A Dynamic Bayesian Network Click Model for Web Searching Ranking

OLIVIER CHAPELLE AND YA ZHANG

Presentation by Adith Swaminathan and Deirdre Quillen

Based on Olivier Chapelle's slides from his talk at Microsoft Research, 2009
http://research.microsoft.com/apps/video/dl.aspx?id=105108

CS 6784: Advanced Topics in Machine Learning

- Structured Output Prediction
- Learning with Humans in the Loop
  - Much of the data used for machine learning is the result of observing human behavior (e.g. search engine logs, purchase data, fraud detection). However, it is known that this data is biased (e.g. users can click only on results that are presented). How can one account for these biases during learning? Or how can the learning algorithm deal with these biases by not being a passive observer, but by actively interacting with the human?
- Learning Representations

Specific application: Web search ranking

Specific bias: Clicks depend on presented ranking

Clicks for Web Search Ranking

- Implicit relevance feedback
- Abundant, cheaper than explicit relevance score
- Personalized, democratic, timely
- Can be used as features, or labels

- Difficult to interpret, noisy
- Presentation bias & Quality of Context bias
  - Results at lower positions are less likely to be clicked, even if relevant
  - Clicks depend on the context and quality of other links

Problem Statement

- "What would have been the click-through rate (CTR) of a url if it was the only shown result?"

- Naïve approach (non-answer)
- Position models (widely used)
- Cascade models (more accurate)

- Dynamic Bayesian Network: extends cascade models, can be used in learning to rank

- Question: How would you evaluate solutions to this problem?

Naïve approach

- $CTR(q,u) = \frac{\#\{Clicks\ on\ u\}}{\#\{u\ shown\ in\ Results\}}$
- Position bias!
- $Clicks@Position(q,p) = \frac{\#\{Clicks\ on\ rank\ p\}}{\#\{Results\}}$
- $CTR_{new}(q,u) = \frac{\sum_{\{Clicks\ on\ u\}} 1/Clicks@Position(q,Rank(u))}{\{u\ shown\ in\ Results\}}$
- Need to separate relevance and position bias…

Position Model

- How to control for relevance? Same url at different positions?

- User click ⇔User examined and found it attractive
- $\Pr(u\ at\ rank\ p\ gets\ clicked) =$
  $\Pr(user\ looks\ at\ rank\ p)\ \Pr(user\ clicks\ u|user\ looks\ at\ u)$
- $\Pr(C = 1|u,p) = \alpha_u\beta_p = \Pr(C = 1|u,E = 1)\ \Pr(E = 1|p)$
- Logistic model: $\Pr(C = 1|u,p) = 1/(1 +$

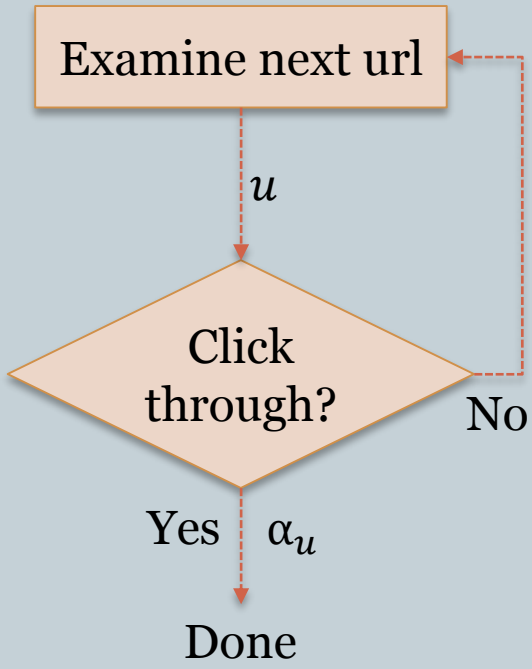
Quality of Context bias?!

- Question: What is the (unrealistic) assumption these models are making?

