Batch Learning from Bandit Feedback

(Adith Swaminathan), Thorsten Joachims

Department of Computer Science
Department of Information Science
Cornell University

Funded in part through NSF Awards IIS-1247637, IIS-1217686, IIS- 1513692.

Batch Learning from Bandit Feedback

- Data

$$S = \big((x_1, y_1, \delta_1), \ldots, (x_n, y_n, \delta_n)\big)$$

context

π_0 action

reward / loss

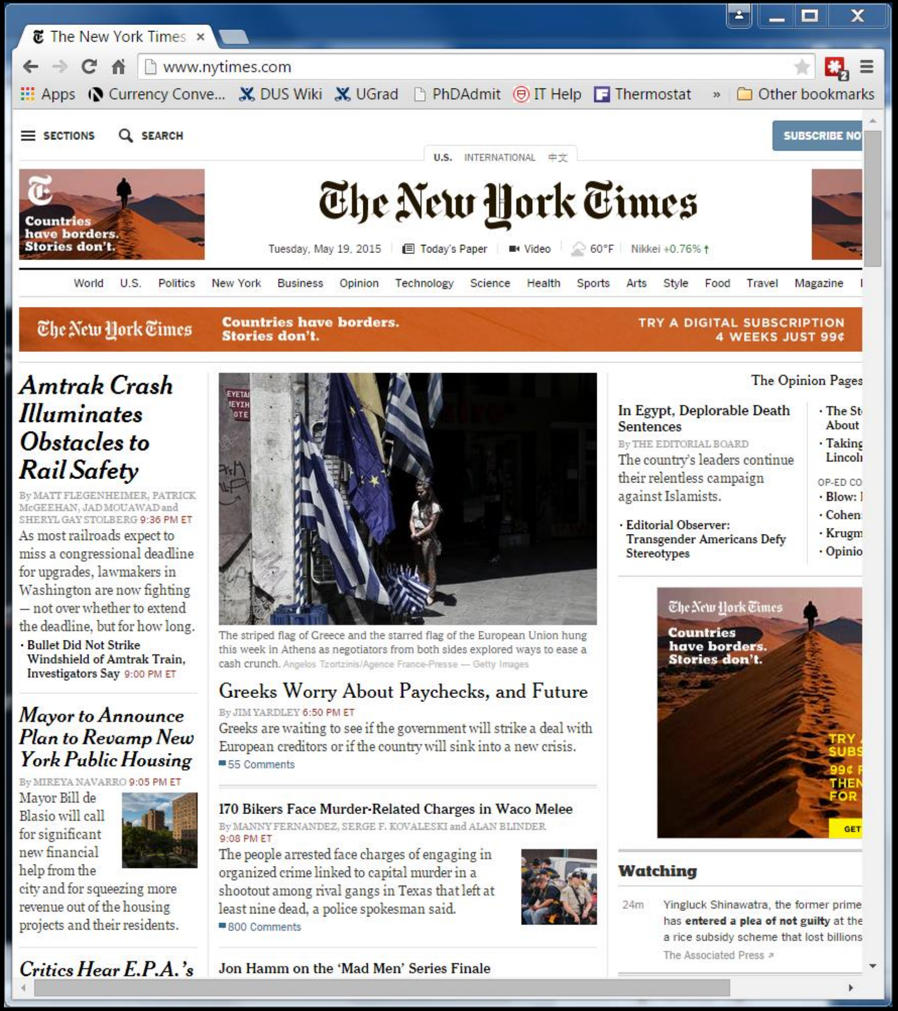
→ Partial Information (aka “Bandit”) Feedback

- Properties
  – Contexts x_i drawn i.i.d. from unknown $P(X)$
  – Actions y_i selected by existing system π_0: $X \to Y$
  – Feedback δ_i drawn i.i.d. from unknown δ: $X \times Y \to \Re$
- Goal of Learning
  – Find new system π that selects y with better δ

[Zadrozny et al., 2003] [Langford & Li], [Bottou, et al., 2014]

