Learning with Humans in the Loop

CS4780/5780 – Machine Learning
Fall 2014
Thorsten Joachims
Cornell University

Optional Reading:

User-Facing Machine Learning
- Examples
  - Search Engines
  - Netflix
  - Smart Home
  - Robot Assistant
- Learning
  - Gathering and maintenance of knowledge
  - Measure and optimize performance
  - Personalization

Interactive Learning System

\[ y \text{ dependent on } x_t \]
\[ x_{t+1} \text{ dependent on } y_t \]

\[ \text{Utility: } U(y_t) \]

Decide between two Ranking Functions

Distribution \( P(u, q) \) of users \( u \), queries \( q \)

\[ f_1(u, q) \rightarrow r_1 \]
\[ f_2(u, q) \rightarrow r_2 \]

Which one is better?

Measuring Utility

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Aggregation</th>
<th>Hypothesized Change with Decreased Quality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abandonment Rate</td>
<td>% of queries with no click</td>
<td>N/A</td>
<td>Increase</td>
</tr>
<tr>
<td>Reformulation Rate</td>
<td>% of queries that are followed by reformulation</td>
<td>N/A</td>
<td>Increase</td>
</tr>
<tr>
<td>Queries per Session</td>
<td>Session = no interruption of more than 30 minutes</td>
<td>Mean</td>
<td>Increase</td>
</tr>
<tr>
<td>Clicks per Query</td>
<td>Number of clicks</td>
<td>Mean</td>
<td>Decrease</td>
</tr>
<tr>
<td>Click@1</td>
<td>% of queries with clicks at position 1</td>
<td>N/A</td>
<td>Decrease</td>
</tr>
<tr>
<td>Max Reciprocal Rank*</td>
<td>1/rank for highest click</td>
<td>Mean</td>
<td>Decrease</td>
</tr>
<tr>
<td>Mean Reciprocal Rank*</td>
<td>Mean of 1/rank for all clicks</td>
<td>Mean</td>
<td>Decrease</td>
</tr>
<tr>
<td>Time to First Click*</td>
<td>Seconds before first click</td>
<td>Median</td>
<td>Increase</td>
</tr>
<tr>
<td>Time to Last Click*</td>
<td>Seconds before final click</td>
<td>Median</td>
<td>Decrease</td>
</tr>
</tbody>
</table>

* only queries with at least one click count

ArXiv.org: User Study

User Study in ArXiv.org
- Natural user and query population
- User in natural context, not lab
- Live and operational search engine
- Ground truth by construction
  - ORIG: Hand-tuned fielded
  - SWAP2: ORIG with 2 pairs swapped
  - SWAP4: ORIG with 4 pairs swapped
  - ORIG > FLAT > RAND
  - ORIG Hand-tuned fielded
  - FLAT: No field weights
  - RAND: Top 10 of FLAT shuffled

[ArXiv.org: User Study]
**ArXiv.org: Experiment Setup**

- **Experiment Setup**
  - Phase I: 36 days
    - Users randomly receive ranking from Orig, Flat, Rand
  - Phase II: 30 days
    - Users randomly receive ranking from Orig, Swap2, Swap4
  - Users are permanently assigned to one experimental condition based on IP address and browser.
- **Basic Statistics**
  - ~700 queries per day / ~300 distinct users per day
- **Quality Control and Data Cleaning**
  - Test run for 32 days
  - Heuristics to identify bots and spammers
  - All evaluation code was written twice and cross-validated

---

**Yahoo! Search: Results**

- **Retrieval Functions**
  - 4 variants of production retrieval function
- **Data**
  - 10M – 70M queries for each retrieval function
  - Expert relevance judgments
- **Results**
  - Still not always significant even after more than 10M queries per function
  - Only Click@1 consistent with DCG@5

---

**Interactive Learning System**

- **Algorithm**
  - Utility: $U(y_t)$
  - $y_t$ dependent on $x_t$ (e.g. ranking for query)
- **User**
  - $x_{t+1}$ dependent on $y_t$ (e.g. click given ranking, new query)

- **Observed Data ≠ Training Data**
  - Observed data is user's decisions
  - Even explicit feedback reflects user's decision process
- **Decisions → Feedback → Learning Algorithm**
  - Model the users decision process to extract feedback
  - Design learning algorithm for this type of feedback

---

**Decide between two Ranking Functions**

Distribution $P(u,q)$ of users $u$, queries $q$.