Cascade Model

Model user decision-making, and design interactive learning systems for the predicted behavior

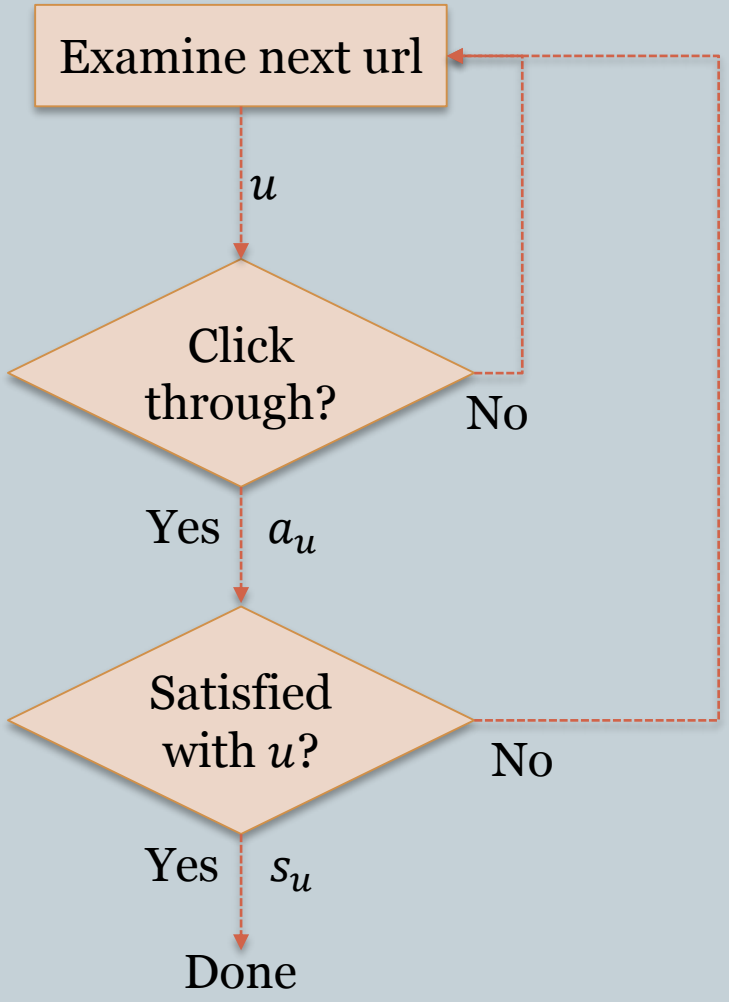
- User scans results list top-down
- Stops when relevant url found
- $\Pr(C = 1|u_3) = (1 - \alpha_1)(1 - \alpha_2)\alpha_3$



