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

\[ S = \left( (x_1, y_1, \delta_1), \ldots, (x_n, y_n, \delta_n) \right) \]

→ Partial Information (aka “Bandit”) Feedback

• Properties

– Contexts \( x_i \) drawn i.i.d. from unknown \( P(X) \)
– Actions \( y_i \) selected by existing system \( \pi_0 : X \rightarrow Y \)
– Feedback \( \delta_i \) drawn i.i.d. from unknown \( \delta : X \times Y \rightarrow \mathbb{R} \)

• Goal of Learning

– Find new system \( \pi \) that selects \( y \) with better \( \delta \)

[Zadrozny et al., 2003] [Langford & Li], [Bottou, et al., 2014]
Historic Interaction Logs: News Recommender

• Context $x$:
  – User

• Action $y$:
  – Portfolio of news articles

• 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

<table>
<thead>
<tr>
<th></th>
<th>Batch Learning from Bandit Feedback</th>
<th>Full-Information Supervised Learning</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Train example</strong></td>
<td>$(x, y, \delta)$</td>
<td>$(x, y^*)$</td>
</tr>
<tr>
<td><strong>Context $x$</strong></td>
<td>drawn i.i.d. from unknown $P(X)$</td>
<td>drawn i.i.d. from unknown $P(X)$</td>
</tr>
<tr>
<td><strong>Action $y$</strong></td>
<td>selected by existing system $\pi_0: X \rightarrow Y$</td>
<td>N/A</td>
</tr>
<tr>
<td><strong>Feedback $\delta$</strong></td>
<td>Observe $\delta(x, y)$ only for $y$ chosen by $\pi_0$</td>
<td>Assume known loss function $\Delta(y, y^*)$ \rightarrow know feedback $\delta(x, y)$ for every possible $y$</td>
</tr>
</tbody>
</table>
## Learning Settings

<table>
<thead>
<tr>
<th></th>
<th>Full-Information (Labeled) Feedback</th>
<th>Partial-Information (Bandit) Feedback</th>
</tr>
</thead>
</table>
| **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)
  \[ S = ((x_1, y_1, \delta_1), \ldots, (x_n, y_n, \delta_n)) \]
  \[ \rightarrow \text{Find new system } \pi \text{ that selects } y \text{ with better } \delta \]

• 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 $\pi$ 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 $\pi$ is

$$R(\pi) = \int \int \delta(x, y)\pi(y|x)P(x) \, dx \, dy$$
On-Policy Risk Estimation

Given $S = ((x_1, y_1, \delta_1), \ldots, (x_n, y_n, \delta_n))$ 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
    $$f(x) \text{ from } x \text{ to } \begin{cases} -\delta(x_i, y_i)/p_i & \text{if } y_i = y_0 \\ +\delta(x_i, y_i)/p_i & \text{otherwise} \end{cases}$$
  - New policy: Given $x$, select $y = \begin{cases} y_0 & \text{if } f(x) < 0 \\ y_1 & \text{otherwise} \end{cases}$

→ More general: “reward simulator approach”, “model-based reinforcement learning”, ...
Approach 2: Model the Selection Bias

Given $S = ((x_1, y_1, \delta_1), \ldots, (x_n, y_n, \delta_n))$ collected under $\pi_0$,

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

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

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

- Setup
  - Stochastic logging using $h_0$ with $p_i = \pi_0(y_i | x_i)$
  - Data $S = (x_1, y_1, \delta_1, p_1, \ldots, x_n, y_n, \delta_n, p_n)$
  - Stochastic prediction rules $\pi \in H$: $\pi(y_i | x_i)$

- Training
  - $h := \arg\min_{\pi \in H} \sum_i^n \frac{\pi(y_i | x_i)}{p_i} \delta_i$

Partial Information

\[ [\text{Zadrozny et al., 2003}] [\text{Langford \& Li}, [\text{Bottou et al., 2014}]\]
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 $1 - \eta$
  
  $$R(\pi) \leq \hat{R}(\pi) + O\left(\sqrt{\text{Var}(\pi)/n}\right) + O(C)$$

  $$\hat{R}(h) = \overline{\text{Mean}} \left( \frac{\pi(y_i|x_i)}{\rho_i} \delta_i \right)$$

  $$\sqrt{\text{ar}}(h) = \sqrt{\text{ar}} \left( \frac{\pi(y_i|x_i)}{\rho_i} \delta_i \right)$$