- **Retrieval Function 1**
  - $f_1(u,q) \rightarrow r_1$
- **Retrieval Function 2**
  - $f_2(u,q) \rightarrow r_2$

Which one is better?

$U(tj, "SVM", r_1)$

$U(tj, "SVM", r_2)$

---

**A Model of how Users Click in Search**

- **Model of clicking**:
  - Users explore ranking to position $k$
  - Users click on most relevant (looking) links in top $k$
  - Users stop clicking when time budget up or other action more promising (e.g. reformulation)
  - Empirically supported by [Granka et al., 2004]

---

**Conclusions**

- None of the absolute metrics reflects expected order.
- Most differences not significant after one month of data.
- Absolute metrics not suitable for ArXiv-sized search engines.

---

[Chapelle et al., 2012]

[Granka et al., 2004]
Experiment Setup

- Bing Web Search [Radlinski & Craswell, 2010]
- Model the user's decision process to extract quality control and data cleaning

Balanced Interleaving

- Five retrieval function pairs
- 10 possible pairs for interactive experiment
- Team Balanced Interleaving of (A,B)

Yahoo Web Search [Chapelle et al., 2012]

Observed Data ≠ Training Data

Example:
4 retrieval functions: A > B >> C > D

Minimizing Regret

Definition: Probability of \( P(f^* > f_j) \) losing against the best \( f^* \)

\[
R(A) = \sum_{t=1}^{T} [P(f^* > f_j) - 0.5] + [P(f^* > f_j) - 0.5]
\]

Dueling Bandits Problem

- [Kus, Broder, Kleinberg, Kraehnke, 2010]
Noisy Sorting/Max Algorithms:
- [Feige et al.]: Triangle Tournament Heap $O(n^{1/2} \log(1/\delta))$ with prob 1-\delta
- [Adler et al., Karp & Kleinberg]: optimal under weaker assumptions

Algorithm: Interleaved Filter 2
- Algorithm InterleavedFilter1($T, W=\{f_1, ..., f_K\}$)
  - Pick random $f'$ from $W$
  - $\delta=1/(TK^2)$
  - WHILE $|W|>1$
    - FOR $b \in W$ DO
      - duel($f', f$)
      - update $P_t$
      - $t=t+1$
    - $c=\log(1/\delta)/n^\delta$ \footnote{WORSE with prob 1-0.5}
    - Remove all $f$ from $W$ with $P_t < 0.5-c$ \footnote{BETTER with prob 1-0.5}
    - Remove $f'$ from $W$
  - UNTIL T: duel($f', f$)

Assumptions
- Preference Relation: $f_i > f_j \Rightarrow P(f_i > f_j) = 0.5+\epsilon_{ij} > 0.5$
- Weak Stochastic Transitivity: $f_i > f_j$ and $f_j > f_k \Rightarrow f_i > f_k$
  \textbf{Theorem:} IF2 incurs expected average regret bounded by
  $$\frac{\epsilon_{1,2}}{2} \log\left(\frac{K}{\epsilon_{1,2}}\right)$$
- Stochastic Triangle Inequality: $f_i > f_j \Rightarrow \epsilon_{ik} = \epsilon_{ik} + \epsilon_{jk}$
  $\epsilon_{1,2} = 0.01$ and $\epsilon_{2,3} = 0.01 \Rightarrow \epsilon_{1,3} \leq 0.02$
- $\epsilon$-Winner exists: $\epsilon = \max\{P(f_i > f_j) - 0.5\} = \epsilon_{1,2} > 0$