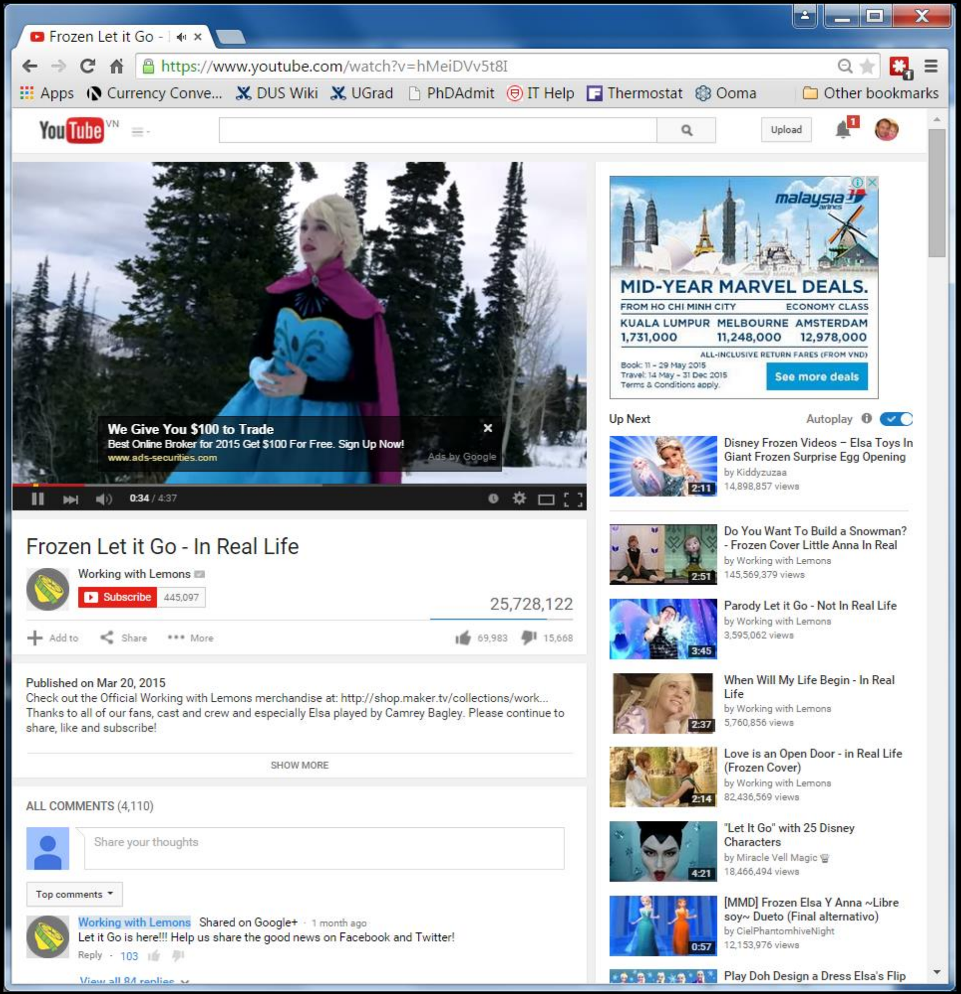
Historic Interaction Logs: News Recommender

- Context x:
  - User
- Action y:
  - Portfolio of newsarticles
- Feedback $\delta(x, y)$:
  - Reading time in minutes



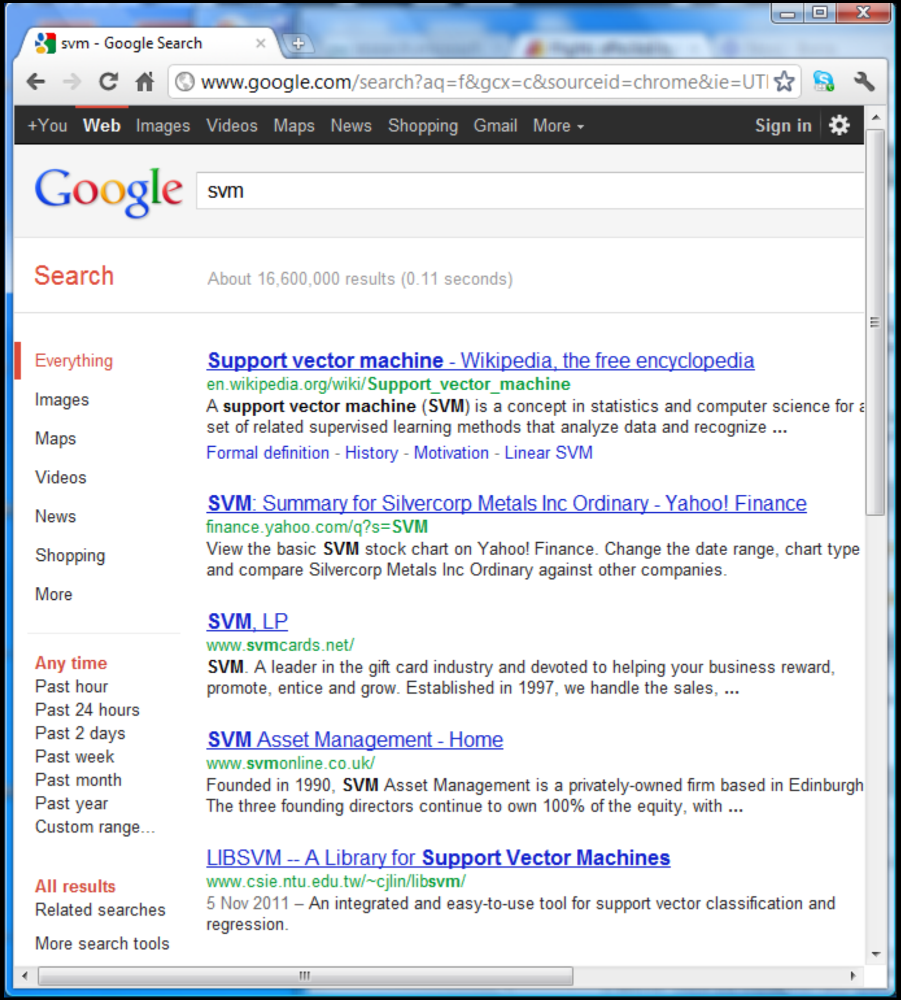
Historic Interaction Logs: Ad Placement

- Context x:
  - User and page
- Action y:
  - Ad that is placed
- Feedback $\delta(x, y)$:
  - Click / no-click



