Caltech

Machine Learning & Data Mining

CS/CNS/EE 155

Lecture 18:

Survey of Advanced Topics

What We Covered

Topic Overview

Supervised Learning

Linear Models

Overfitting

Loss Functions

Non-Linear Models

Learning Algorithms & Optimization

Probabilistic Modeling

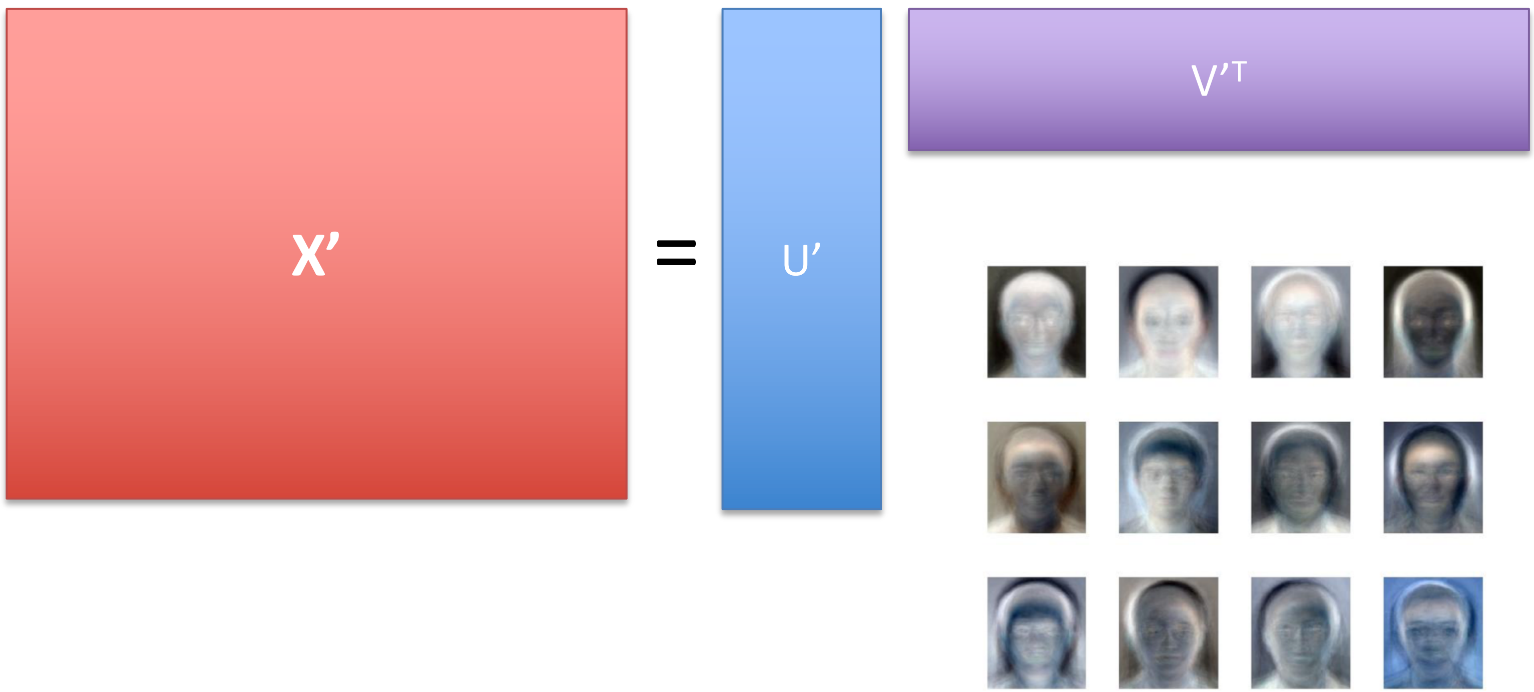
Unsupervised Learning

Basic Supervised Learning

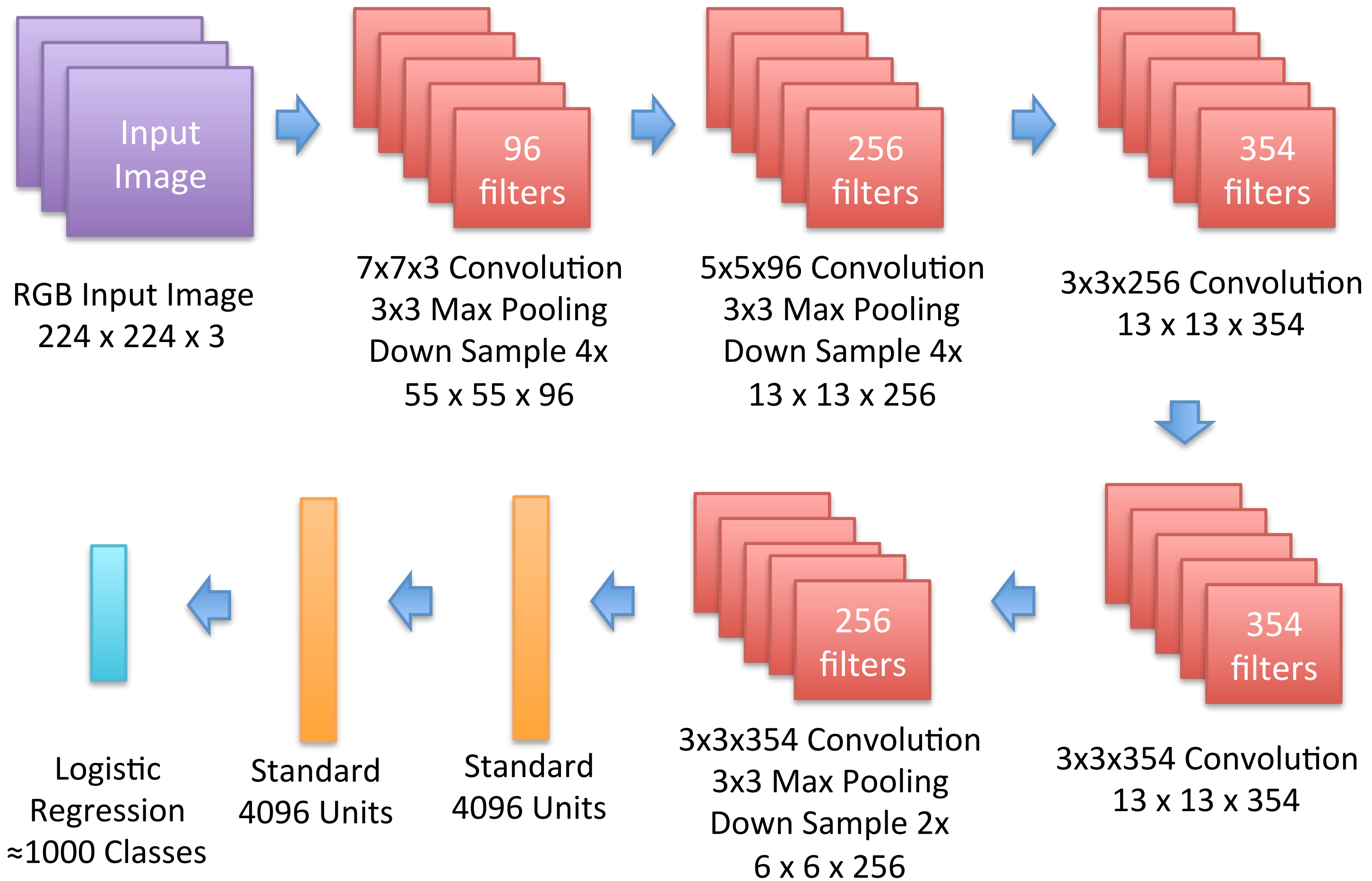
- Training Data: $S = \{(x_i, y_i)\}_{i=1}^N$ $\quad x \in R^D$, $y \in \{-1,+1\}$
- Model Class: $f(x \mid w,b) = w^T x - b$ **Linear Models**
- Loss Function: $L(a,b) = (a-b)^2$ **Squared Loss**
- Learning Objective: $$\operatorname*{argmin}_{w,b} \sum_{i=1}^N L\big(y_i, f(x_i \mid w,b)\big)$$

Optimization Problem

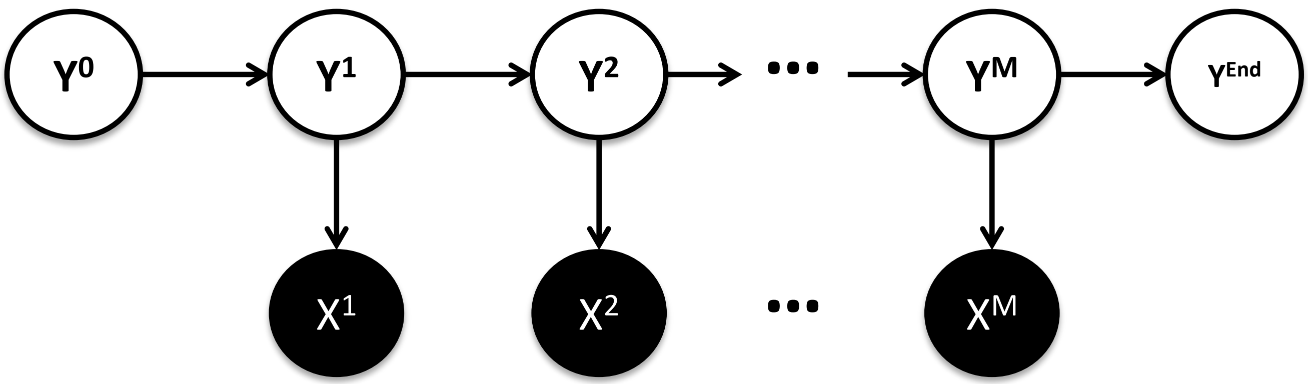
Basic Unsupervised Learning



Deep Learning



Sequence Prediction



Simple Optimization Algorithms

- Stochastic Gradient Descent

- EM algorithm (for HMMs)

Other Basic Concepts

- Cross Validation
- Overfitting
- Bias-Variance Tradeoff

Learning Theory

Generalization Bounds

- Formal characterization of over-fitting
- Example result:

With Prob. ≥ 1-δ :

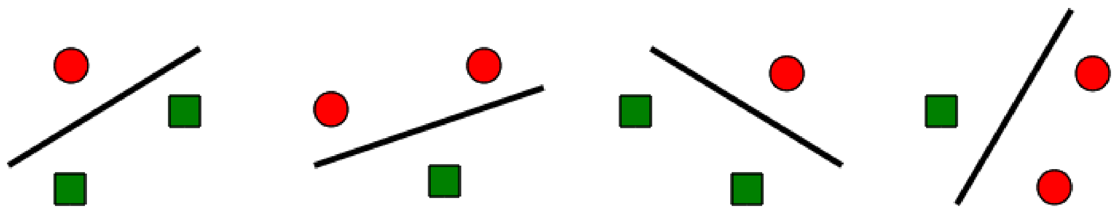
Test Error　　Training Error　　**Make rigorous!**

$$\underbrace{E_{out}(h)}_{\text{Test Error}} \leq \underbrace{E_{in}(h)}_{\text{Training Error}} + \overbrace{O\left(\frac{\log(1/\delta)}{\sqrt{N}}\right)}^{\text{Make rigorous!}}$$

Trained Model　　Training Size

Shattering

- **Definition:** A set of points is **shattered** by H if for all **possible binary labelings** of points, there exists some h that classifies perfectly.

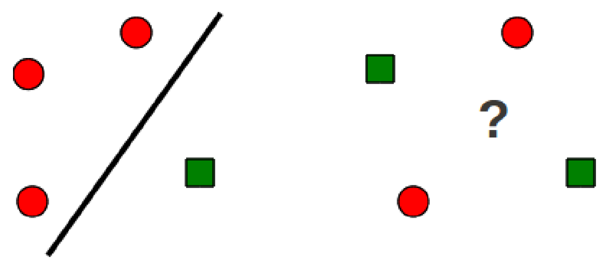


In 2D, any 3 points can always be shattered by linear models!