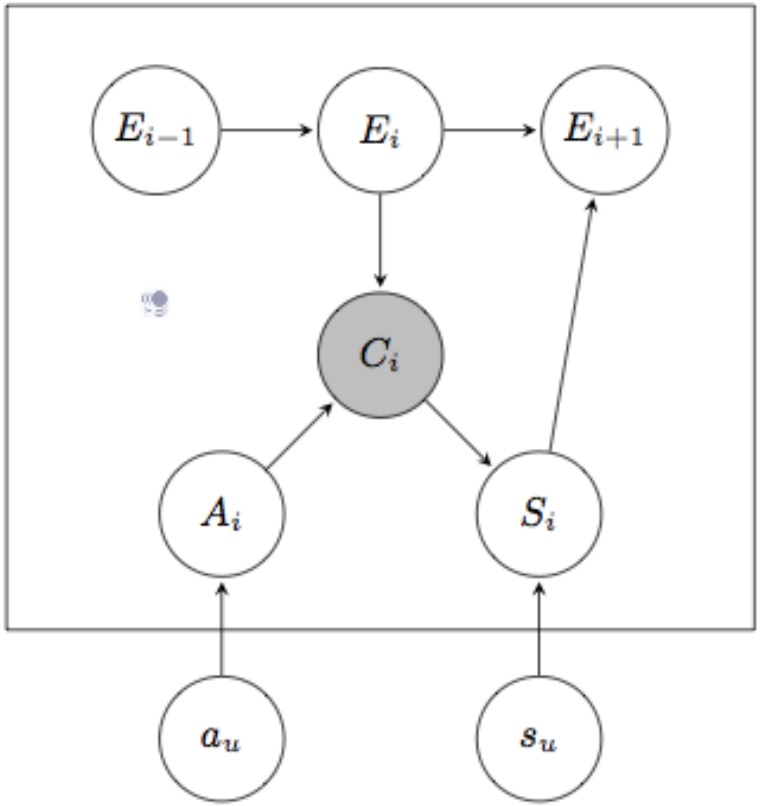
Limitations

- Position Model
  - Click == Relevant
  - Ignores quality-of-context bias
- Cascade Model
  - Exactly one click per session modeled
  - Click == Relevant
  - User never abandons search
- Proposed extension
  - Click $\Longrightarrow$snippet was attractive, but link may be a dud
  - User may abandon search before finding a relevant result
  - Allows 0 and multiple click sessions to be modeled

A more expressive user flowchart



Dynamic Bayesian Network



- E_i: Did the user examine the url at position i?
- A_i: Was the user attracted by the url?
- C_i: Did the user click on it?
- S_i: Was the user satisfied after clicking the url?

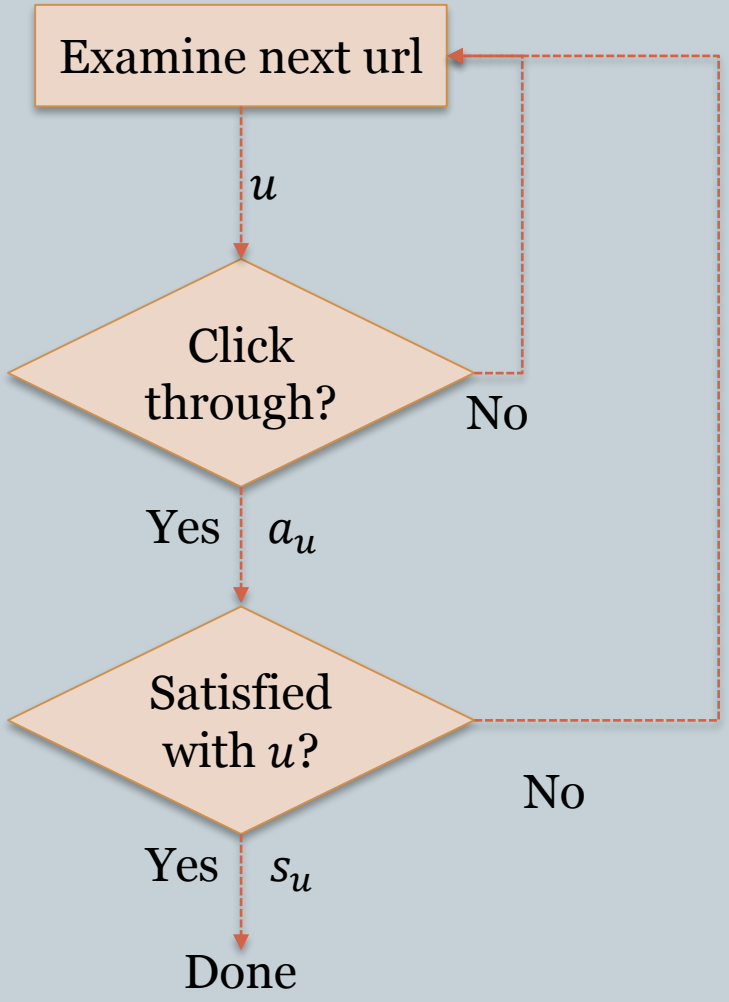
- Url-specific params a_u and s_u are the important hidden variables to estimate
- C_i observed, all else hidden