Historic Interaction Logs: Search Engine

- Context x:
  - Query
- Action y:
  - Ranking
- Feedback $\delta(x, y)$:
  - win/loss against baseline in interleaving



Comparison with Supervised Learning

	Batch Learning from Bandit Feedback	**Full-Information Supervised Learning**
Train example	(x, y, δ)	(x, y^*)
Context x	drawn i.i.d. from unknown $P(X)$	drawn i.i.d. from unknown $P(X)$
Action y	selected by existing system π_0: $X \to Y$	N/A
Feedback δ	Observe $\delta(x, y)$ only for y chosen by π_0	Assume known loss function $\Delta(y, y^*)$ → know feedback $\delta(x, y)$ for every possible y

Learning Settings

	Full-Information (Labeled) Feedback	**Partial-Information (Bandit) Feedback**
Online Learning	• Perceptron • Winnow • Etc.	• EXP3 • UCB1 • Etc.
Batch Learning	• SVM • Random Forests • Etc.	• Offset Tree • (Off-Policy RL)

Outline of Lecture

- Batch Learning from Bandit Feedback (BLBF)

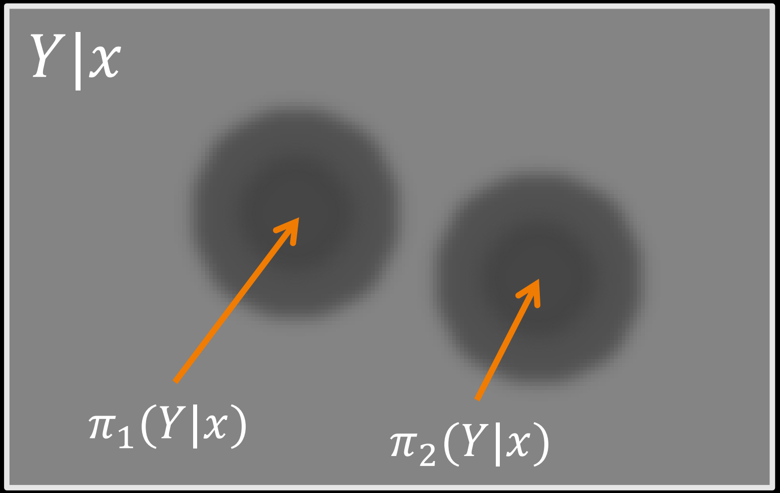
$$S = \big((x_1, y_1, \delta_1), \dots, (x_n, y_n, \delta_n)\big)$$

  → Find new system π that selects y with better δ
- Learning Principle for BLBF
  - Hypothesis Space, Risk, Empirical Risk, and Overfitting
  - Counterfactual Risk Minimization
- Learning Algorithm for BLBF
  - POEM for Structured Output Prediction
- Improved Counterfactual Risk Estimators
  - Self-Normalizing Estimator

Hypothesis Space

Definition [Stochastic Hypothesis / Policy]:

Given context x, hypothesis/policy π selects action y with probability $\pi(y|x)$



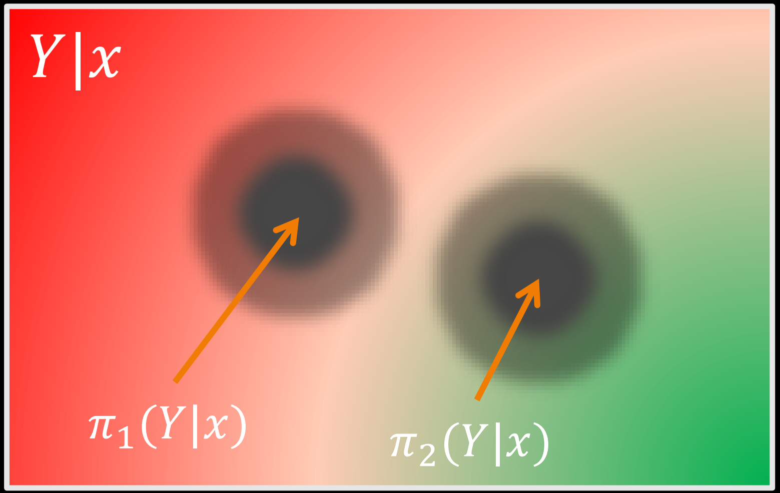
Note: stochastic prediction rules $\supset$ deterministic prediction rules

Risk

Definition [Expected Loss (i.e. Risk)]:

The expected loss / risk R(h) of policy π is

$$\mathrm{R}(\pi) = \int\int \delta(x, y)\pi(y|x)P(x)\, dx\, dy$$



On-Policy Risk Estimation

Given $S = \big((x_1, y_1, \delta_1), \ldots, (x_n, y_n, \delta_n)\big)$ collected under h_0,

$$\hat{R}(\pi_0) = \frac{1}{n} \sum_{i=1}^{n} \delta_i$$

→ A/B Testing

Field h_1: Draw $x \sim P(x)$, predict $y \sim \pi_1(Y|x\ )$, get $\delta(x, y)$
Field h_2: Draw $x \sim P(x)$, predict $y \sim \pi_2(Y|x\ )$, get $\delta(x, y)$
⋮
Field $h_{|H|}$: Draw $x \sim P(x)$, predict $y \sim \pi_{|H|}(Y|x)$, get $\delta(x, y)$

Approach 1: Model the World

- Approach [Athey & Imbens, 2015] for $Y = \{y_0, y_1\}$:
  - Learning: estimate CATE $E[\delta(x, y_1) - \delta(x, y_0)|x]$ via regression $f(x)$ from x to $\begin{cases} -\delta(x_i, y_i)/p_i & if\ y_i = y_0 \\ +\delta(x_i, y_i)/p_i & otherwise \end{cases}$
  - New policy: Given x, select $y = \begin{cases} y_0 & if\ f(x) < 0 \\ y_1 & otherwise \end{cases}$

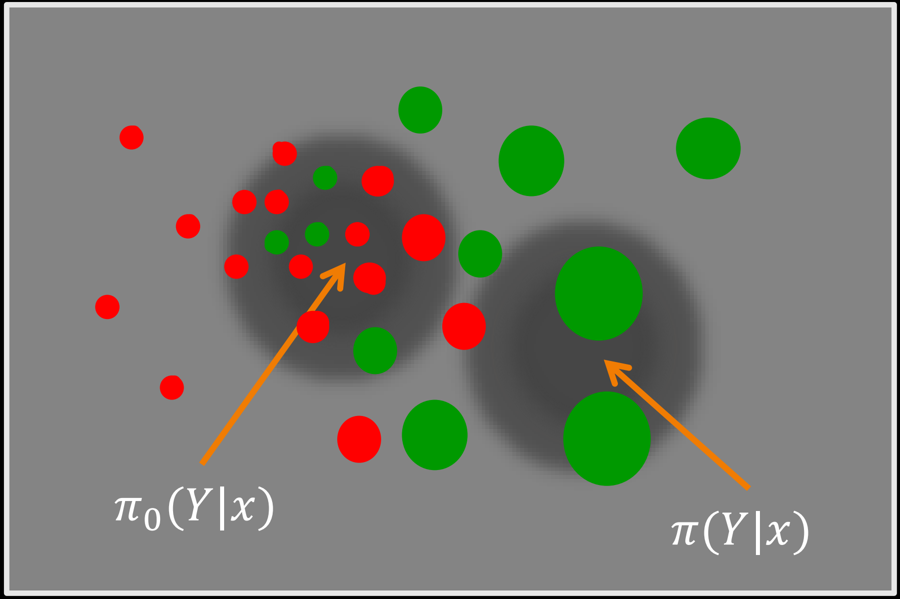
→ More general: "reward simulator approach", "model-based reinforcement learning", …

Approach 2: Model the Selection Bias

Given $S = \left((x_1, y_1, \delta_1), \ldots, (x_n, y_n, \delta_n)\right)$ collected under π_0,

$$\hat{R}(\pi) = \frac{1}{n} \sum_{i=1}^{n} \delta_i \frac{\pi(y_i|x_i)}{\pi_0(y_i|x_i)}$$