Slide Material Borrowed From Piyush Rai:
https://www.cs.utah.edu/~piyush/teaching/27-9-print.pdf

Shattering

- **Definition:** A set of points is **shattered** by H if for all **possible binary labelings** of points, there exists some h that classifies perfectly.



In 2D, linear models cannot shatter 4 points!

Slide Material Borrowed From Piyush Rai:
https://www.cs.utah.edu/~piyush/teaching/27-9-print.pdf

VC Dimension

- VC(H) = most # points that can be shattered
  - If H is linear models in 2D feature space:
    - VC(H) = 3

With Prob. ≥ 1-δ :

$$E_{out}(h) \leq E_{in}(h) + O\left(\sqrt{\frac{VC(H)\log\left(\frac{2N}{VC(H)}+1\right)+\log\left(\frac{1}{\delta}\right)}{N}}\right)$$

Structured Prediction

Examples of Complex Output Spaces

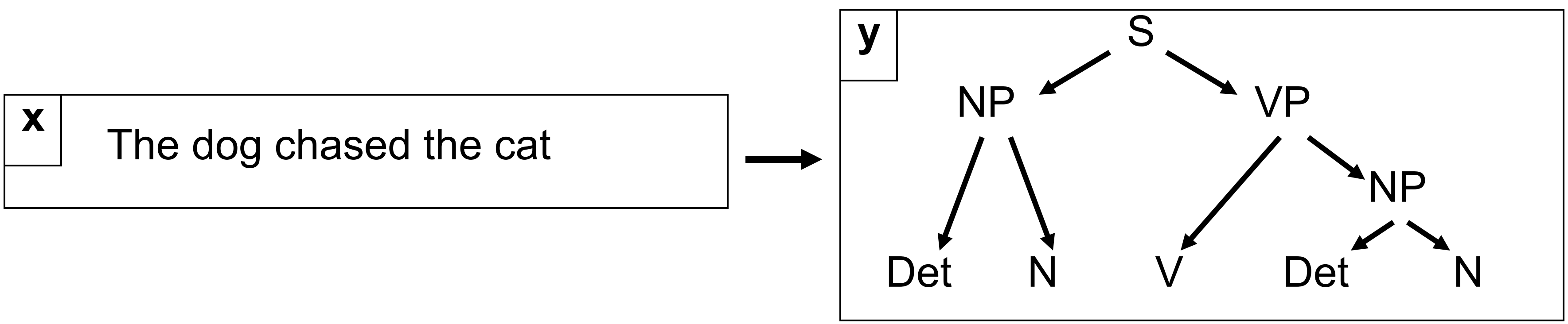
- Part-of-Speech Tagging
  - Given a sequence of words *x*, predict sequence of tags *y*.
  - Dependencies from tag-tag transitions in Markov model.

x The rain wet the cat → **y** Det → N → V → Det → N

→ Similarly for other sequence labeling problems, e.g., RNA Intron/Exon Tagging.

Examples of Complex Output Spaces

- Natural Language Parsing
  - Given a sequence of words *x*, predict the parse tree *y.*
  - Dependencies from structural constraints, since *y* has to be a tree.



Examples of Complex Output Spaces

- Information Retrieval
  - Given a query x, predict a ranking *y*.
  - Dependencies between results (e.g. avoid redundant hits)
  - Loss function over rankings (e.g. Average Precision)

x

SVM

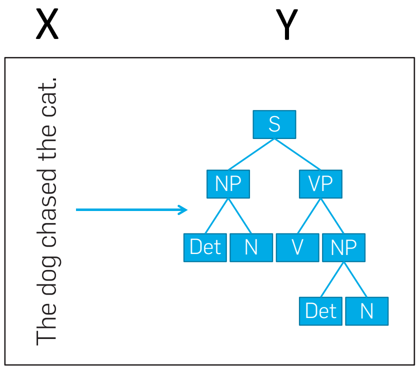
$\longrightarrow$

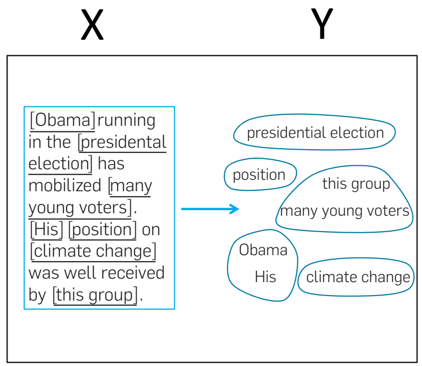
y

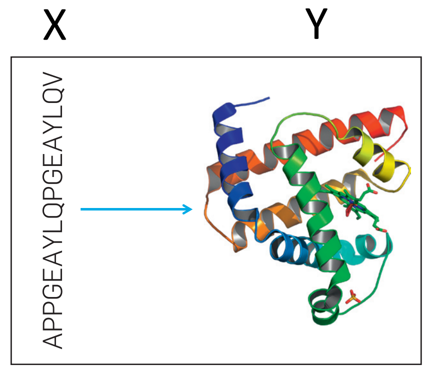
1. **Kernel-Machines**
2. **SVM-Light**
3. **Learning with Kernels**
4. **SV Meppen Fan Club**
5. **Service Master & Co.**
6. **School of Volunteer Management**
7. **SV Mattersburg Online**

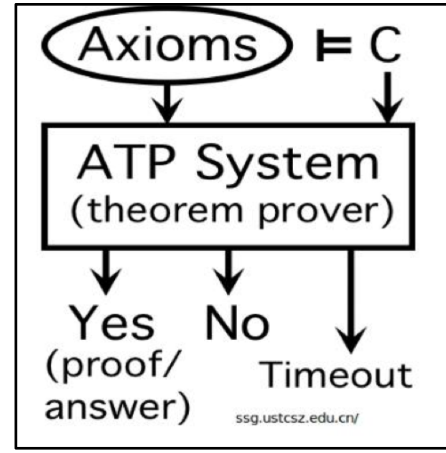
…

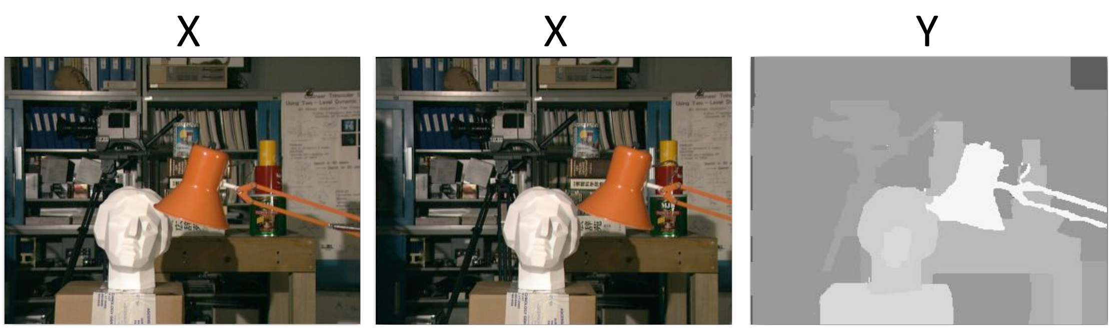
Structured Prediction

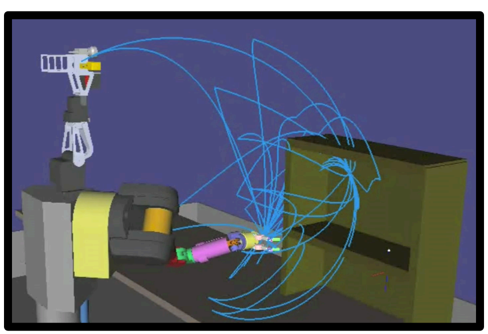


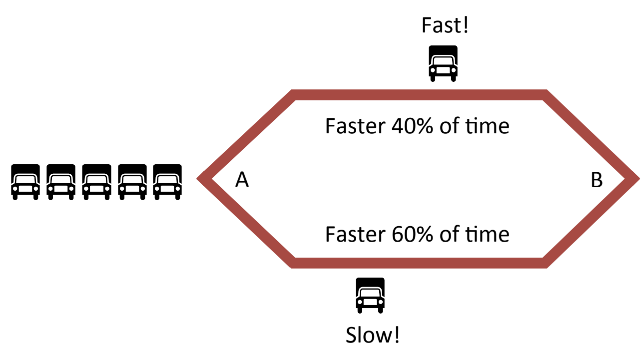


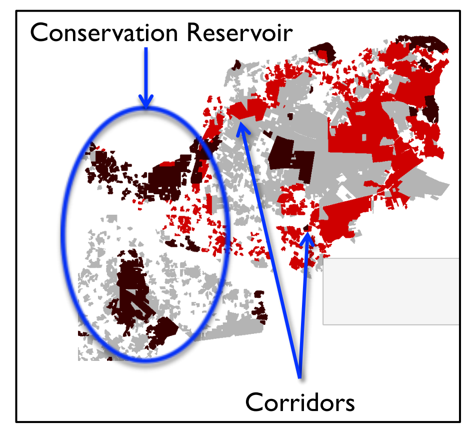












General Formula
(Linear Models)

- Assume scoring function F

$$h(\mathbf{x};w) = \underset{\mathbf{y}\in Y(\mathbf{x})}{\operatorname{argmax}}\, F(\mathbf{x},\mathbf{y};w)$$

- Assume F is linear:

$$F(\mathbf{x},\mathbf{y};w) = w^T\Psi(\mathbf{x},\mathbf{y})$$

Example 1

$$h(\mathbf{x};w) = \underset{\mathbf{y}\in Y(\mathbf{x})}{\operatorname{argmax}} F(\mathbf{x},\mathbf{y};w) \qquad F(\mathbf{x},\mathbf{y};w) = w^T\Psi(\mathbf{x},\mathbf{y})$$

Binary Classification:

$$Y(\mathbf{x}) = \{-1,+1\}$$

$$\Psi(\mathbf{x},y) = y\mathbf{x}$$

$$F(\mathbf{x},y;w) = y(w^T\mathbf{x})$$

$$h(\mathbf{x};w) = \underset{y\in\{-1,+1\}}{\operatorname{argmax}}\, y\left(w^T\mathbf{x}\right)$$

Examples

$$h(\mathbf{x};w) = \operatorname*{argmax}_{\mathbf{y}\in Y(\mathbf{x})} F(\mathbf{x},\mathbf{y};w)$$

$$F(\mathbf{x},\mathbf{y};w) = w^T \Psi(\mathbf{x},\mathbf{y})$$

1st Order Sequences:

$Y(\mathbf{x}) =$ all possible output sequences

$$\Psi(\mathbf{x},\mathbf{y}) = \sum_j \phi(y^j, y^{j-1} \mid \mathbf{x})$$

$$F(\mathbf{x},\mathbf{y};w) = w^T \sum_j \phi(y^j, y^{j-1} \mid \mathbf{x})$$

Solve using Viterbi!