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

[Swaminathan & Joachims, 2015]
Counterfactual Risk Minimization

• Theorem [Generalization Error Bound]

\[ R(\pi) \leq \hat{R}(\pi) + O\left(\sqrt{\text{Var}(\pi)/n}\right) + O(C) \]

→ Constructive principle for designing learning algorithms

\[
\pi^{crm} = \arg\min_{\pi \in H_i} \hat{R}(\pi) + \lambda_1 \left(\sqrt{\text{Var}(\pi)/n}\right) + \lambda_2 C(H_i)
\]

\[
\hat{R}(\pi) = \frac{1}{n} \sum_{i}^{n} \pi(y_i|x_i) \frac{\delta_i}{p_i}
\]

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

[Swaminathan & Joachims, 2015]
Outline of Lecture

• Batch Learning from Bandit Feedback (BLBF)
  \[ S = \left( (x_1, y_1, \delta_1, p_1), \ldots, (x_n, y_n, \delta_n, p_n) \right) \]
  \[ \rightarrow \text{Find new system } h \text{ that selects } y \text{ with better } \delta \]

• 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

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

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

$$w = \arg\min_{w \in \mathbb{R}^N} \left[ \frac{1}{n} \sum_{i=1}^{n} z_i(w) + \lambda_1 \sqrt{\left( \frac{1}{n} \sum_{i=1}^{n} z_i(w)^2 \right) - \left( \frac{1}{n} \sum_{i=1}^{n} z_i(w) \right)^2} + \lambda_2 \|w\|^2 \right]$$

Unbiased Risk Estimator
Variance Control
Capacity Control

[Swaminathan & Joachims, 2015]
POEM Experiment

Multi-Label Text Classification

• Data: $S = \{(x_1, y_1, \delta_1, p_1), \ldots, (x_n, y_n, \delta_n, p_n)\}$
  - $x$: Text document
  - $y$: Predicted label vector
  - $\delta$: 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

|$|S|$ = Quantity (in epochs) of Training Interactions from $\pi_0$

[Swaminathan & Joachims, 2015]
Does Variance Regularization Improve Generalization?

- **IPS:**
  \[ w = \arg\min_{w \in \mathbb{R}^N} \left[ \hat{R}(w) + \lambda_2 ||w||^2 \right] \]

- **POEM:**
  \[ w = \arg\min_{w \in \mathbb{R}^N} \left[ \hat{R}(w) + \lambda_1 \left( \sqrt{\text{Var}(w)}/n \right) + \lambda_2 ||w||^2 \right] \]

<table>
<thead>
<tr>
<th>Hamming Loss</th>
<th>Scene</th>
<th>Yeast</th>
<th>TMC</th>
<th>LYRL</th>
</tr>
</thead>
<tbody>
<tr>
<td>( h_0 )</td>
<td>1.543</td>
<td>5.547</td>
<td>3.445</td>
<td>1.463</td>
</tr>
<tr>
<td>IPS</td>
<td>1.519</td>
<td>4.614</td>
<td>3.023</td>
<td>1.118</td>
</tr>
<tr>
<td>POEM</td>
<td><strong>1.143</strong></td>
<td><strong>4.517</strong></td>
<td><strong>2.522</strong></td>
<td><strong>0.996</strong></td>
</tr>
</tbody>
</table>

| # 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:

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

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

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

• 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?

<table>
<thead>
<tr>
<th>CPU Seconds</th>
<th>Scene</th>
<th>Yeast</th>
<th>TMC</th>
<th>LYRL</th>
</tr>
</thead>
<tbody>
<tr>
<td>POEM</td>
<td>4.71</td>
<td>5.02</td>
<td>276.13</td>
<td>120.09</td>
</tr>
<tr>
<td>IPS</td>
<td>1.65</td>
<td>2.86</td>
<td>49.12</td>
<td>13.66</td>
</tr>
<tr>
<td>CRF (L-BFGS)</td>
<td>4.86</td>
<td>3.28</td>
<td>99.18</td>
<td>62.93</td>
</tr>
<tr>
<td># examples</td>
<td>4*1211</td>
<td>4*1500</td>
<td>4*21519</td>
<td>4*23149</td>
</tr>
<tr>
<td># features</td>
<td>294</td>
<td>103</td>
<td>30438</td>
<td>47236</td>
</tr>
<tr>
<td># labels</td>
<td>6</td>
<td>14</td>
<td>22</td>
<td>4</td>
</tr>
</tbody>
</table>
Outline of Lecture