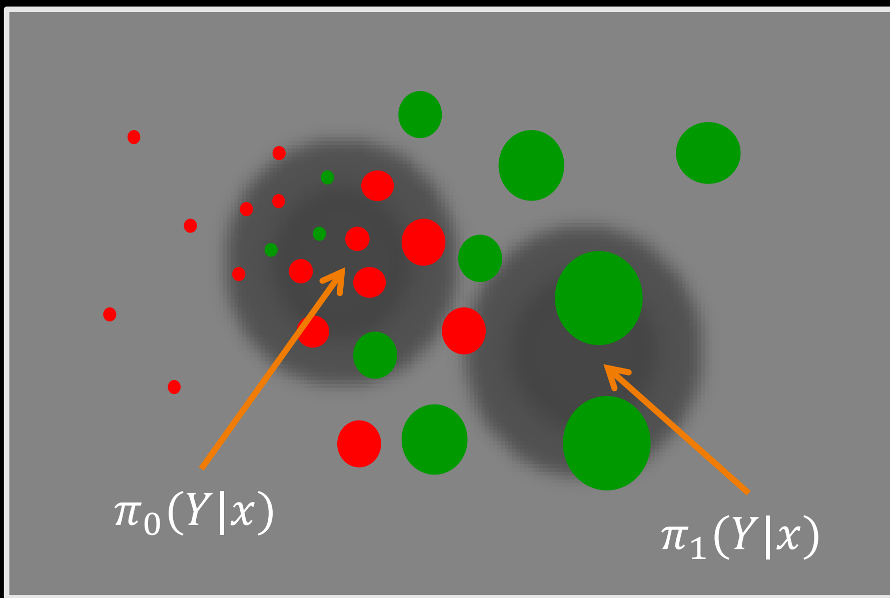
Propensity
p_i

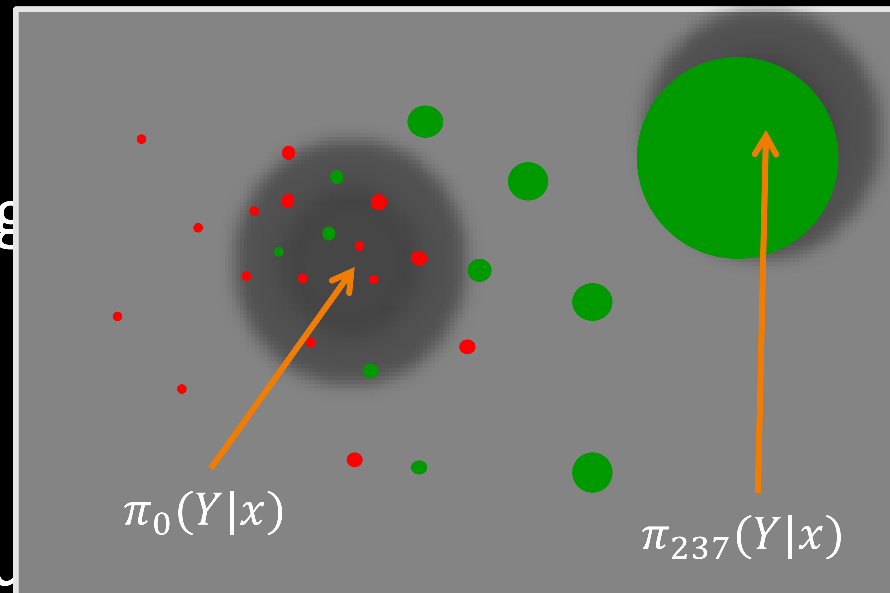


→ Get unbiased estimate of risk, if propensity nonzero everywhere (where it matters).

[Horvitz & Thompson, 1952] [Rubin, 1983] [Zadrozny et al., 2003] [Langford, Li, 2009.]

Partial Information Empirical Risk Minimization





- Training

$$\hat{h} := \operatorname{argmin}_{\pi \in H} \sum_{i}^{n} \frac{\pi(y_i|x_i)}{p_i} \delta_i$$

[Zadrozny et al., 2003] [Langford & Li], [Bottou et al., 2014]

Generalization Error Bound for BLBF

- Theorem [Generalization Error Bound]
  - For any hypothesis space H with capacity C, and for all $\pi \in H$ with probability $1 - \eta$

$$\mathrm{R}(\pi) \leq \hat{R}(\pi) + O\left(\sqrt{\widehat{Var}(\pi)/n}\right) + O(C)$$

Unbiased Estimator

Variance Control

Capacity Control

$$\hat{R}(h) = \widehat{Mean}\left(\frac{\pi(y_i|x_i)}{p_i}\delta_i\right)$$

$$\widehat{Var}(h) = \widehat{Var}\left(\frac{\pi(y_i|x_i)}{p_i}\delta_i\right)$$

→ Bound accounts for the fact that variance of risk estimator can vary greatly between different $\pi \in \mathrm{H}$

[Swaminathan & Joachims, 2015]

Counterfactual Risk Minimization

- Theorem [Generalization Error Bound]

$$\mathrm{R}(\pi) \leq \hat{R}(\pi) + O\left(\sqrt{\widehat{Var}(\pi)/n}\right) + O(C)$$

→ Constructive principle for designing learning algorithms

$$\boxed{\pi^{crm} = \underset{\pi \in H_i}{\operatorname{argmin}}\, \hat{R}(\pi) + \lambda_1 \left(\sqrt{\widehat{Var}(\pi)/n}\right) + \lambda_2 C(H_i)}$$

$$\hat{R}(\pi) = \frac{1}{n}\sum_i^n \frac{\pi(y_i|x_i)}{p_i}\delta_i \qquad \widehat{Var}(\pi) = \frac{1}{n}\sum_i^n \left(\frac{\pi(y_i|x_i)}{p_i}\,\delta_i\right)^2 - \hat{R}(\pi)^2$$

[Swaminathan & Joachims, 2015]

Outline of Lecture

- Batch Learning from Bandit Feedback (BLBF)

$$S = \left((x_1, y_1, \delta_1, p_1), \dots, (x_n, y_n, \delta_n, p_n) \right)$$

  → Find new system h that selects y with better δ
- Learning Principle for BLBF
  – Hypothesis Space, Risk, Empirical Risk, and Overfitting
  – Counterfactual Risk Minimization
- Learning Algorithm for BLBF
  – POEM for Structured Output Prediction
- Improved Counterfactual Risk Estimators
  – Self-Normalizing Estimator

POEM Hypothesis Space

Hypothesis Space: Stochastic prediction rules

$$\pi(y|x,w) = \frac{1}{Z(x)} \exp\big(w \cdot \Phi(x,y)\big)$$

with

- w: parameter vector to be learned
- $\Phi(x,y)$: joint feature map between input and output
- Z(x): partition function