Assumptions of DBN

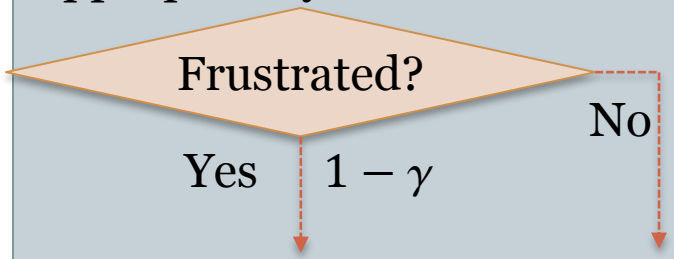
Click on url u at position $i \Leftrightarrow$ user examined u and found it attractive	$A_i = 1, E_i = 1 \Leftrightarrow C_i = 1$
Attraction only depends on the url u *(Perceived relevance)*	$\Pr(A_i = 1) = \alpha_u$
After clicking, user may be satisfied with the page *(Actual relevance)*	$\Pr(S_i = 1 \mid C_i = 1) = s_u$
Not clicking on a url $\Longrightarrow$ user cannot be satisfied with the page	$C_i = 0 \Longrightarrow S_i = 0$
If user is satisfied, done.	$S_i = 1 \Longrightarrow E_{i+1} = 0$
If user is not satisfied, there's a chance they will get frustrated and abandon	$\Pr(E_{i+1} = 0 \mid C_i = 1, S_i = 0) = 1 - \gamma$
If a user stops examining urls at position i, all subsequent positions are left un-examined	$E_i = 0 \Longrightarrow E_{i+1} = 0$

Flowchart Activity

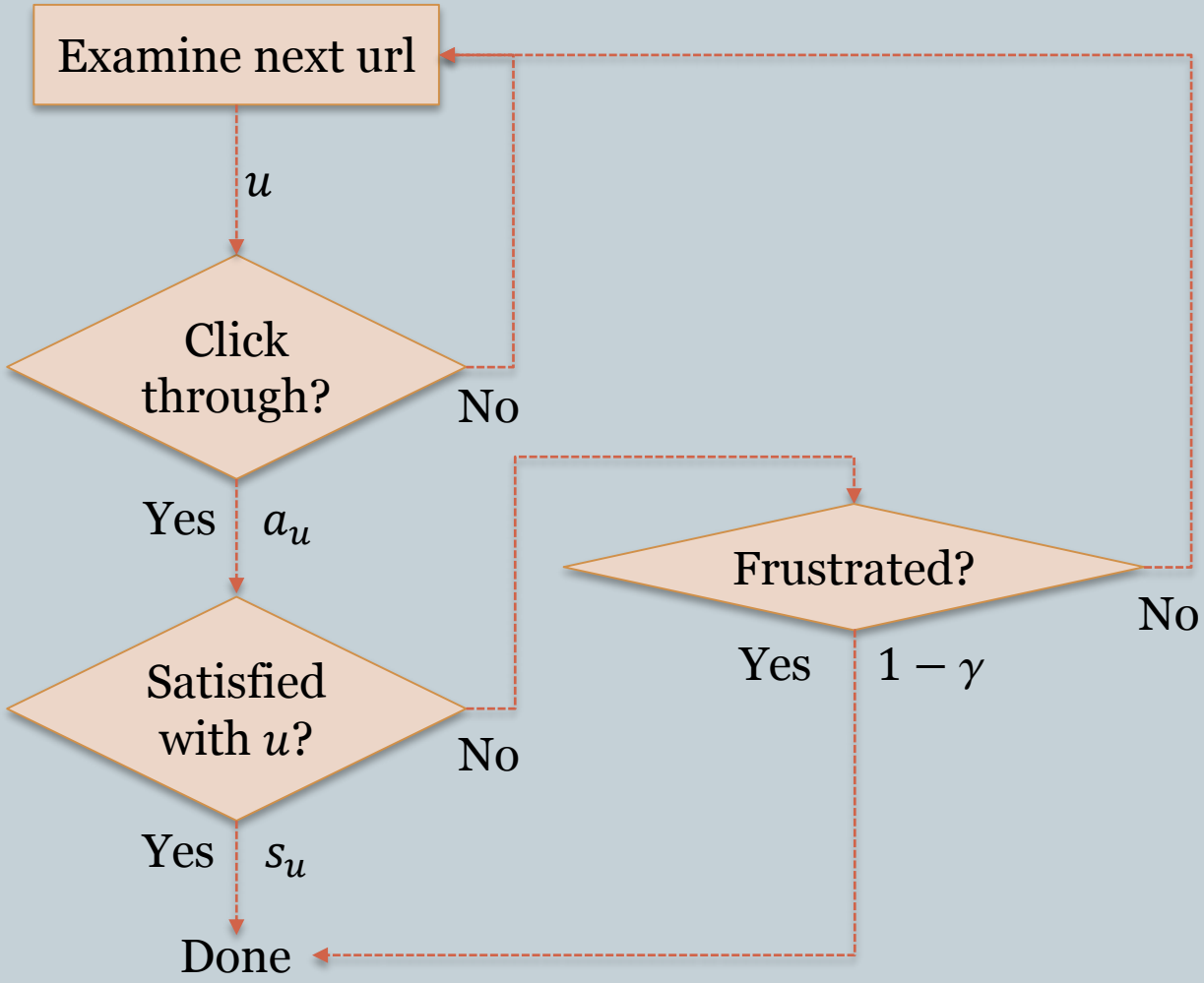
Modify this flowchart to account for users getting frustrated and abandoning the search.