- Batch Learning from Bandit Feedback (BLBF)
  \[ S = \left( (x_1, y_1, \delta_1, p_1), \ldots, (x_n, y_n, \delta_n, p_n) \right) \]
  → Find new system \( h \) that selects \( y \) with better \( \delta \)

- 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]

\[ R(\pi) \leq \hat{R}(\pi) + O \left( \sqrt{\text{Var}(\pi)/n} \right) + O(C) \]

→ Constructive principle for designing learning algorithms

\[
\pi^{crm} = \arg\min_{h \in H_i} \hat{R}(\pi) + \lambda_1 \left( \sqrt{\text{Var}(\pi)/n} \right) + \lambda_2 C(H_i)
\]

\[
\hat{R}(\pi) = \frac{1}{n} \sum_{i} \pi(y_i|x_i) \frac{\delta_i}{p_i}
\]

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

[Swaminathan & Joachims, 2015]
Propensity Overfitting Problem

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

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

\[ \rightarrow \text{Problem 1: Unbounded risk estimate!} \]
Propensity Overfitting Problem

- **Example**
  - Instance Space $X = \{1, \ldots, k\}$
  - Label Space $Y = \{1, \ldots, k\}$
  - Loss $\delta(x, y) = \begin{cases} -2 & \text{if } y = x \\ -1 & \text{otherwise} \end{cases}$
  - Training data: uniform $x, y$ sample
  - Hypothesis space: all deterministic functions
    - $\pi_{opt}(x) = x$ with risk $R(\pi_{opt}) = 0$

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

  - 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:
    \[ \hat{S}(\pi) = \frac{1}{n} \sum_{i}^{n} \frac{\pi(y_{i}|x_{i})}{p_{i}} \]
    \[ E[\hat{S}(\pi)] = \frac{1}{n} \sum_{i}^{n} \int \frac{\pi(y_{i}|x_{i})}{\pi_{0}(y_{i}|x_{i})} \pi_{0}(y_{i}|x_{i})P(x_{i})dy_{i}dx_{i} = 1 \]
  → New Risk Estimator: Self-normalizing estimator
    \[ \hat{R}^{SN}(\pi) = \frac{\hat{R}(\pi)}{\hat{S}(\pi)} \]
Norm-POEM Learning Method

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

\[
    w = \arg\min_{w\in\mathbb{R}^N} \left[ \hat{R}^{SN}(w) + \lambda_1 \sqrt{\text{Var}(\hat{R}^{SN}(w))} + \lambda_2 \|w\|^2 \right]
\]

- **Self-Normalized Risk Estimator**
- **Variance Control**
- **Capacity Control**

[Swaminathan & Joachims, 2015]
How well does Norm-POEM generalize?

<table>
<thead>
<tr>
<th>Hamming Loss</th>
<th>Scene</th>
<th>Yeast</th>
<th>TMC</th>
<th>LYRL</th>
</tr>
</thead>
<tbody>
<tr>
<td>$h_0$</td>
<td>1.511</td>
<td>5.577</td>
<td>3.442</td>
<td>1.459</td>
</tr>
<tr>
<td>POEM</td>
<td>1.200</td>
<td>4.520</td>
<td>2.152</td>
<td>0.914</td>
</tr>
<tr>
<td>Norm-POEM</td>
<td><strong>1.045</strong></td>
<td><strong>3.876</strong></td>
<td><strong>2.072</strong></td>
<td><strong>0.799</strong></td>
</tr>
<tr>
<td># examples</td>
<td>4*1211</td>
<td>4*1500</td>
<td>4*21519</td>
<td>4*23149</td>
</tr>
<tr>
<td># features</td>
<td>294</td>
<td>103</td>
<td>30438</td>
<td>47236</td>
</tr>
<tr>
<td># labels</td>
<td>6</td>
<td>14</td>
<td>22</td>
<td>4</td>
</tr>
</tbody>
</table>
Conclusions

• Batch Learning from Bandit Feedback (BLBF)
  \[ S = \{(x_1, y_1, \delta_1, p_1), \ldots, (x_n, y_n, \delta_n, p_n)\} \]

• Learning Principle for BLBF
  \[ \rightarrow \] Counterfactual Risk Minimization

• Learning Algorithm for BLBF
  \[ \rightarrow \] POEM for Structured Output Prediction
  \[ \rightarrow \] Efficient Training Method

• Open Questions
  – Counterfactual Risk Estimators
    \[ \rightarrow \] Self-normalizing Estimator
  – Exploiting Smoothness in Loss Space
  – Exploiting Smoothness in Predictor Space
  – Propensity Estimation