Note: same form as CRF or Structural SVM

POEM Learning Method

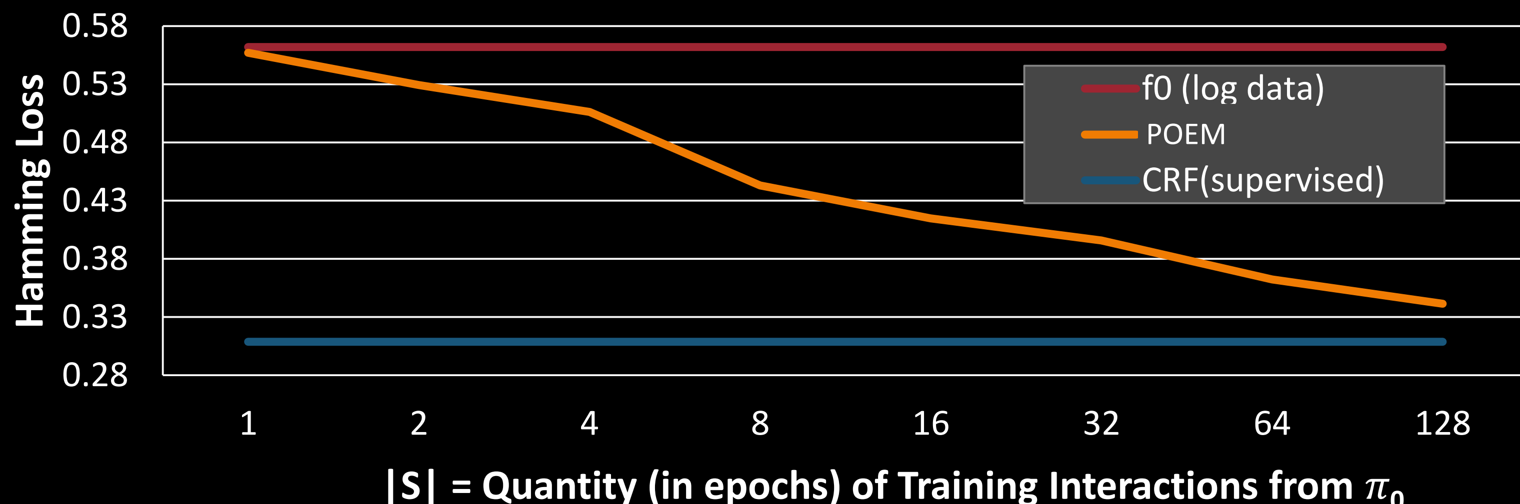
- Policy Optimizer for Exponential Models (POEM)
  - Data: $S = \big((x_1, y_1, \delta_1, p_1), \ldots, (x_n, y_n, \delta_n, p_n)\big)$
  - Hypothesis space: $\pi(y|x, w) = \exp\big(w \cdot \phi(x, y)\big)/Z(x)$
  - Training objective: Let $z_i(w) = \pi(y_i|x_i, w)\delta_i/p_i$

$$w = \underset{w \in \Re^N}{\operatorname{argmin}} \left[\underbrace{\frac{1}{n}\sum_{i=1}^{n} z_i(w)}_{\text{Unbiased Risk Estimator}} + \lambda_1 \underbrace{\sqrt{\left(\frac{1}{n}\sum_{i=1}^{n} z_i(w)^2\right) - \left(\frac{1}{n}\sum_{i=1}^{n} z_i(w)\right)^2}}_{\text{Variance Control}} + \underbrace{\lambda_2 ||w||^2}_{\text{Capacity Control}} \right]$$

[Swaminathan & Joachims, 2015]

POEM Experiment
Multi-Label Text Classification

- Data: $S = \left((x_1, y_1, \delta_1, p_1), \ldots, (x_n, y_n, \delta_n, p_n)\right)$
  - x: Text document
  - y: Predicted label vector
  - δ: number of incorrect labels in y
  - p_n: propensity under logging policy h_0
- Results: Reuters LYRL RCV1 (top 4 categories)
  - POEM with H isomorphic to CRF with one weight vector per label



[Swaminathan & Joachims, 2015]

Does Variance Regularization Improve Generalization?

- IPS: $w = \underset{w \in \Re^N}{\operatorname{argmin}} \left[\hat{R}(w) + \lambda_2 ||w||^2\right]$
- POEM: $w = \underset{w \in \Re^N}{\operatorname{argmin}} \left[\hat{R}(w) + \lambda_1 \left(\sqrt{\widehat{Var}(w)/n}\right) + \lambda_2 ||w||^2\right]$

Hamming Loss	Scene	Yeast	TMC	LYRL
h_0	1.543	5.547	3.445	1.463
IPS	1.519	4.614	3.023	1.118
POEM	**1.143**	**4.517**	**2.522**	**0.996**
# examples	4*1211	4*1500	4*21519	4*23149
# features	294	103	30438	47236
# labels	6	14	22	4