Examples

$$h(\mathbf{x};w) = \operatorname*{argmax}_{\mathbf{y}\in Y(\mathbf{x})} F(\mathbf{x},\mathbf{y};w)$$

$$F(\mathbf{x},\mathbf{y};w) = w^T \Psi(\mathbf{x},\mathbf{y})$$

Integer Linear Program:

$Y(\mathbf{x}) =$ Feasible settings of **y**

Each $y^j \in \{0,1\}$

$$\Psi(\mathbf{x},\mathbf{y}) = \sum_j y^j \phi^j(\mathbf{x})$$

$$F(\mathbf{x},\mathbf{y};w) = \mathbf{y}^T \mathbf{c} \qquad \mathbf{c} = \begin{bmatrix} w^T\phi^1(\mathbf{x}) \\ w^T\phi^2(\mathbf{x}) \\ \vdots \end{bmatrix}$$

$$h(\mathbf{x};w) = \operatorname*{argmax}_{\mathbf{y}\in Y(\mathbf{x})} \mathbf{y}^T \mathbf{c}$$

Structured Prediction Learning Problem

- **Efficient Inference/Prediction**

$$\boxed{h(\mathbf{x};w) = \underset{\mathbf{y}}{\operatorname{argmax}}\, w^T \Psi(\mathbf{y},\mathbf{x})}$$

  - Viterbi in sequence labeling
  - CKY Parser for parse trees
  - Sorting for ranking

- **Efficient Learning/Training**
  - Learn parameters *w* from training data $\{\mathbf{x}_i,\mathbf{y}_i\}_{i=1..N}$
  - Structural SVM: Hinge Loss Minimization
  - Conditional Random Fields: Log Loss Minimization
  - Structured Perceptron, etc…

Perceptron Learning Algorithm

- $w^1 = 0$, $b^1 = 0$
- For t = 1 ….
  - Receive example (x,y)
  - If $h(x|w^t) = y$
    - $[w^{t+1}, b^{t+1}] = [w^t, b^t]$
  - Else
    - $w^{t+1} = w^t + yx$
    - $b^{t+1} = b^t + y$

$$h(x\,|\,w) = sign(w^T x - b)$$

Training Set:

$$S = \{(x_i, y_i)\}_{i=1}^{N}$$

$$y \in \{+1, -1\}$$

Go through training set in arbitrary order (e.g., randomly)

Structured Perceptron

- $w^1 = 0$
- For t = 1 ….
  - Receive example (x,y)
  - If $h(x|w^t) = y$
    - $w^{t+1} = w^t$
  - Else
    - $w^{t+1} = w^t + \Psi(x,y)$

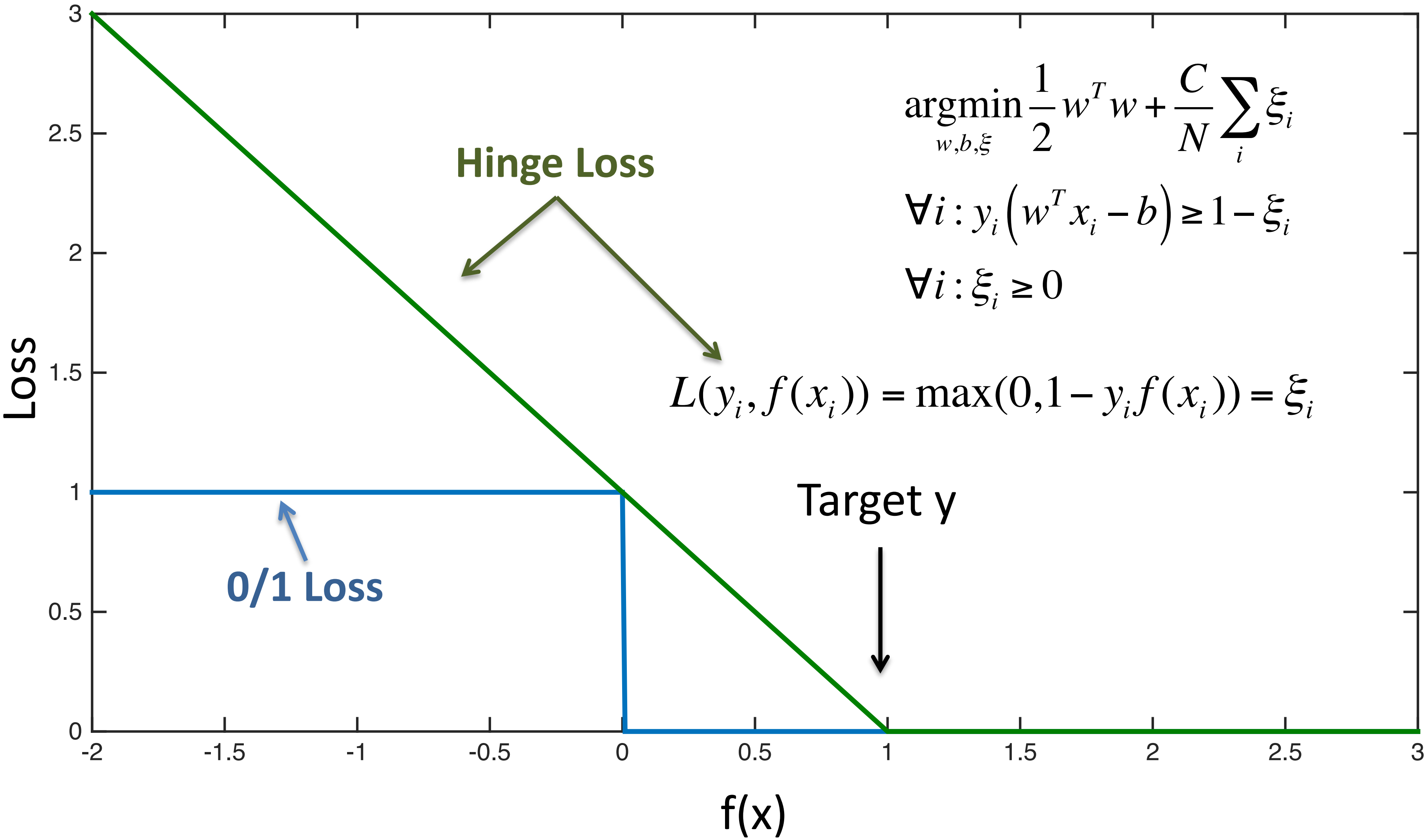
$$h(x \mid w) = \underset{y'}{\operatorname{argmax}}\, w^T \Psi(x, y')$$

Training Set:

$$S = \left\{(x_i, y_i)\right\}_{i=1}^{N}$$

Go through training set in arbitrary order (e.g., randomly)

Conventional SVMs



Structural SVM

- Let **x** denote a structured input (sentence)
- Let **y** denote a structured output (POS tags)
- Standard objective function: $\frac{1}{2}w^2 + \frac{C}{N}\sum_i \xi_i$
- Constraints are defined for each incorrect labeling **y'** over each **x**.

$$\forall i, \forall \mathbf{y}' \neq \mathbf{y}^{(i)} : \quad \underbrace{w^T \Psi(\mathbf{y}^{(i)}, \mathbf{x}^{(i)})}_{\text{Score}(\mathbf{y}^{(i)})} \geq \underbrace{w^T \Psi(\mathbf{y}', \mathbf{x}^{(i)})}_{\text{Score}(\mathbf{y}')} + \underbrace{\Delta_i(\mathbf{y}')}_{\text{Loss}(\mathbf{y}')} - \underbrace{\xi_i}_{\text{Slack}}$$

http://www.cs.cornell.edu/People/tj/svm_light/svm_struct.html

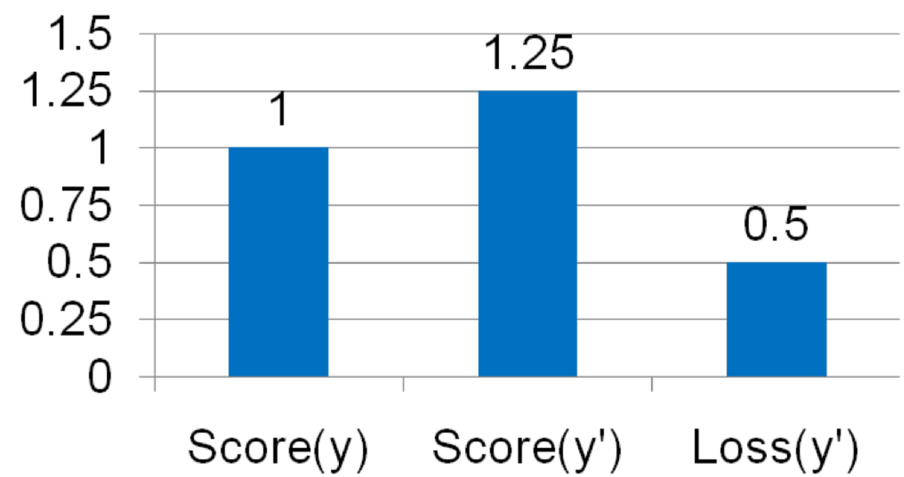
Interpreting Constraints

$$\frac{1}{2}w^2 + \frac{C}{N}\sum_i \xi_i$$

$$\forall i, \forall \mathbf{y}' \neq \mathbf{y}^{(i)} : \quad \underbrace{w^T\Psi(\mathbf{y}^{(i)}, \mathbf{x}^{(i)})}_{\textbf{Score}(\mathbf{y}^{(\mathbf{i})})} \geq \underbrace{w^T\Psi(\mathbf{y}', \mathbf{x}^{(i)})}_{\textbf{Score}(\mathbf{y}')} + \underbrace{\Delta_i(\mathbf{y}')}_{\textbf{Loss}(\mathbf{y}')} - \underbrace{\xi_i}_{\textbf{Slack}}$$

Suppose for incorrect **y'** :

Then: $\boxed{\xi_i \geq 0.75 \geq \Delta(\mathbf{y}')}$



http://www.cs.cornell.edu/People/tj/svm_light/svm_struct.html

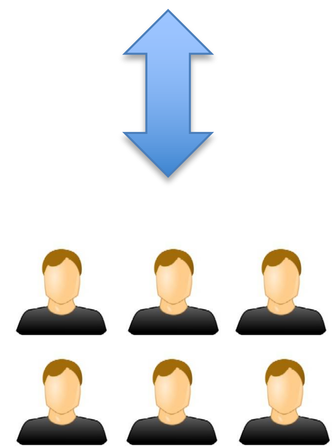
Sample Research Questions

- Scale
  - Predicting over millions of variables
- Structured Representation Learning
  - Deep learning for structured outputs?
- Cost of labeling

Crowdsourcing

Acquiring Labels from Annotators

amazon
mechanical turk
beta



How Reliable are Annotators?

- If we knew what the labels were
  - Can judge workers on label quality
- If we knew who the good workers were
  - Can create labels from their annotations
- **Chicken and egg problem!**

Worker Reliability as Latent Variable

- Let z_m denote the reliability of worker m

Estimated label

$$y_i = \frac{1}{\sum_m z_m} \sum_m y_{im} z_m$$

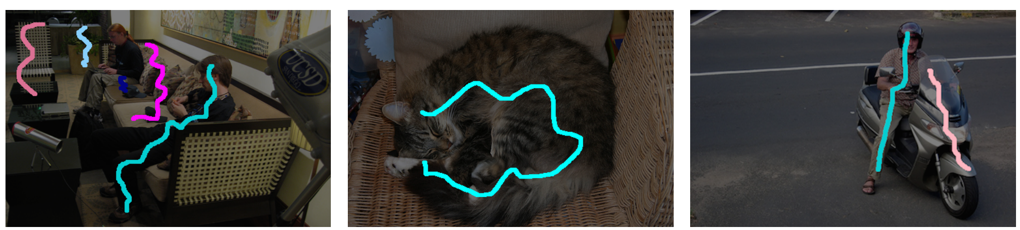
$$z_m = \frac{1}{N} \sum_i L(y_i, y_{im})$$

Differing Ambiguities Across Tasks

- Often collecting annotations for many tasks
- Some tasks are harder than others
- How many labels to collect for each task?

