

Computational Advertising and Recommendation

Ye Chen (yec@microsoft.com)

CS294-1, Behavioral Data Mining, Berkeley

Outline

Introduction

- Computational advertising: Definition
- Computational advertising: Landscape
- Recommendation: An alternate view
- A general methodology

Behavioral Targeting (BT)

- Problem definition
- Linear Poisson regression
- Large-scale implementation

Factor Modeling for Ad Targeting

- Problem definition
- The GaP factorization
- The GaP application for sponsored search

Recommendation

- Problem definition
- A latent product model
- Naïve Bayes recommender

Summary

Computational advertising: Definition

- ▶ show the best ads to a user under a context;
- ▶ to optimize some utilities of publishers, advertisers, users, and intermediaries;
- ▶ an emerging subdiscipline that involves:
 - ▶ machine learning: clustering, classification and regression.
 - ▶ optimization: linear, integer, convex optimization.
 - ▶ information retrieval: query-ad selection, learning-to-rank.
 - ▶ economics: game theory, mechanism design, auction theory.
 - ▶ large-scale computing: hadoop, distributed computing.
 - ▶ recommender systems: content and collaborative filtering.

Computational advertising: Sponsored search

- ▶ sponsored search;
 - ▶ context: a user issues a query.
 - ▶ publishers: Google (AdWords), Bing (AdCenter), Yahoo!
 - ▶ max: publisher revenue, s.t. advertiser campaign goal, budget, user satisfaction.
 - ▶ marketplace: keyword-based GSP with cost-per-click (CPC) pricing.
 - ▶ system sketch: query analysis → ad selection and relevance → click prediction → GSP.

The screenshot shows a Bing search results page for the query "data mining job salary". The search results are listed on the left, and sponsored ads are on the right. Two red circles highlight specific areas:

- The first circle highlights the search results section, including the search query, the number of results (1-10 of 149,000,000), and the first few search results: "BI Certification Online", "Job By Salary", "Need a Career Upgrade?", and "Indeed.com Job Search".
- The second circle highlights a sponsored ad for "Easy Data Mining Software" with the text "Analyze data visually & be up & running in minutes - free trial!" and the URL "www.TableauSoftware.com".

Other visible elements include the Bing logo, navigation tabs (Web, Maps), related searches on the left, and search history at the bottom.

Computational advertising: Contextual ads

- ▶ contextual ads: an extension of sponsored search;
 - ▶ context: page content and user behavior.
 - ▶ publishers: content providers, and major search engines operate the marketplace.
 - ▶ system sketch: starts with keyword extraction in absence of user query.

The screenshot shows the Money.com website interface. At the top, there is a green navigation bar with the 'msn MONEY' logo and a search bar labeled 'Search Money' with a 'bing Site Search' button. Below the navigation bar, there are links for 'HOME', 'NEWS', 'INVESTING', 'PERSONAL FINANCE', 'MY MONEY', 'REAL ESTATE', 'CAREERS', 'AUTOS', and 'TAXES'. A secondary navigation bar includes 'saving & budgeting', 'credit & debt', 'loans', 'retirement', 'insurance', 'taxes', and 'your money'. A market data section displays 'DJI 12,949.87 +45.79 +0.35%', 'NASDAQ 2,951.78 unchanged', and 'S&P 1,361.23 +3.19 +0.23%'. Below this, there are 'CALCULATORS' for various financial topics. The main content area features an image of a smiling woman in a red vest and headset, with the headline 'The best, worst of customer support'. The text below the image discusses phone-based customer service. To the right of the main content, there is a 'Sponsored Links' section with three advertisements: '2.50% Mortgage Refinance', 'Open Checking Account', and 'Invest In Huge Savings'. A red circle highlights the 'Sponsored Links' section.

Money | Web

msn MONEY Search Money bing Site Search

HOME NEWS INVESTING PERSONAL FINANCE MY MONEY REAL ESTATE CAREERS AUTOS TAXES

saving & budgeting credit & debt loans retirement insurance taxes your money

Enter a name or symbol GET QUOTE DJI 12,949.87 +45.79 +0.35% NASDAQ 2,951.78 unchanged S&P 1,361.23 +3.19 +0.23% Scottrade: Open an Account U.S. markets closed

CALCULATORS: Car affordability College savings Credit card payoff date Credit scores Debt consolidation Home affordability Home loans Life insurance Retirement

The best, worst of customer support

What do businesses do right – or wrong – when they provide phone-based customer service? Research points to some of the reasons why customers come away either satisfied or frustrated.