Interactive Learning System
- Algorithm $y_i$ dependent on $x_i$ (e.g. ranking for query)
- Utility: $U(y_i)$
- Model $x_{i+1}$ dependent on $y_i$ (e.g. click given ranking, new query)

Lower Bound
- \textbf{Theorem:} Any algorithm for the dueling bandits problem has regret
  $$\frac{\epsilon_{1,2}}{2} \log\left(\frac{K}{\epsilon_{1,2}}\right)$$
- Proof: [Karp, Kleinberg, 2007] [Kleinberg et al., 2007]
- Intuition:
  - Magically guess the best bandit, just verify guess
  - Worst case: $\forall f_i > f_j: P(f_i > f_j) = 0.5+c$
  - Need $O(1/c^2 \log T)$ duels to get $1-1/T$ confidence.

Interactive Learning System
- Algorithm $y_i$ dependent on $x_i$ (e.g. ranking for query)
- Utility: $U(y_i)$
- Model $x_{i+1}$ dependent on $y_i$ (e.g. click given ranking, new query)

Who does the exploring?
- \textbf{Example 1}

- Observed Data ≠ Training Data
- Decisions $\rightarrow$ Feedback $\rightarrow$ Learning Algorithm
  - Model the users decision process to extract feedback
  - Pairwise comparison test $P(y_i > y_j | U(y_i)=U(y_j))$
  - Design learning algorithm for this type of feedback
  - Dueling Bandits problem and algorithms (e.g. IF2)
Coactive Feedback Model

- Interaction: given $x$
  - Set of all $y$ for context $x$
  - Algorithm prediction
  - User explored
  - Improved Prediction
- Feedback:
  - Improved prediction $\hat{y}_t$
    $U(\hat{y}_t | x_t) > U(y_t | x_t)$
  - Supervised learning: optimal prediction $y_t^* = \text{argmax}_y U(y | x_t)$
- Relationship to other online learning models
  - Expert setting: receive $U(y | x)$ for all $y$
  - Bandit setting: receive $U(y | x)$ only for selected $y$
  - Dueling bandits: for selected $y$ and $\hat{y}$, receive $U(\hat{y} | x) > U(y | x)$
  - Coactive setting: for selected $y$, receive $\hat{y}$ with $U(\hat{y} | x) > U(y | x)$

Coactive Learning Model

- Unknown Utility Function: $U(y | x)$
  - Boundedly rational user
- Algorithm/User Interaction:
  - LOOP FOREVER
    - Observe context $x$ (e.g. query)
    - Learning algorithm presents $y$ (e.g. ranking)
    - User returns $\hat{y}$ with $U(\hat{y} | x) > U(y | x)$
    - Regret = Regret + $[U(y^* | x) - U(y_t | x)]$
- Relationship to other online learning models
  - Expert setting: receive $U(y | x)$ for all $y$
  - Bandit setting: receive $U(y | x)$ only for selected $y$
  - Dueling bandits: for selected $y$ and $\hat{y}$, receive $U(\hat{y} | x) > U(y | x)$
  - Coactive setting: for selected $y$, receive $\hat{y}$ with $U(\hat{y} | x) > U(y | x)$

Coactive Preference Perceptron

- Model
  - Linear model of user utility: $U(y | x) = w^T \phi(x, y)$
- Algorithm
  - FOR $t = 1$ TO $T$ DO
    - Observe $x_t$
    - Present $y_t = \text{argmax}_y \{ w_t^T \phi(x_t, y) \}$
    - Obtain feedback $\hat{y}_t$ from user
    - Update $w_{t+1} = w_t + \phi(x_t, \hat{y}_t) - \phi(x_t, y_t)$
  - This may look similar to a multi-class Perceptron, but
    - Feedback $\hat{y}_t$ is different (not get the correct class label)
    - Regret is different (misclassifications vs. utility difference)

Machine Translation

We propose Coactive Learning as a model of interaction between a learning system and a human user, where both have the common goal of providing results of maximum utility to the user.