Structured Annotations

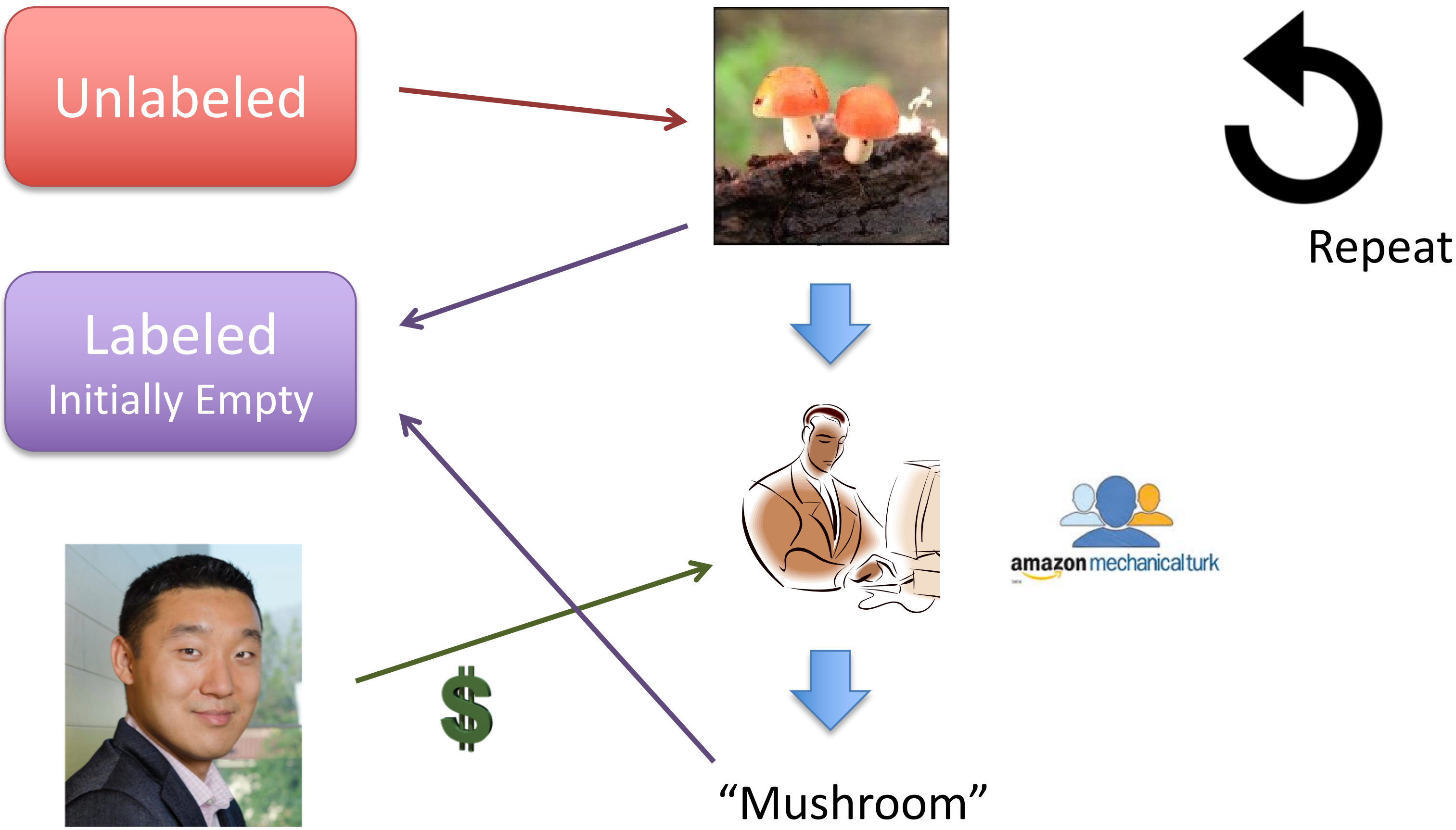
Full supervision | Image-level supervision | Point-level supervision



http://arxiv.org/pdf/1506.02106v4.pdf

Active Learning

Crowdsourcing



Passive Learning

Unlabeled

Random

Repeat

Labeled

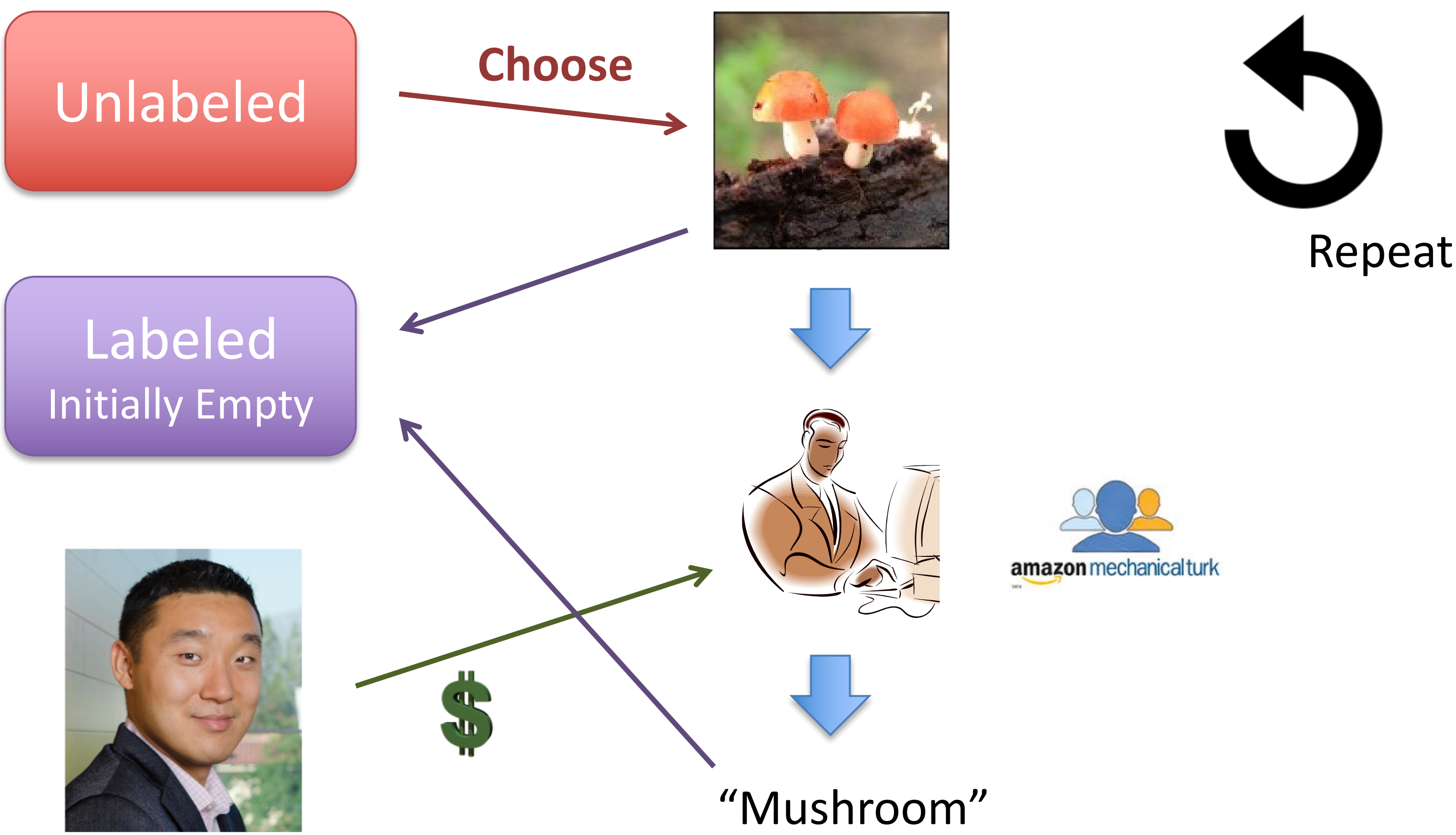
Initially Empty

amazon mechanicalturk

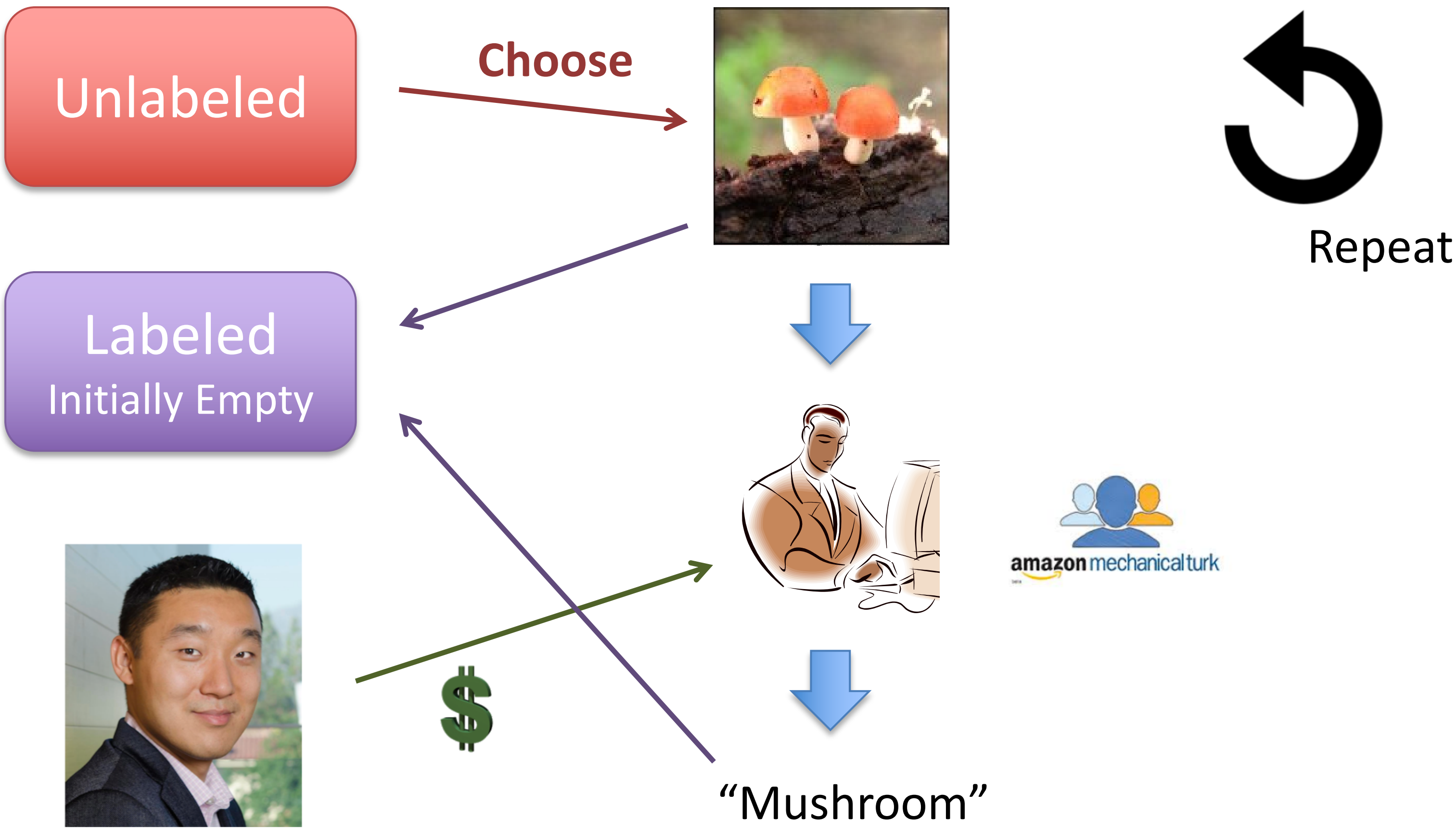
$

“Mushroom”