America's coming homeless surge

Spring breaks that won't break you

Health care reforms coming in '12

Sponsored Links

2.50% Mortgage Refinance
\$150,000 Mortgage for \$632/Mo. Secure. No Hidden Fees. 3.2% APR
standingover.com

Open Checking Account
Open A Checking Account Online Today. No Hidden Fees.
builtconline.com

Invest In Huge Savings
Half off at top restaurants, spas & more in your area. Try us now!
www.Groupon.com/Sign-Up

Top 5 401K IRA Rollovers
Top 5 401K To IRA Rollovers: Get Up To \$600 When You Rollover Over!
FitzPatrick.com/IRA401K

Computational advertising: Display ads

- ▶ display ads;
 - ▶ context: page, application and user behavior.
 - ▶ publishers: content providers in display ad-networks operated by Google (doubleclick), Microsoft (aquantive), and Yahoo! (rightmedia).
 - ▶ two types of display ads:
 1. reserved: delivery guaranteed, contracts negotiated upfront, pricing based on CPM (cost-per-(k)impression), e.g., brand ads, direct response.
 2. performance-based: max publisher revenue, s.t. advertiser budget, real-time bidding on exchange, pricing based on CPC/CPA (cost-per-action)/CPM.

The screenshot shows the msn MONEY website interface. At the top, there is a search bar labeled "Search Money" with a "bing Site Search" button. Below the search bar is a navigation menu with links for HOME, NEWS, INVESTING, PERSONAL FINANCE, MY MONEY, REAL ESTATE, CAREERS, AUTOS, and TAXES. The main content area displays market data for various indices, including DAX, NASDAQ, and S&P. Below the market data, there are several advertisements for financial services, including E*TRADE, Scottrade, Fidelity, and Ameritrade. A red circle highlights the Scottrade advertisement, which features a 3D pie chart and the text "Do more without spending more. Rollover your IRA. Get \$100.*". Another red circle highlights a section of the page containing "NEWS FOR YOUR RECENT QUOTES" and "RESEARCH A STOCK" sections.

Computational advertising: Emerging formats

- ▶ emerging formats;
 - ▶ social targeting: Facebook, Twitter, and LinkedIn have user profiles and social graphs.
 - ▶ mobile ads: ads in apps, real-time location.
 - ▶ local ads and deals: Groupon.

The image shows a screenshot of a Facebook news feed. The interface includes a top navigation bar with the Facebook logo, a search bar, and user information for 'Ye Chen'. The main feed area contains several posts and advertisements. Two red circles are drawn around specific elements: one around a 'People You May Know' suggestion for 'Ting Liu' (3 mutual friends), and another around a 'Sponsored' advertisement for 'TurboTax Free Edition' featuring a '\$0' graphic. The left sidebar shows navigation options like 'News Feed', 'Messages', 'Events', and 'Find Friends'. The bottom of the page has a navigation bar with various icons.

Recommendation: An alternate view

- ▶ show the best **items** to a **user** under a **context**;
- ▶ a classic problem in e-commerce domain, e.g., Amazon, eBay.
- ▶ some based on explicit feedbacks, e.g., the popular Netflix problem or rating-based recommender.
- ▶ many more rely on implicit feedbacks, e.g., clicks, purchases, dwell-time, social graphs.

The screenshot displays a user interface with three main sections: 'Your last viewed items', 'Your recent searches', and 'Sign in'. The 'Your last viewed items' section features a product image of a 'GRIFFIN POWERJOLT DUAL USB CAR' for \$4.99. The 'Your recent searches' section lists 'iphone car charger', 'iphone charger', and 'tr china'. The 'Sign in' section includes a 'Sign in' button and a 'Register' button. To the right, there is a promotional banner for 'Get Special Financing on Your 1st Purchase of \$200 or more with eBay MasterCard' with an 'Apply Now' button and a 'Valid today through February 29, 2012' notice.

Recommendations for you

The screenshot shows a horizontal carousel of product recommendations for a car charger. The items are: 'NEW! Griffin Powerjolt 3K Dual USB Car Charge...' for \$3.99, 'NEW! Griffin Powerjolt 3K Dual USB Car Charge...' for \$10.00, 'Griffin PowerJolt Dual USB Micro Car' for \$0.99, 'NEW! GRIFFIN POWERJOLT DUAL USB CAR' for \$5.28, 'NEW! Griffin Powerjolt 3K Dual USB Car Charge...' for \$5.99, and 'NEW! Griffin Powerjolt 3K Dual USB Car Charge...' for \$1.09. Each item includes a 'See suggestions' link. The fourth item is highlighted with a 'NEW!' badge.

A general methodology

- ▶ formulate the learning problem;
 - ▶ objective function: min quadratic or hinge loss; max precision/recall, AUC, log-likelihood, clicks, or revenue.
 - ▶ it is critical yet nontrivial to align with real business goals, e.g., revenue, long-term ROI, user engagement.
 - ▶ the gap usually reflects the challenges of non-objectively-measurable, e.g., CTR vs. brand recognition.
- ▶ feature representation for users, ads (advertisers), context (publishers).
 - ▶ mostly counts or categorical.
 - ▶ highly sparse.
 - ▶ very rare positive feedbacks.
 - ▶ right level of granularity vs. concept drifts.
- ▶ solve the optimization problem at large scale (offline) and real-time (online).
- ▶ experiments: offline evaluation (due diligence) vs. online A/B testing (true test).