Hint: Insert the following node appropriately:



Flowchart Activity Solution



Relevance estimates from DBN

- Relevance == probability that user is satisfied by url
- $R_u = \Pr(S_i = 1 \mid E_i = 1) = \Pr(S_i = 1 \mid C_i = 1) \Pr(C_i =$

Inference for the Simpler DBN model

- Experimentally, optimal $\gamma = 0.9$
- If $\gamma = 1$, we know that the user was satisfied with the last click.
- Inference by counting

- $s_u =$ $\#\{u \text{ is last click}\}/\#\{\text{Clicks on } u\}$

- $a_u =$ $\#\{\text{Clicks on } u\}/\#\{u \text{ viewed in Result}\}$

- $\{u \text{ viewed in Result}\} =$ $\{u \text{ clicked}\}\ OR\ \{\text{url below } u \text{ clicked}\}$

Algorithm 1 Simplified model estimation for $\gamma = 1$.

Initialize $a_u^N, a_u^D, s_u^N, s_u^D$ to 0 for all urls u associated with current query.
for all sessions **do**
  for all u above or at the last clicked url **do**
    $a_u^D \leftarrow a_u^D + 1$
  end for
  for all u that got clicked **do**
    $a_u^N \leftarrow a_u^N + 1$
    $s_u^D \leftarrow s_u^D + 1$
  end for
  $s_u^N \leftarrow s_u^N + 1$, where u is the last clicked url.
end for// $\alpha_a, \beta_a, \alpha_s, \beta_s$ are prior Beta parameters for a_u and s_u.
for all urls u **do**
  $a_u = (a_u^N + \alpha_a)/(a_u^D + \alpha_a + \beta_a)$.
  $s_u = (s_u^N + \alpha_s)/(s_u^D + \alpha_s + \beta_s)$.
end for