Wir schlagen vor, koaktive Learning als ein Modell der Wechselwirkung zwischen einem Lernsystem und menschlichen Benutzer, wobei sowohl die gemeinsame Ziel, die Ergebnisse der maximalen Nutzen für den Benutzer zu liefern.
**Preference Perceptron: Regret Bound**

- **Assumption**
  - $U(y|x) = w^T \phi(x,y)$, but $w$ is unknown

- **Theorem**
  - For user feedback $\mathbf{y}$ that is $\alpha$-informative, the average regret of the Preference Perceptron is bounded by
  $\frac{1}{T} \sum_{t=1}^{T} \mathbb{E}[\mathbf{y}_t] - \frac{\alpha}{2} \sqrt{T} \log(T)$
  - This bound is zero for strongly convex noise

- **Other Algorithms and Results**
  - Feedback that is $\alpha$-informative only in expectation
  - General convex loss functions of $U(y^{*}|x) - U(\mathbf{y}|x)$
  - Regret that scales $\log(T)/T$ instead of $T^{-0.5}$ for strongly convex noise

---

**Preference Perceptron: Experiment**

- **Experiment**
  - Automatically optimize Arxiv.org Fulltext Search

- **Model**
  - Utility of ranking $y$ for query $x$: $U(y|x) = \sum w_i \phi(x,y)$ (~1000 features)
  - Computing argmax ranking: sort by $w_i \phi(x,y)$

- **Feedback**
  - Construct $\mathbf{y}_t$ from $y_t$ by moving clicked links one position higher.
  - Perturbation [Raman et al., 2013]

- **Baseline**
  - Handtuned $w_{bas}$ for $U_{bas}(y|x)$
  - Evaluation
    - Interleaving of ranking from $U(y|x)$ and $U_{bas}(y|x)$

- **Evaluation**
  - Analogous to DCG

- **Summary and Conclusions**
  - $y_t$ dependent on $x_t$ (e.g. ranking for query)
    - Algorithm: Preference Perceptron, $O(\|w\|T^{0.5})$ regret
  - $x_{t+1}$ dependent on $y_t$ (e.g. click given ranking, new query)
    - Algorithm: Preference Perceptron, $O(\|w\|T^{0.5})$ regret

---

**Preference Perceptron:**

- **Definition:** Strict $\alpha$-Informative Feedback
  - Definition: $\alpha$-Informative Feedback

- **Optimal Feedback**
  - Slacks both pos/neg

- **Preference Perceptron:**
  - $\mathbf{y}_t$ that is $\alpha$-informative, the average regret of the Preference Perceptron is bounded by
  $\frac{1}{T} \sum_{t=1}^{T} \mathbb{E}[\mathbf{y}_t] - \frac{\alpha}{2} \sqrt{T} \log(T)$
  - This bound is zero for strongly convex noise

- **Other Algorithms and Results**
  - Feedback that is $\alpha$-informative only in expectation
  - General convex loss functions of $U(y^{*}|x) - U(\mathbf{y}|x)$
  - Regret that scales $\log(T)/T$ instead of $T^{-0.5}$ for strongly convex noise

---

**Summary and Conclusions**

- $y_t$ dependent on $x_t$ (e.g. ranking for query)
  - Algorithm: Preference Perceptron, $O(\|w\|T^{0.5})$ regret

- $x_{t+1}$ dependent on $y_t$ (e.g. click given ranking, new query)
  - Algorithm: Preference Perceptron, $O(\|w\|T^{0.5})$ regret

- **Observed Data ≠ Training Data**
  - Decisions $\rightarrow$ Feedback $\rightarrow$ Learning Algorithm
    - Dueling Bandits
      - Model: Pairwise comparison test $P(y_i > y_j | U(y_i) > U(y_j))$
      - Algorithm: Interleaved Filter 2, $O(\|Y\| \log(T))$ regret
    - Coactive Learning
      - Model: for given $y$, user provides $\mathbf{y}$ with $U(y) > U(y|x)$
      - Algorithm: Preference Perceptron, $O(\|w\| \log(T))$ regret