- Text: Aurélien Géron (2019) *Hands-On Machine Learning with Scikit-Learn and TensorFlow (Concepts, Tools, and Techniques to Build Intelligent Systems)* ***2nd Edition***, O'Reilly Media
- "Depth" reading (optional): Ian Goodellow, Yoshua Bengio, Aaron Courville (2016) Deep Learning, MIT Press
  - Link to PDF in the syllabus
- Web resources: documentation, tutorials, papers (linked from the schedule or announced on Canvas)

Computing Environment

Setting up a ML environment can be a challenge…

- Tools you need: Python 3.x, Tensorflow, pypl (graphviz), Keras. And - Jupyter is good, too
- Getting your own GPU set up can be a bear

- We will be setting up access to OSCER (OU supercomputer), which will have all of these configured for you
  - Common storage of data and code
  - Support for Jupyter (largely, a testing interface)

Grading

- In-class participation: 10%
- Homework: 45%
- Project work: 45%

Homework

- Keras basics
- Shallow networks
- Deep networks
- Convolutional neural networks
- Deep reinforcement learning x2

Project

- Latter half of the semester
- Three stages:
  - Proposal / advertisement
  - Checkpoint
  - Final presentation

Project

Topic is yours to choose (with constraints)

- Mechanics of doing experiments must be pretty easy to set up
- Testing with multiple runs
- Hyperparameter sensitivity analysis
- Comparison of architectures

Proper Academic Conduct

- Homework assignments are to be done on your own
  - No communication of solutions in any form
  - Do not copy code off the net

- Projects:
  - Groups of 2 working on related problems

For Next Time

- Next time: multi-layer neural networks
- For those needing help in getting into Python, there is a link from Canvas to a set of videos of mine that talk about the key features of Python