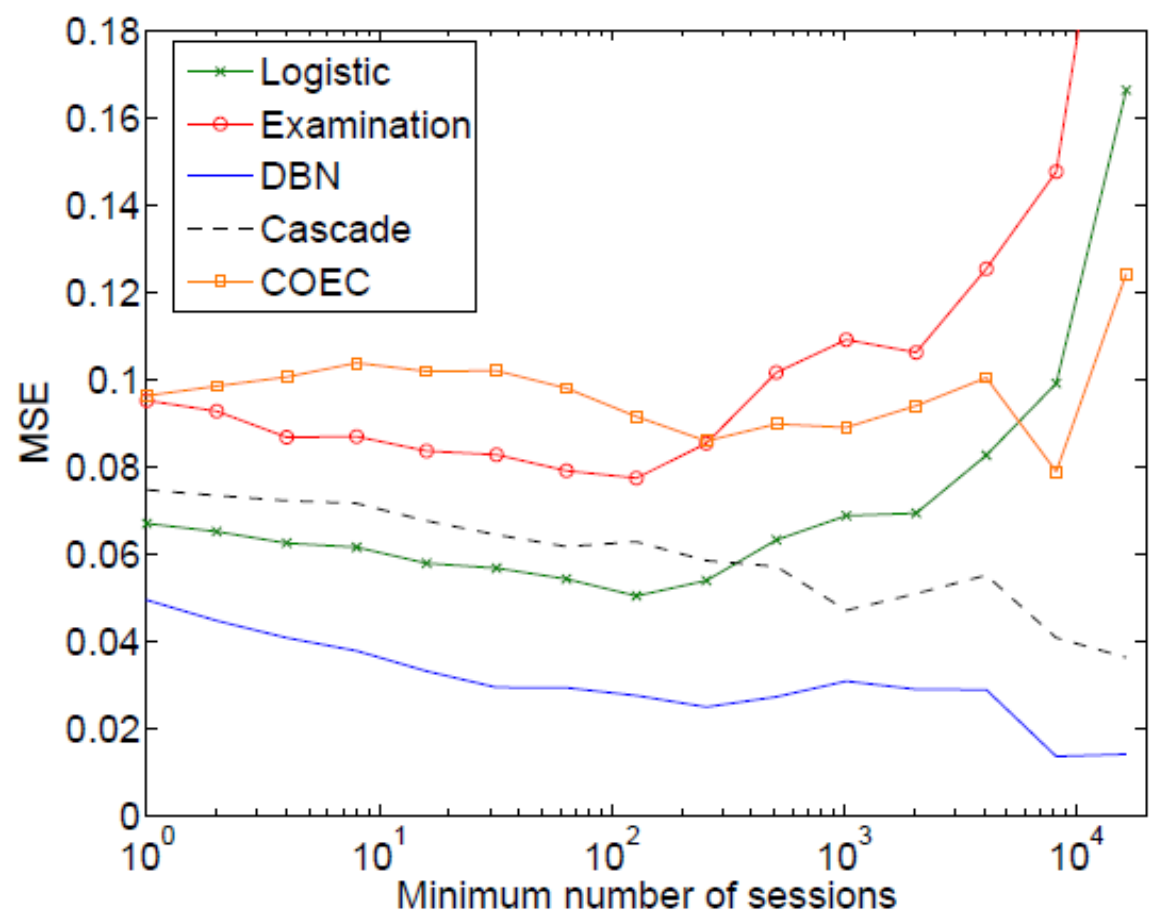
Evaluation

- How accurate is the predicted CTR?
  - $\Longrightarrow$Predict attractiveness, i.e., CTR@1
- How useful is the predicted relevance as a feature for learning to rank?
- Can we use predicted relevance as a proxy label for learning to rank?
- **Experiment setup:**
  - Session == Unique user and query, ended by 60min. idle time
  - Only first page of results
  - Discard sessions where clicks aren't in order of ranking
  - Queries with at least 10 sessions kept (682K queries, 58M sessions)

Predicting CTR@1

- Retrieve all sessions for a query
- Consider a url u that appeared both in position 1 and other positions
- Hold out sessions where the url appears in pos. 1
- Train model on remaining sessions, predict a_u
- Compute observed CTR@1 on the test set $\widehat{a_u}$
- $MSE = (a_u - \widehat{a_u})^2$, average over all such urls and queries, weighted by the number of test sessions

Predicting CTR@1: Results



- DBN > Cascade > Position models
- X-axis = 100 $\Longrightarrow$ those urls whose train set >= 100
- More sessions $\Longrightarrow$ priors not as important. Cascade & DBN improve.
- Navigational queries have quality-of-context bias, and lots of sessions. Position models suffer.