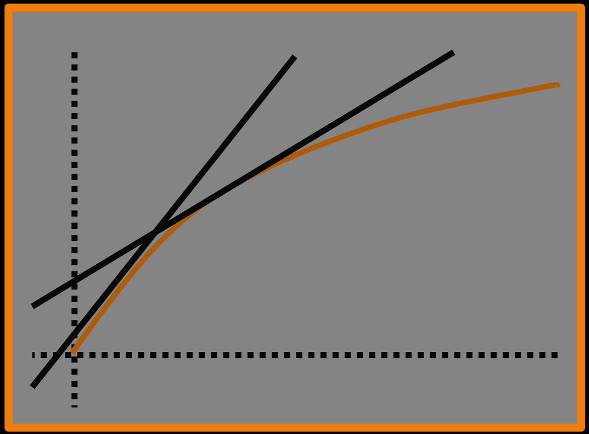
POEM Efficient Training Algorithm

- Training Objective:

$$OPT = \min_{w \in \Re^N} \left[\frac{1}{n} \sum_{i=1}^{n} z_i(w) + \lambda_1 \sqrt{ \left(\frac{1}{n} \sum_{i=1}^{n} z_i(w)^2 \right) - \left(\frac{1}{n} \sum_{i=1}^{n} z_i(w) \right)^2 } \right]$$

- Idea: First-order Taylor Majorization
  - Majorize $\sqrt{\quad}$ at current value
  - Majorize $-(\quad)^2$ at current value

$$OPT \leq \min_{w \in \Re^N} \left[\frac{1}{n} \sum_{i=1}^{n} A_i \, z_i(w) + B_i \, z_i(w)^2 \right]$$



- Algorithm:
  - Majorize objective at current w_t
  - Solve majorizing objective via Adagrad to get w_{t+1}

[De Leeuw, 1977+] [Groenen et al., 2008] [Swaminathan & Joachims, 2015]

How computationally efficient is POEM?

CPU Seconds	Scene	Yeast	TMC	LYRL
POEM	4.71	5.02	276.13	120.09
IPS	1.65	2.86	49.12	13.66
CRF (L-BFGS)	4.86	3.28	99.18	62.93
# examples	4*1211	4*1500	4*21519	4*23149
# features	294	103	30438	47236
# labels	6	14	22	4

Outline of Lecture

- Batch Learning from Bandit Feedback (BLBF)

$$S = \big((x_1, y_1, \delta_1, p_1), \ldots, (x_n, y_n, \delta_n, p_n)\big)$$

  → Find new system h that selects y with better δ
- Learning Principle for BLBF
  – Hypothesis Space, Risk, Empirical Risk, and Overfitting
  – Counterfactual Risk Minimization
- Learning Algorithm for BLBF
  – POEM for Structured Output Prediction
- Improved Counterfactual Risk Estimators
  – Self-Normalizing Estimator

Counterfactual Risk Minimization

- Theorem [Generalization Error Bound]

$$\mathrm{R}(\pi) \le \hat{R}(\pi) + O\left(\sqrt{\widehat{Var}(\pi)/n}\right) + O(C)$$

→ Constructive principle for designing learning algorithms

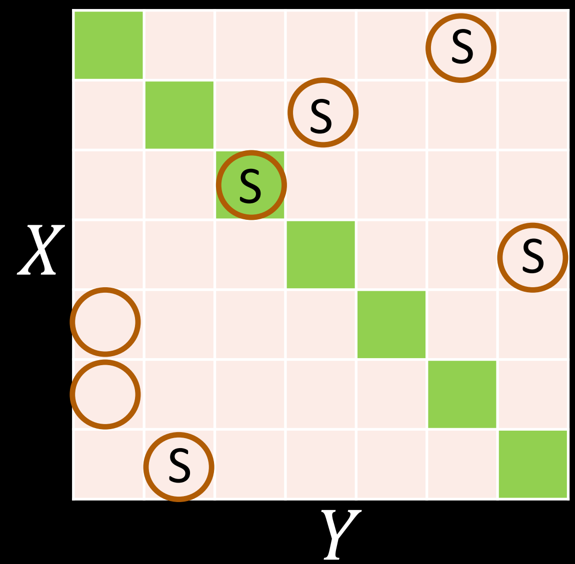
$$\boxed{\pi^{crm} = \underset{h \in H_i}{\operatorname{argmin}} \hat{R}(\pi) + \lambda_1 \left(\sqrt{\widehat{Var}(\pi)/n}\right) + \lambda_2 C(H_i)}$$

$$\hat{R}(\pi) = \frac{1}{n} \sum_{i}^{n} \frac{\pi(y_i|x_i)}{p_i} \delta_i \qquad \widehat{Var}(\pi) = \frac{1}{n} \sum_{i}^{n} \left(\frac{\pi(y_i|x_i)}{p_i} \delta_i\right)^2 - \hat{R}(\pi)^2$$

[Swaminathan & Joachims, 2015]

Propensity Overfitting Problem

- Example
  - Instance Space $X = \{1, \dots, k\}$
  - Label Space $Y = \{1, \dots, k\}$
  - Loss $\delta(x, y) = \begin{cases} -2 & if\ y == x \\ -1 & otherwise \end{cases}$
  - Training data: uniform x,y sample
  - Hypothesis space: all deterministic functions
    → $\pi_{opt}(x) = x$ with risk $R(\pi_{opt}) =$



$$R(\hat{\pi}) = \min_{\pi \in H} \frac{1}{n} \sum_{i}^{n} \frac{\pi(y_i | x_i)}{p_i} \delta_i =$$

→ Problem 1: Unbounded risk estimate!