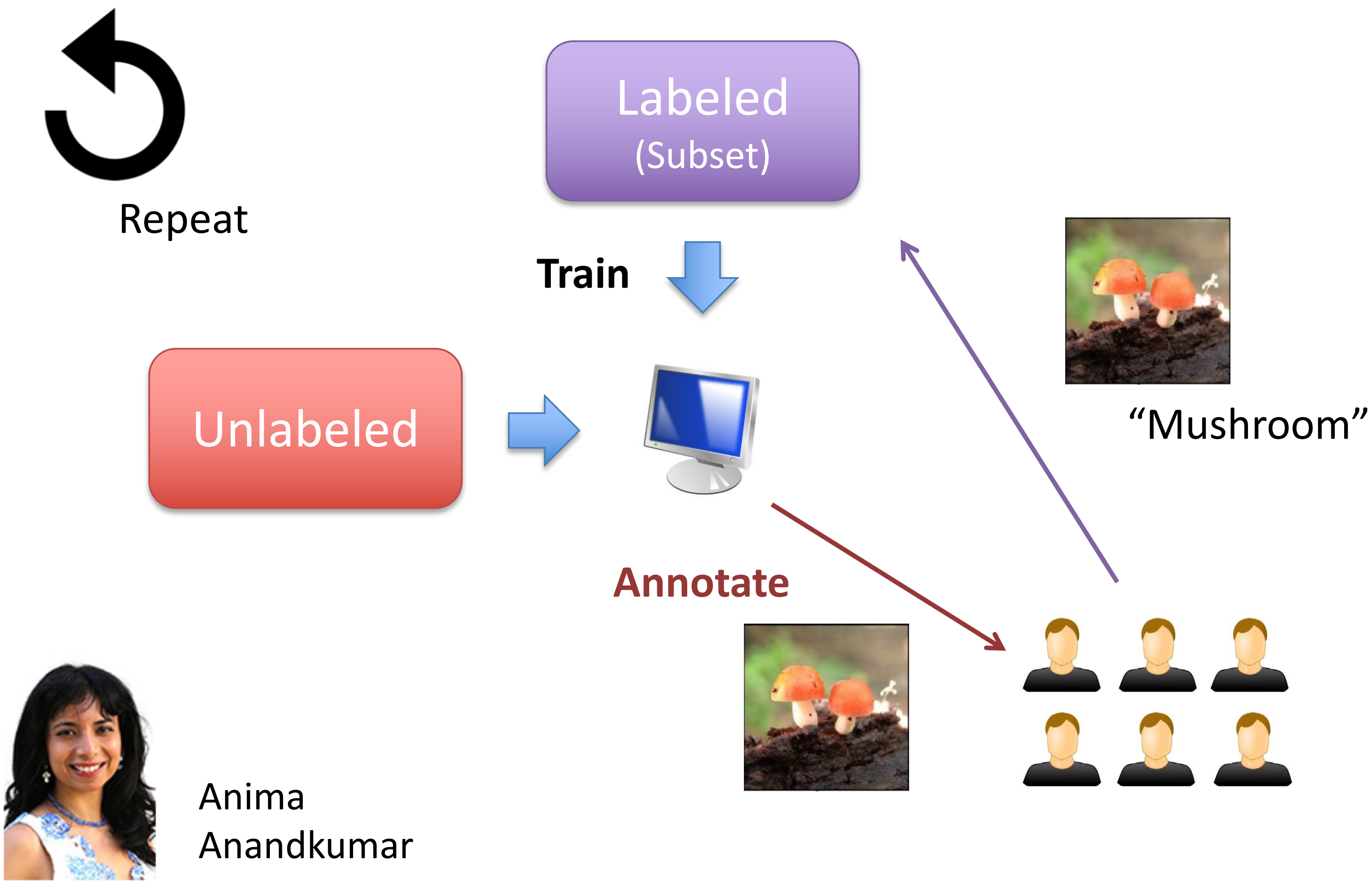
Active Learning



Goal: Maximize Accuracy with Minimal Cost



On-Demand Crowdsourcing

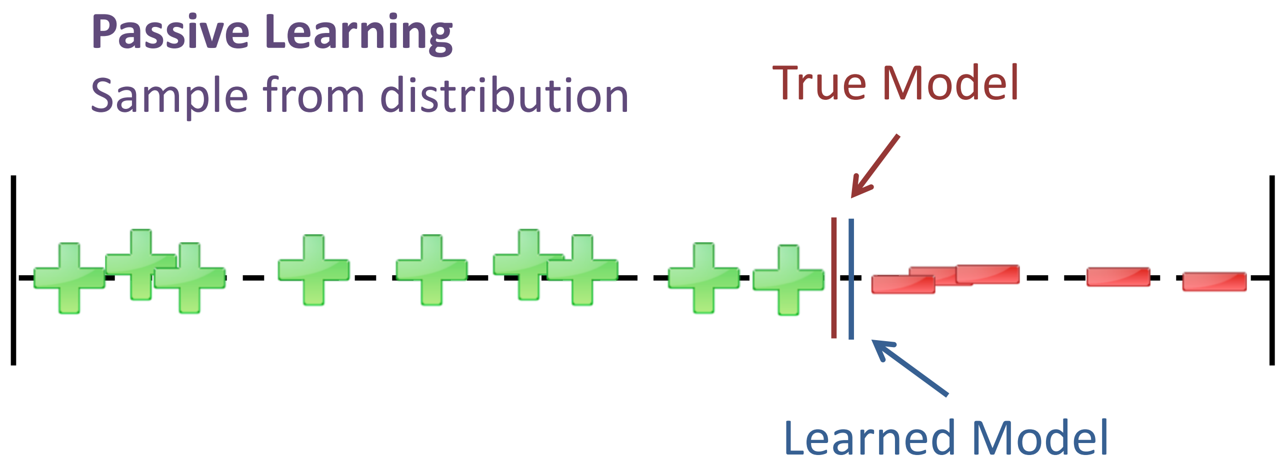


Comparison with Passive Learning

- Conventional Supervised Learning is considered "Passive" Learning
- Unlabeled training set sampled according to test distribution
- So we label it at random
  - **Very Expensive!**

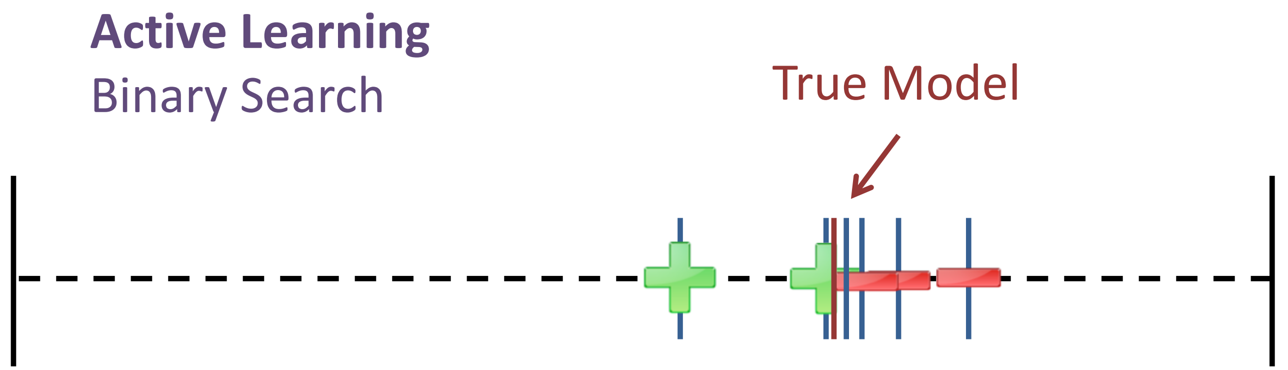
Simple Example

- 1 feature
- Learn threshold function



Simple Example

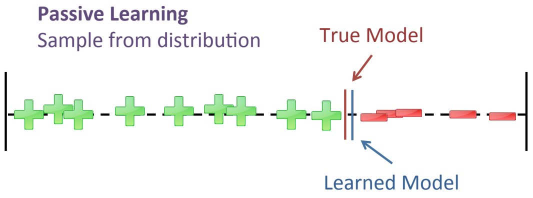
- 1 feature
- Learn threshold function



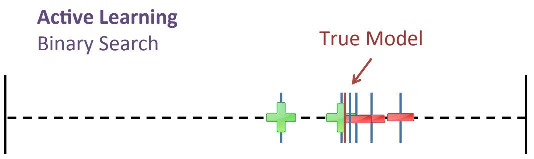
Comparison with Passive Learning

- # samples to be within ε of true model

- Passive Learning: $O\left(\frac{1}{\varepsilon}\right)$



- Active Learning: $O\left(\log\frac{1}{\varepsilon}\right)$

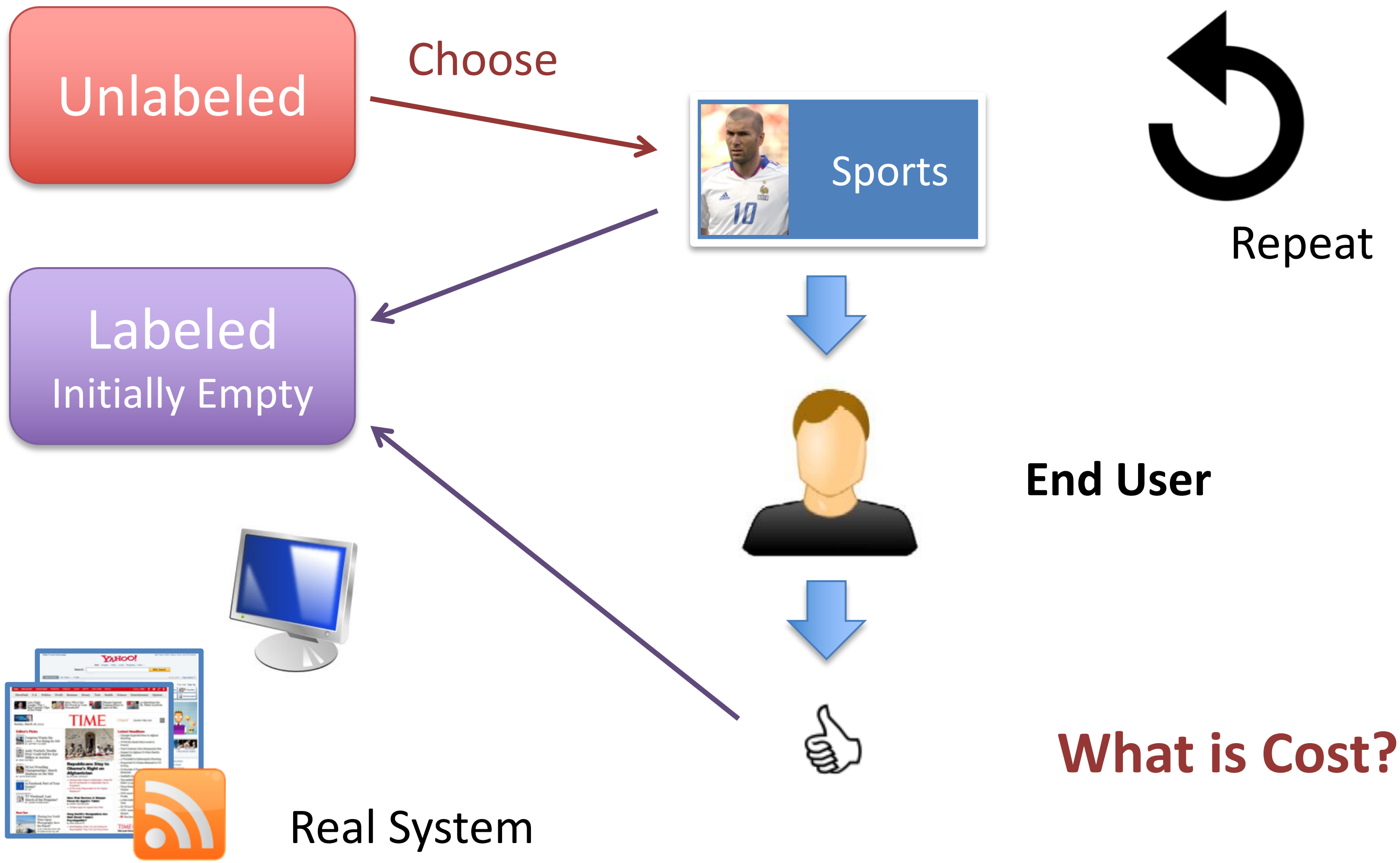


Multi-Armed Bandits

Problems with Crowdsourcing

- Assumes you can label by proxy
  - E.g., have someone else label objects in images
- But sometimes you can't!
  - Personalized recommender systems
    - Need to ask the user whether content is interesting
  - Personalized medicine
    - Need to try treatment on patient
  - **Requires actual target domain**

Personalized Labels



Formal Definition

- K actions/classes
- Each action has an average reward: μ_k
  - Unknown to us
  - Assume WLOG that u_1 is largest

Basic Setting
K classes
No features

- For t = 1…T
  - Algorithm chooses action a(t)
  - Receives random reward y(t)
    - Expectation $\mu_{a(t)}$

Algorithm Simultaneously
Predicts & Receives Labels

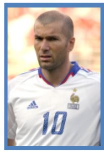
- **Goal:** minimize $Tu_1 - (\mu_{a(1)} + \mu_{a(2)} + \ldots + \mu_{a(T)})$