Predicted Relevance as a feature

- Accurate CTR@1 need not directly translate to relevance

- Retrieve all sessions for a query
- Consider all urls with editorial relevance judgments (3153 queries, 44.5M sessions)
- Train model, predict a_u, s_u
- Sort urls according to predicted relevance $a_u s_u$
- Compute $NDCG_5$, average across queries

$NDCG_5$ is the cumulative gain or usefulness of our ranking (measured with editorial judgments) normalized by the cumulative gain of the optimal ranking over the first 5 elements

Predicted Relevance as a feature: Results

Model	$NDCG_5$	Gain
Logistic	0.705	-7.8%
Cascade	0.73	-4.6%
DBN	0.748	-2.2%
DBN – 12 nodes	**0.765**	
DBN –12 ($a_u$only)	0.756	-1.2%
Baseline ϕ	0.795	+3.9%

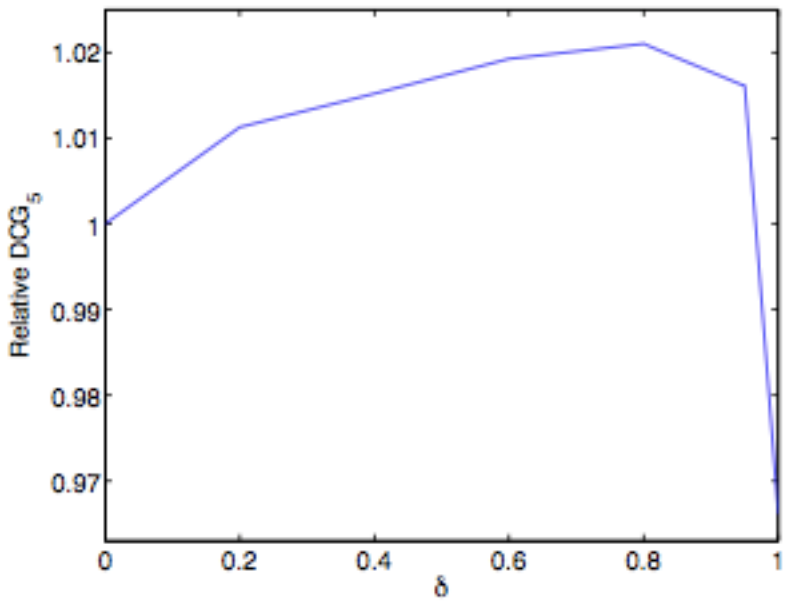
- DBN > Cascade > Position models
- Modeling satisfaction and attraction separately helps
- Modeling user clicks on header (sponsored search) and footer (next page of results) improves results
- Baseline ϕ hand-tuned ranking function used by Yahoo
  - $NDCG_5$ improved 0.8% on Baseline ϕ when $a_u s_u$ added as feature

Predicted Relevance as a label

- How do the DBN predicted relevance compare with editorial relevance judgments?
- Technique: Boosted decision trees trained on pairwise preferences
- Use two kinds of preferences:
  - Preference P_E from editorial judgments (4180 queries, 126K urls, 1M preference pairs)
  - Preference P_C from the DBN model relevance predictions (420K queries, 1.1M urls, 2M preference pairs)
- Learn a ranking function that weights preferences in P_C with δ and P_E with weight $1 - \delta$
- Test on held out set of editorial judgments (DCG_5)

Predicted relevance as a label: Results

- DCG_5 relative to $\delta = 0$
- Left = Only editorial judgments, Right = Only clicks



- Only 4% worse with only click data
- 2% better using both sources of data
- Pessimistic evaluation: even better if using click-based metrics

Limitations and Discussion

- Completely blind to query reformulations
- Unrealistic prior on a_u and s_u
- Assumes homogeneous user population
- Cannot model out-of-order clicks
- Any others?

- Click not necessarily == relevant, models attraction and satisfaction separately

- Good example of the "Interactive Learning System design philosophy" in action: ***Model user decision-making***, and design algorithms to work with predicted behavior