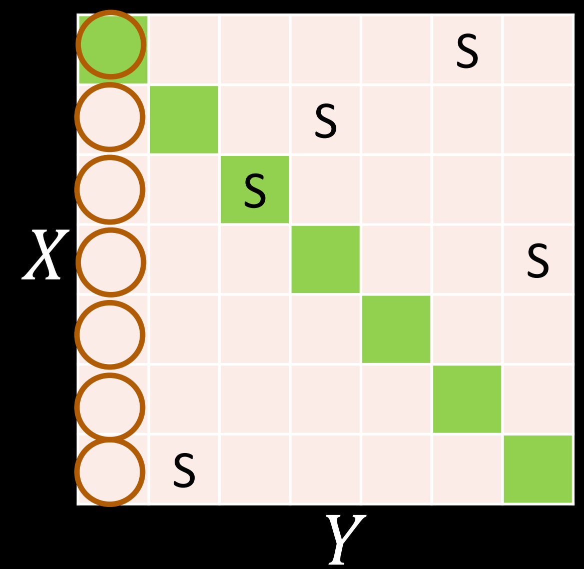
Propensity Overfitting Problem

- Example
  - Instance Space $X = \{1, \dots, k\}$
  - Label Space $Y = \{1, \dots, k\}$
  - Loss $\delta(x, y) = \begin{cases} \cancel{-2}\ 0 & if\ y == x \\ \cancel{-1}\ 1 & otherwise \end{cases}$
  - Training data: uniform x,y sample
  - Hypothesis space: all deterministic functions

    → $\pi_{opt}(x) = x$ with risk $R(\pi_{opt}) = 0$

$$R(\hat{\pi}) = \min_{\pi \in H} \frac{1}{n} \sum_{i}^{n} \frac{\pi(y_i|x_i)}{p_i} \delta_i =$$

→ Problem 2: Lack of equivariance!



Control Variates

- Idea: Inform estimate when expectation of correlated random variable is known.
  - Estimator:
$$\hat{R}(\pi) = \frac{1}{n}\sum_{i}^{n} \frac{\pi(y_i|x_i)}{p_i}\delta_i$$
  - Correlated RV with known expectation:
$$\hat{S}(\pi) = \frac{1}{n}\sum_{i}^{n} \frac{\pi(y_i|x_i)}{p_i}$$
$$E\left[\hat{S}(\pi)\right] = \frac{1}{n}\sum_{i}^{n}\int \frac{\pi(y_i|x_i)}{\pi_0(y_i|x_i)}\pi_0(y_i|x_i)P(x_i)dy_i dx_i = 1$$

→ New Risk Estimator: Self-normalizing estimator
$$\hat{R}^{SN}(\pi) = \frac{\hat{R}(\pi)}{\hat{S}(\pi)}$$

Norm-POEM Learning Method

- Method:
  - Data: $S = \big((x_1, y_1, \delta_1, p_1), \ldots, (x_n, y_n, \delta_n, p_n)\big)$
  - Hypothesis space: $\pi(y|x, w) = \exp\big(w \cdot \phi(x, y)\big)/Z(x)$
  - Training objective: Let $z_i(w) = \pi(y_i|x_i, w)\delta_i/p_i$

$$w = \underset{w \in \Re^N}{\operatorname{argmin}} \left[\underbrace{\hat{R}^{SN}(w)}_{\text{Self-Normalized Risk Estimator}} + \lambda_1 \underbrace{\sqrt{\widehat{Var}\big(\hat{R}^{SN}(w)\big)}}_{\text{Variance Control}} + \underbrace{\lambda_2 ||w||^2}_{\text{Capacity Control}} \right]$$

[Swaminathan & Joachims, 2015]

How well does Norm-POEM generalize?

Hamming Loss	**Scene**	**Yeast**	**TMC**	**LYRL**
h_0	1.511	5.577	3.442	1.459
POEM	1.200	4.520	2.152	0.914
Norm-POEM	**1.045**	**3.876**	**2.072**	**0.799**
# examples	4*1211	4*1500	4*21519	4*23149
# features	294	103	30438	47236
# labels	6	14	22	4

Conclusions

- Batch Learning from Bandit Feedback (BLBF)

$$S = \big((x_1, y_1, \delta_1, p_1), \dots, (x_n, y_n, \delta_n, p_n)\big)$$

- Learning Principle for BLBF
  → Counterfactual Risk Minimization
- Learning Algorithm for BLBF
  → POEM for Structured Output Prediction
  → Efficient Training Method
- Open Questions
  - Counterfactual Risk Estimators
    → Self-normalizing Estimator
  - Exploiting Smoothness in Loss Space
  - Exploiting Smoothness in Predictor Space
  - Propensity Estimation