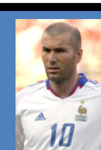
If we had perfect information to start

Expected Reward of Algorithm

Interactive Personalization

(5 Classes, No features)

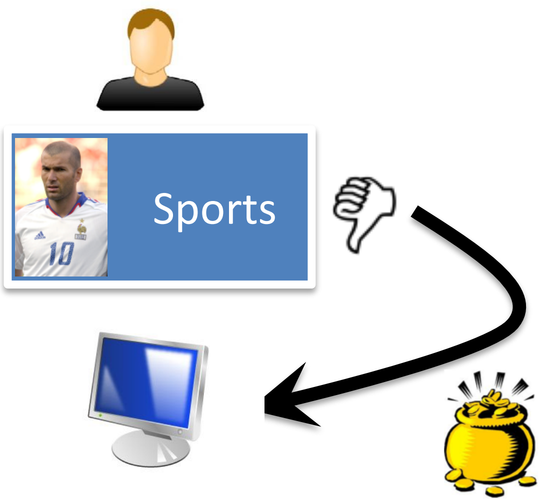
 Sports

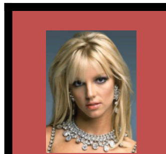
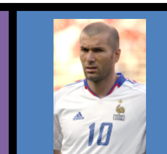
					
Average Likes	--	--	--	--	--
# Shown	0	0	0	1	0

 : 0

Interactive Personalization

(5 Classes, No features)

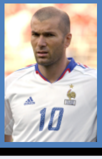


					
Average Likes	--	--	--	0	--
# Shown	0	0	0	1	0

 : 0

Interactive Personalization

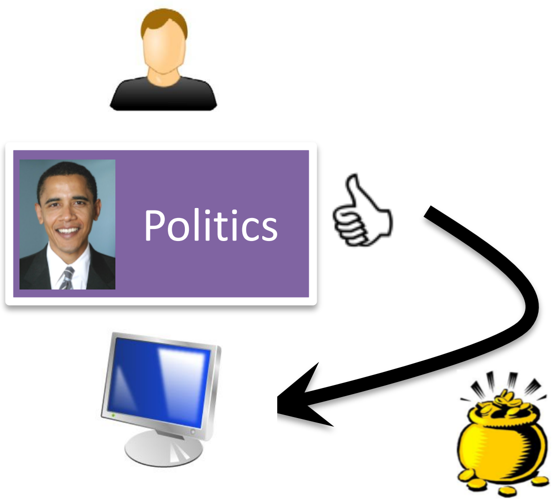
(5 Classes, No features)

					
Average Likes	--	--	--	0	--
# Shown	0	0	1	1	0

 : 0

Interactive Personalization

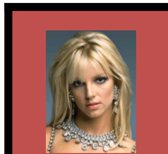
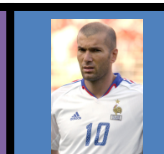
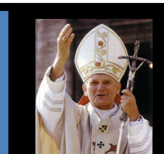
(5 Classes, No features)



Average Likes	--	--	1	0	--
# Shown	0	0	1	1	0

Interactive Personalization

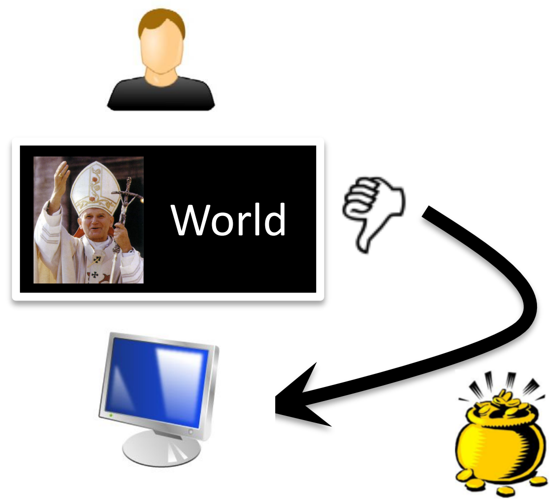
(5 Classes, No features)

					
Average Likes	--	--	1	0	--
# Shown	0	0	1	1	1

 : 1

Interactive Personalization

(5 Classes, No features)



Average Likes	--	--	1	0	0
# Shown	0	0	1	1	1

 : 1

Interactive Personalization

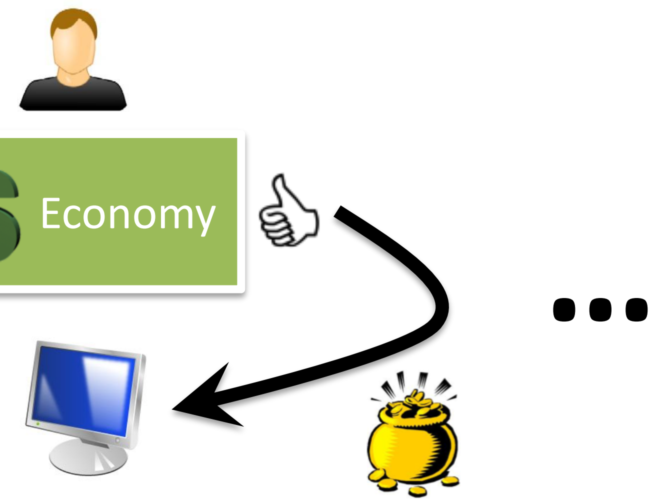
(5 Classes, No features)

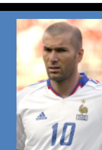
					
Average Likes	--	--	1	0	0
# Shown	0	1	1	1	1

 : 1

Interactive Personalization

(5 Classes, No features)



					
Average Likes	--	1	1	0	0
# Shown	0	1	1	1	1

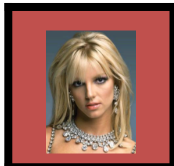
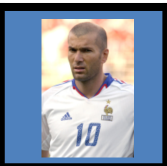
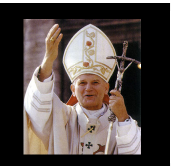
 : 2

What should Algorithm Recommend?

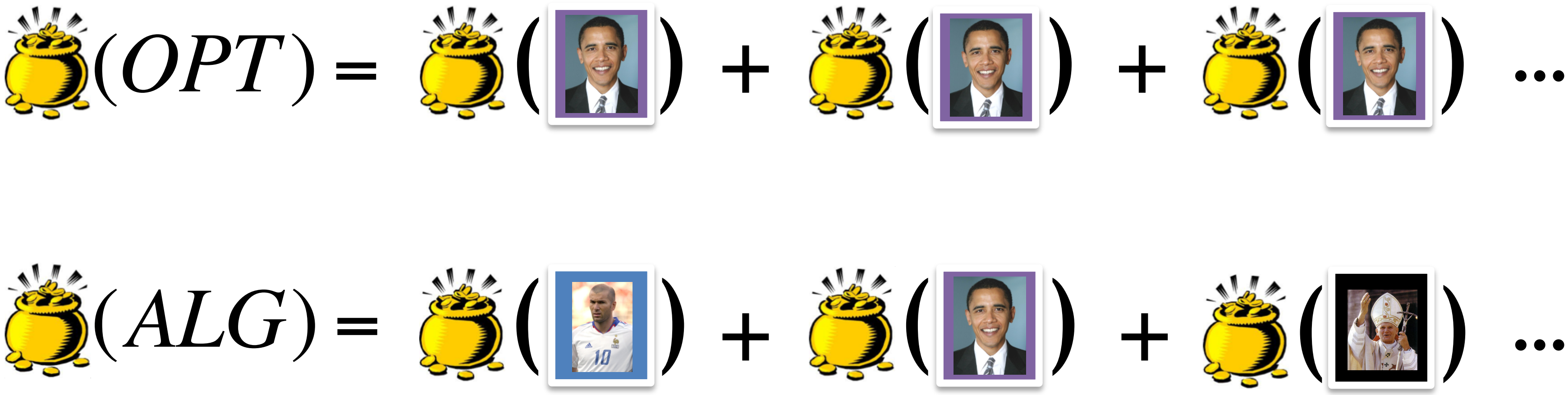
Exploit:	**Explore:**	**Best:**
		

How to Optimally Balance Explore/Exploit Tradeoff?

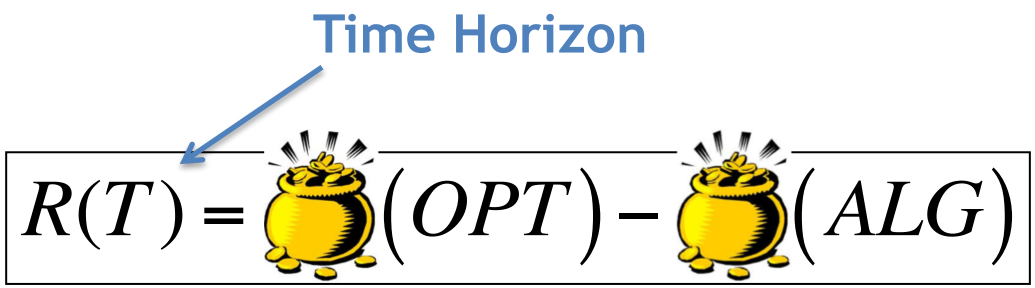
Characterized by the Multi-Armed Bandit Problem

					
Average Likes	--	0.44	0.4	0.33	0.2
# Shown	0	25	10	15	20

 : 24



Regret:



- Opportunity cost of not knowing preferences
- "**no-regret**" if R(T)/T ➔ 0
  - Efficiency measured by convergence rate

Recap: The Multi-Armed Bandit Problem

- K actions/classes
- Each action has an average reward: μ_k
  - All unknown to us
  - Assume WLOG that u_1 is largest

Basic Setting
K classes
No features

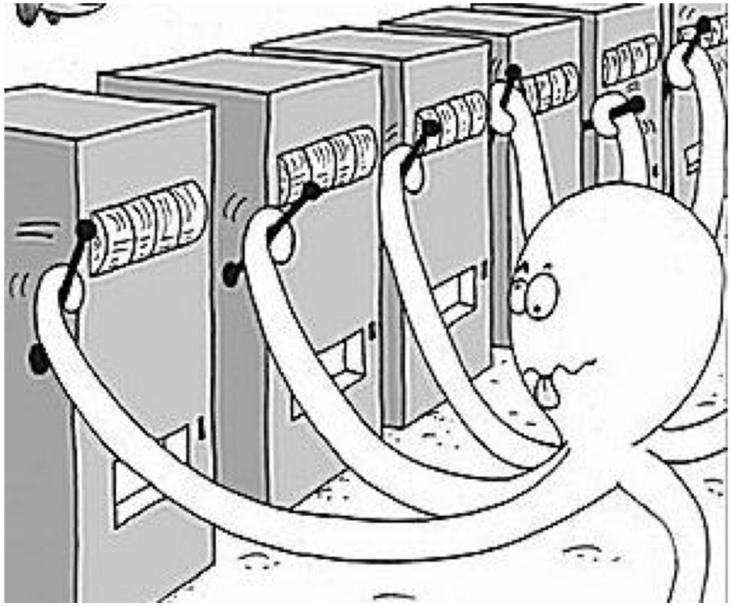
- For t = 1…T
  - Algorithm chooses action a(t)
  - Receives random reward y(t)
    - Expectation $\mu_{a(t)}$

Algorithm Simultaneously
Predicts & Receives Labels

- Goal: minimize $T u_1 - \underbrace{(\mu_{a(1)} + \mu_{a(2)} + \ldots + \mu_{a(T)})}_{\text{Regret}}$

The Motivating Problem

- Slot Machine = One-Armed Bandit



Each Arm Has
Different Payoff

- **Goal:** Minimize regret From pulling suboptimal arms

http://en.wikipedia.org/wiki/Multi-armed_bandit

Implications of Regret

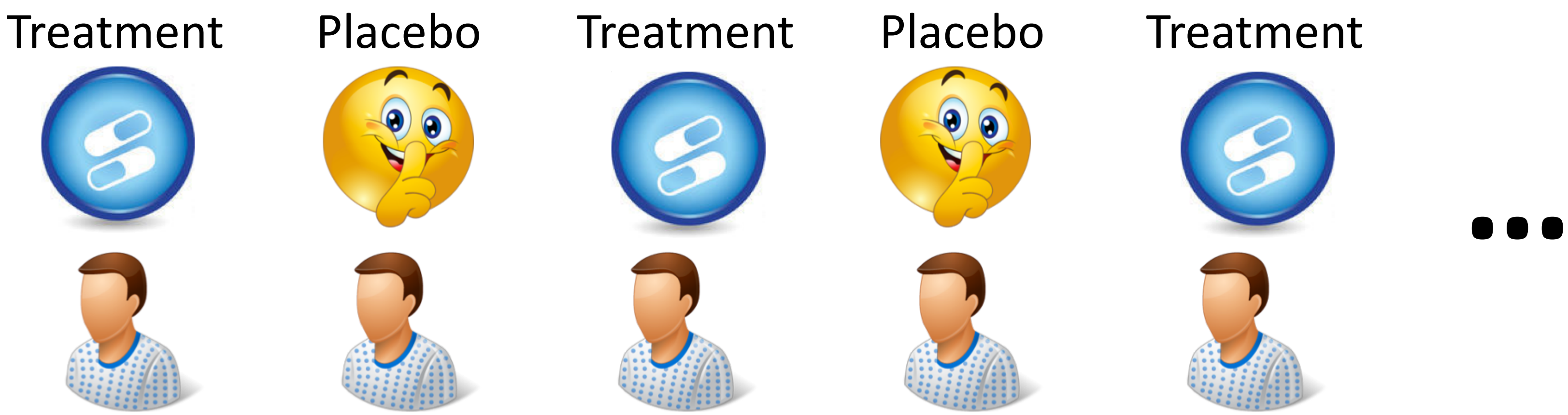
Regret: $R(T) =$  $(OPT) -$  (ALG)

- If R(T) grows linearly w.r.t. T:
  - Then R(T)/T ➔ constant > 0
  - I.e., we converge to predicting something suboptimal

- If R(T) is sub-linear w.r.t. T:
  - Then R(T)/T ➔ 0
  - I.e., we converge to predicting the optimal action

Experimental Design

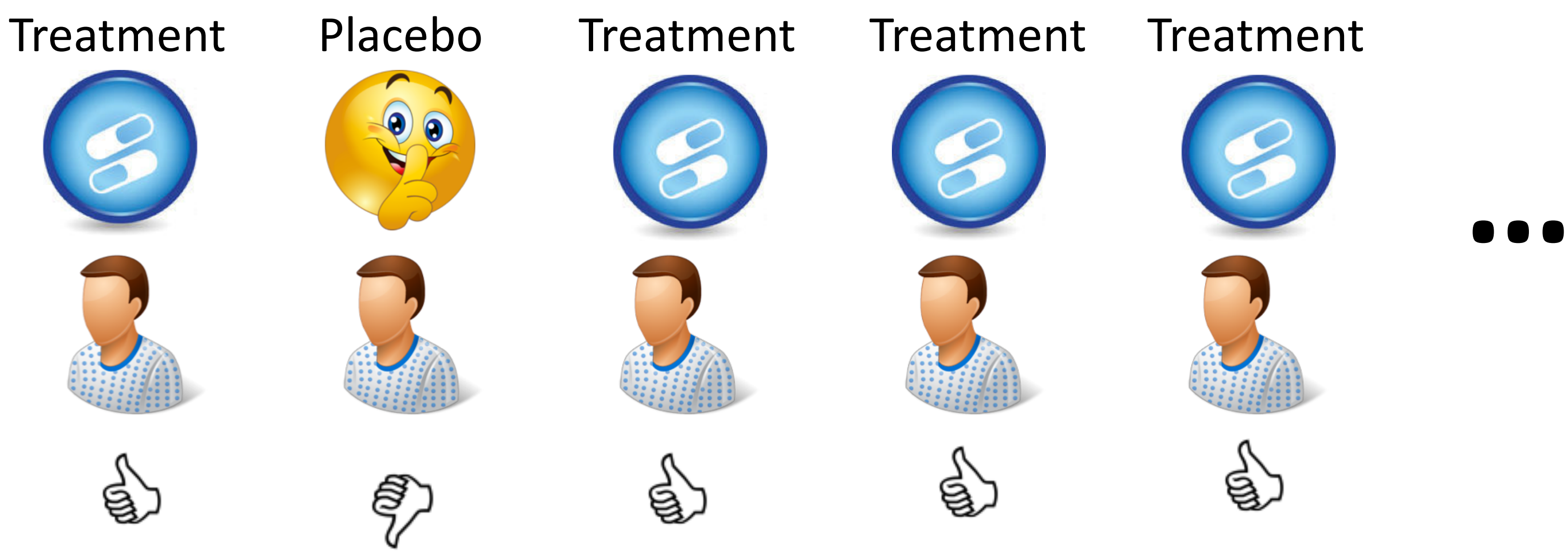
- How to split trials to collect information
- **Static Experimental Design**
  - Standard practice
  - (pre-planned)



http://en.wikipedia.org/wiki/Design_of_experiments

Sequential Experimental Design

- Adapt experiments based on outcomes



Sequential Experimental Design Matters

Monica Almeida/The New York Times, left

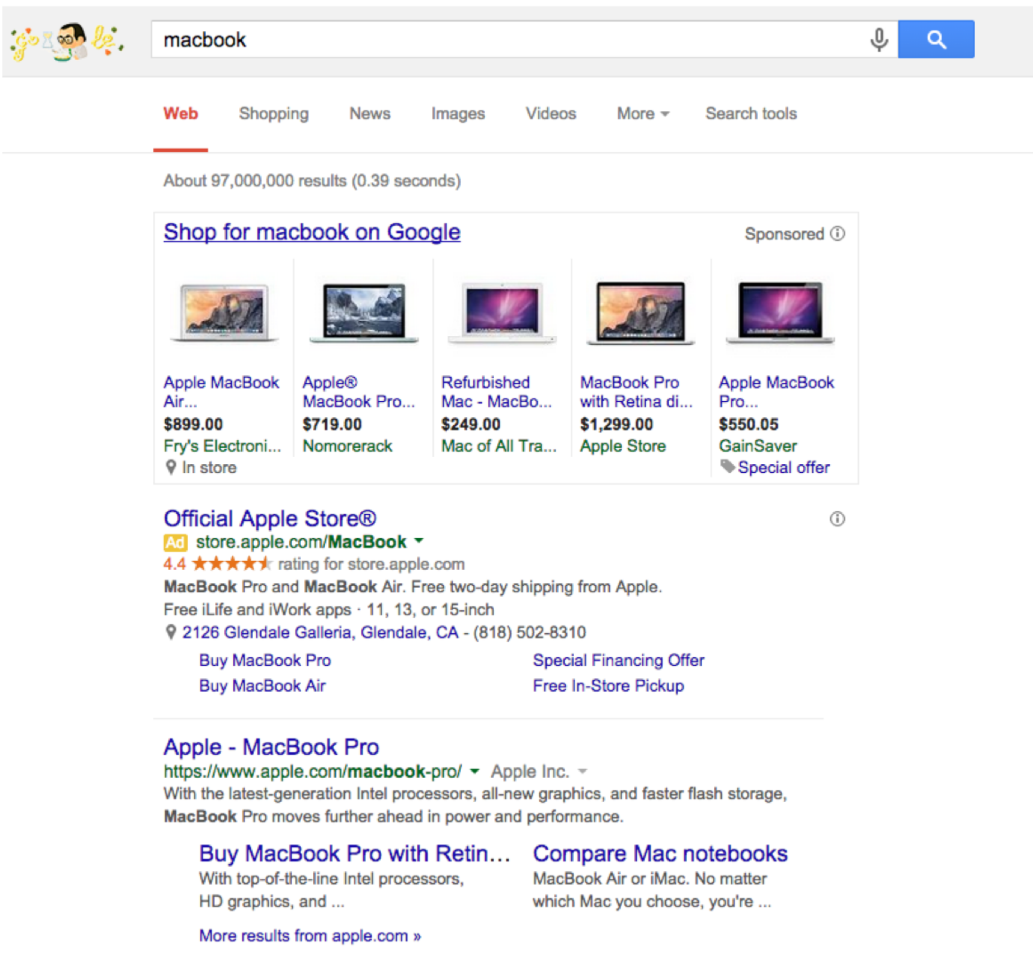
Two Cousins, Two Paths Thomas McLaughlin, left, was given a promising experimental drug to treat his lethal skin cancer in a medical trial; Brandon Ryan had to go without it.

http://www.nytimes.com/2010/09/19/health/research/19trial.html

Sequential Experimental Design

- MAB models sequential experimental design!
  basic
- Each treatment has hidden expected value
  - Need to run trials to gather information
  - "Exploration"
- In hindsight, should always have used treatment with highest expected value
- **Regret = opportunity cost of exploration**

Online Advertising



Largest Use-Case of Multi-Armed Bandit Problems

Treating Lower Spine Injuries

Medtronic human array

10 mm

49 mm

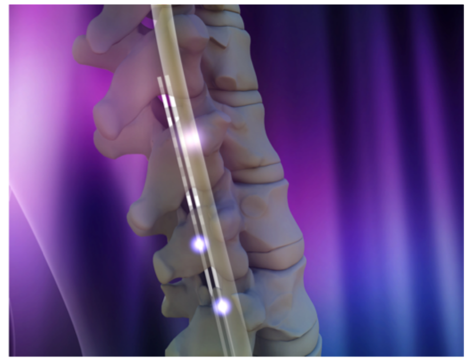


Image source: williamcapicottomd.com

SCI Patient

Each patient is unique

10^9 possible configurations!


00:05:35.05

Reinforcement Learning

Topic of CS159

Actions Impact State

- In MAB:
  - Actions do not impact state
  - Constant reward function
- Reinforcement Learning
  - Actions effect state you're in
  - Reward function depends on state

Video Demo

(Deep Reinforcement Learning for Atari)

https://www.youtube.com/watch?v=iqXKQf2BOSE

What is State?

Reward of each action varies depending on state!

Action at current state impacts future states!

Much harder to do exploration!

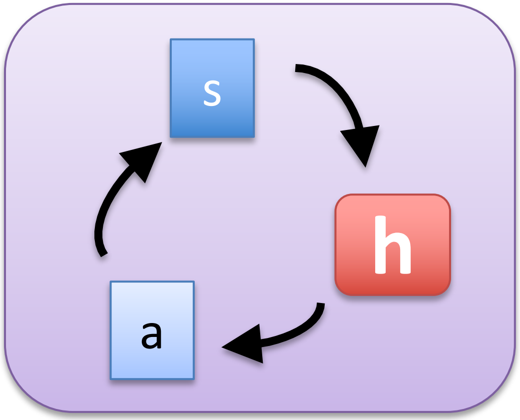
http://www.nature.com/nature/journal/v518/n7540/pdf/nature14236.pdf

Imitation Learning

Topic of CS159

Imitation Learning

- Input:
  - Sequence of contexts/states:



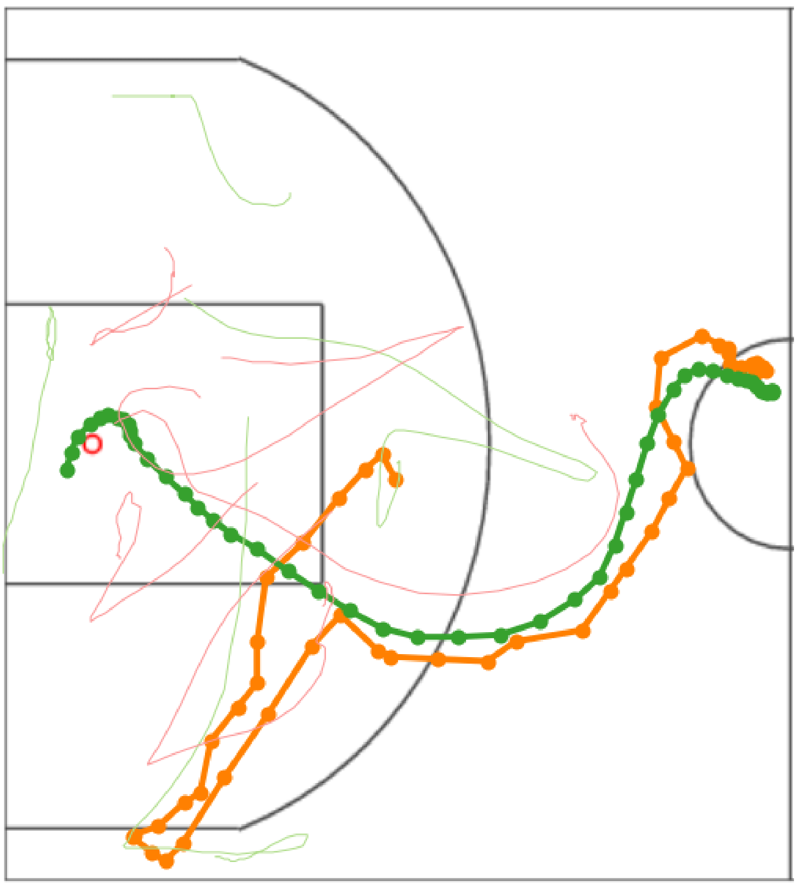
- Predict:
  - Sequence of actions
- Learn Using:
  - Sequences of demonstrated actions

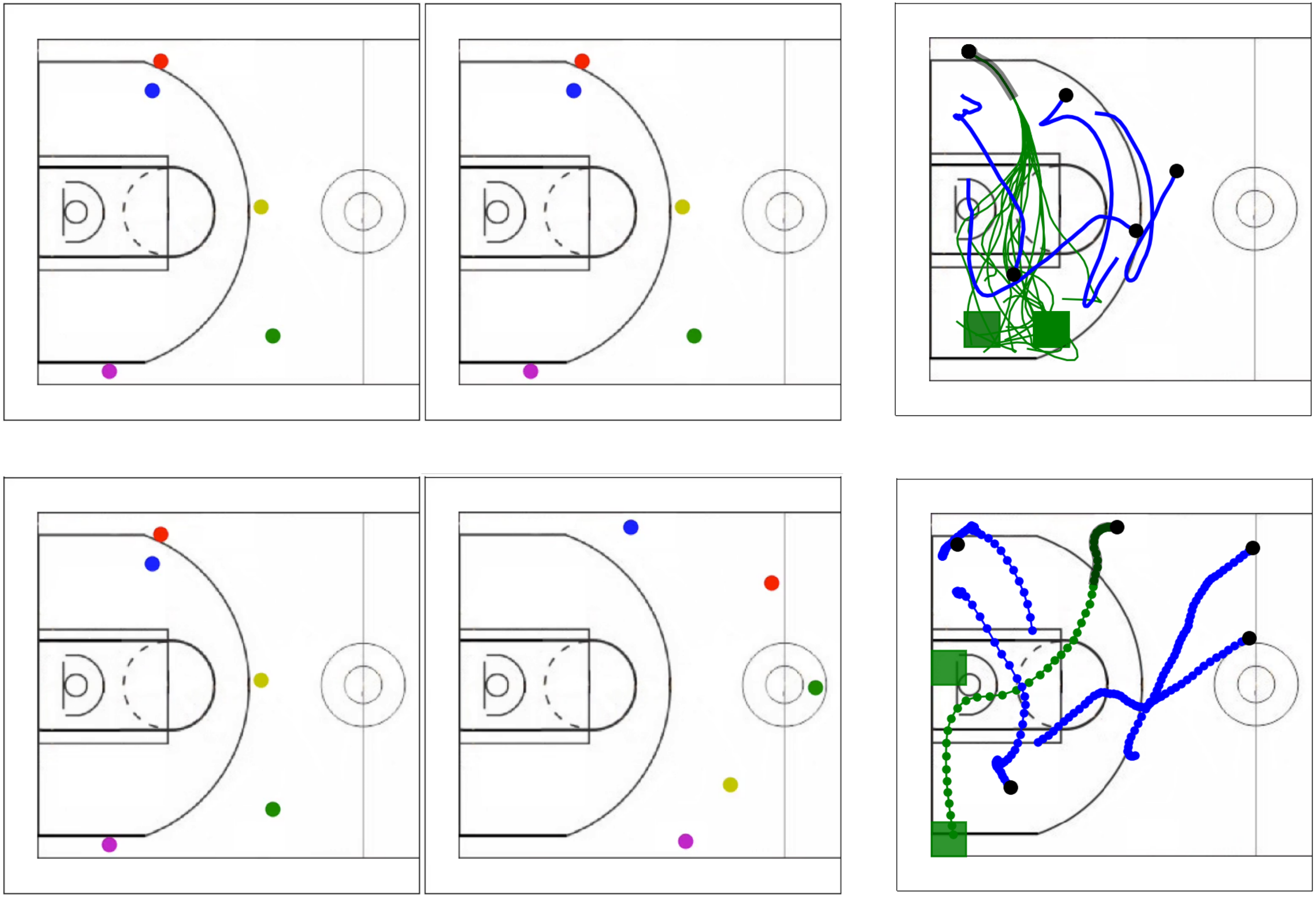
Example: Basketball Player Trajectories

- s = location of players & ball
- a = next location of player

- Training set: $D = \{(\vec{s}, \vec{a})\}$
  - $\vec{s}$ = sequence of s
  - $\vec{a}$ = sequence of a

- **Goal:** learn $h(s) \rightarrow a$

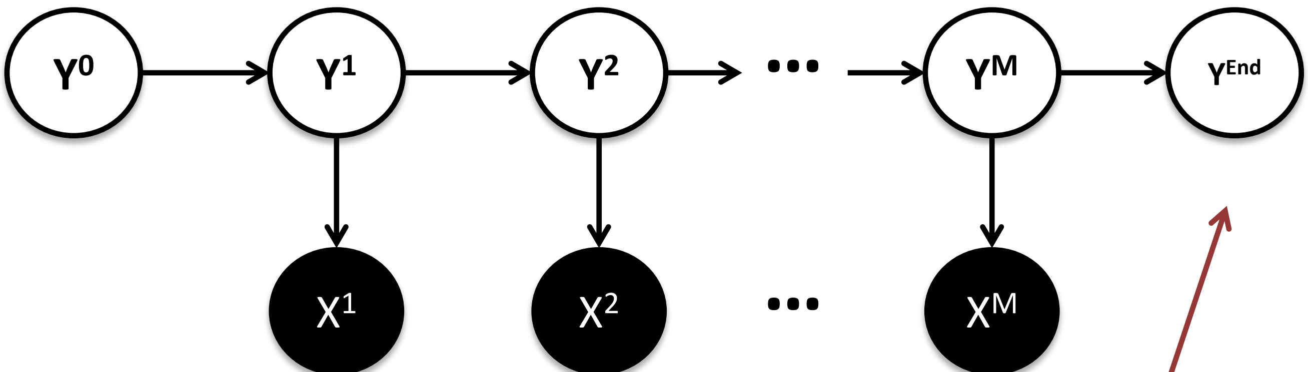




Non-Convex Optimization

Anima
Anandkumar

Recall: Hidden Markov Models



Optional

$$P(x,y) = P(End \mid y^M) \prod_{i=1}^{M} P(y^i \mid y^{i-1}) \prod_{i=1}^{M} P(x^i \mid y^i)$$

Recall: EM Algorithm for HMMs

- If we had y's → max likelihood.
- If we had (A,O) → predict y's

Chicken vs Egg!

1. Initialize A and O arbitrarily
2. Predict prob. of y's for each training x
3. Use y's to estimate new (A,O)
4. Repeat back to Step 1 until convergence

Expectation Step

Maximization Step

http://en.wikipedia.org/wiki/Baum%E2%80%93Welch_algorithm

Non-Convex Optimization Problem!
Converges to local optimum.

- If we had y's ➔ max likelihood.
- If we had (A,O) ➔ predict y's

Chicken vs Egg!

1. Initialize A and O arbitrarily

Expectation Step

2

3

4

Can We Train HMMs Optimally?

http://en.wikipedia.org/wiki/Baum%E2%80%93Welch_algorithm

Inspiration from Dimensionality Reduction

- Find best rank K approximation to Y:

$$\underset{U \in R^{NxK}, V \in R^{MxK}}{\operatorname{argmin}} \left\| Y - UV^T \right\|_2^2$$

- Non-convex optimization problem!
  - Due to non-convex feasible region

- **But optimally solved via SVD!**

Spectral Learning of HMMs

Want to Estimate:

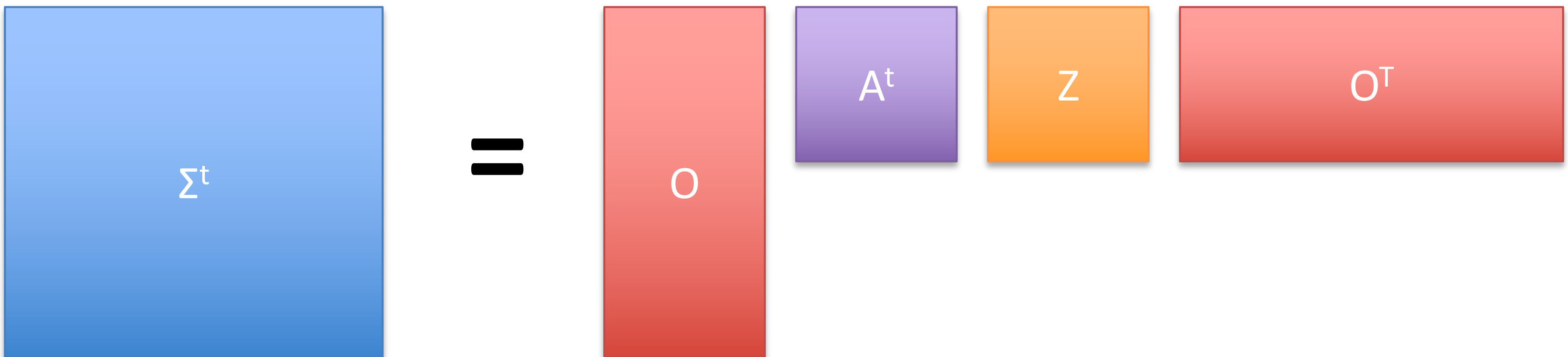
$$P(y^j \mid y^{j-1}) = A \qquad P(x^j \mid y^j) = O$$

$$\begin{aligned}
\Sigma^t = E\left[x^{j+t}\left(x^j\right)^T\right] &= E\left[E\left[x^{j+t}\left(x^j\right)^T \middle| y^j\right]\right] \\
&= E\left[E\left[x^{j+t} \middle| y^j\right]E\left[\left(x^j\right)^T \middle| y^j\right]\right] \\
&= E\left[\left(OA^t k y^j\right)\left(O y^j\right)^T\right] \\
&= OA^t E\left[y^j\left(y^j\right)^T\right]O^T \\
&= OA^t Z O^T
\end{aligned}$$

Treat each x^j and y^j as indicator vector

http://www.cs.cmu.edu/~ggordon/spectral-learning/

Spectral Learning of HMMs



Rank-K SVD of Σ^1

Optimal Solution: $$A = U^T \Sigma^2 \left(U^T \Sigma^1\right)^{-1}$$

(requires a lot of data)

http://www.cs.cmu.edu/~ggordon/spectral-learning/

…and many more topics!

- Probabilistic Models & Bayesian Reasoning
- Representation Learning
  - Deep learning is the most visible example
- Causal Reasoning
- ML + Game Theory
- ML + Systems
  - Large Scale Machine Learning
- Fairness & Privacy
- Etc …

CS 159

- Special Topics in Machine Learning
  - Taught Every Spring Term
  - Topics Rotate
- **Next Term:**
  - Interactive Machine Learning
  - Reinforcement Learning & Imitation Learning
- Paper Reading & Presenting + Final Project
  - Graded on participation and final project