Behavioral targeting: Problem definition

(Chen, Pavlov, and Canny, KDD'09, TKDD'10)

- ▶ Behavioral targeting (BT)
 - ▶ leverages historical user behavior to select the most relevant ads.
 - ▶ y : predicts and maximizes click-through rate (CTR).
 - ▶ x : ad clicks and views, page views, search queries and clicks.
- ▶ Challenges:
 - ▶ large scale, e.g., Y! logged 9TB ad data with 500B entries on Aug'08.
 - ▶ sparse, e.g., the CTR of automotive display ads is 0.05%.
 - ▶ dynamic, i.e., user behavior changes over time.

Non-negative linear Poisson regression

- ▶ Poisson distribution for counts:

$$p(y) = \frac{\lambda^y \exp(-\lambda)}{y!}, \text{ where } \lambda = \mathbf{w}^\top \mathbf{x}.$$

- ▶ MLE by multiplicative recurrence:

$$w'_j \leftarrow w_j \frac{\sum_i \frac{y_i}{\lambda_i} x_{ij}}{\sum_i x_{ij}}, \text{ where } \lambda_i = \mathbf{w}^\top \mathbf{x}_i.$$

- ▶ CTR prediction:

$$\widehat{\text{CTR}}_{ik} = \frac{\lambda_{ik}^{\text{click}} + \alpha}{\lambda_{ik}^{\text{view}} + \beta}.$$

- ▶ Notation:

y, λ the observed and expected counts.
 \mathbf{w}, \mathbf{x} the weight and bag-of-words feature vector.
 i, j, k the indices of user, feature, and ad category.
 α, β the smoothing constants for clicks and views.

Large-scale implementation: Data reduction and information loss

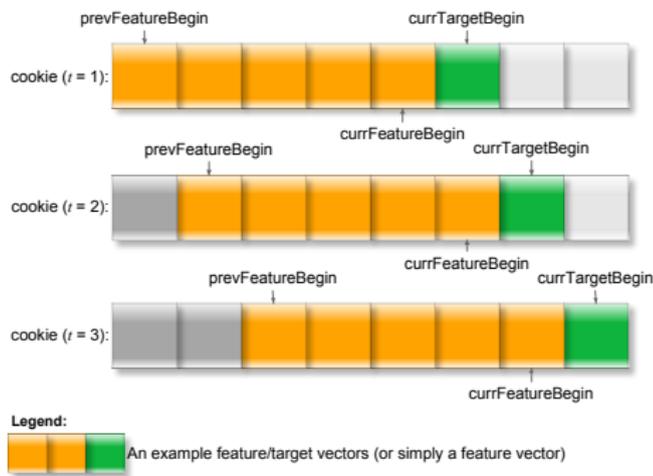
- ▶ Many practical learning algorithms are IO-bound and scan-bound.
- ▶ For BT, one needs to preprocess 20-30TB raw data feeds of ads and searches.
- ▶ Reduce data size at the earliest, by projection, aggregation and merging, e.g., on (cookie, time).
- ▶ Data prep should have minimum information loss and redundancy, e.g., time resolution.
- ▶ Data prep should be loosely coupled with specific modeling logics for data reusability, e.g., neither decays counts nor categorizes ads.
- ▶ After preprocessing, the data size is reduced to 2-3TB.

Large-scale implementation: Feature selection and indexing

- ▶ A data-driven approach is to use granular events as features.
- ▶ Frequency-based feature selection works almost best in practice for sparse data.
- ▶ Frequency is counted in cookie rather than event occurrence (robot filtering).
- ▶ Thresholding immediately after summing Mapper, locally and in-memory, thus cut the long tail of the power-law like sparse data.
- ▶ Output of feature selection is three dictionaries (ads, pages, queries), which collectively define an indexing of the feature space.

Large-scale implementation: Feature vector generation in $O(1n)$

- ▶ Linear time algorithms are of great interest for large-scale learning.
- ▶ The scalar c of a linear complexity $O(cn)$ should be seriously taken into account when n is easily in the order of billion.
- ▶ To generate $D = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ in $O(1n)$ time:



Large-scale implementation: Data-driven weight initialization

- ▶ To exploit the sparseness, one shall use some data-driven approaches.
 1. feature-specific normalization (the idea of tf-idf):

$$w_{kj} \leftarrow \frac{\sum_i \frac{y_{ik} x_{ij}}{\sum_{j'} x_{ij'}}}{\sum_i x_{ij}}.$$

2. target-specific normalization (respect the highly skewed distribution of traffic over categories):

$$w_{kj} \leftarrow \frac{\sum_i (y_{ik} x_{ij}) \sum_i y_{ik}}{\sum_{j'} [\sum_i (y_{ik} x_{ij'}) \sum_i x_{ij'}]}.$$

Large-scale implementation: Parallel multiplicative recurrence

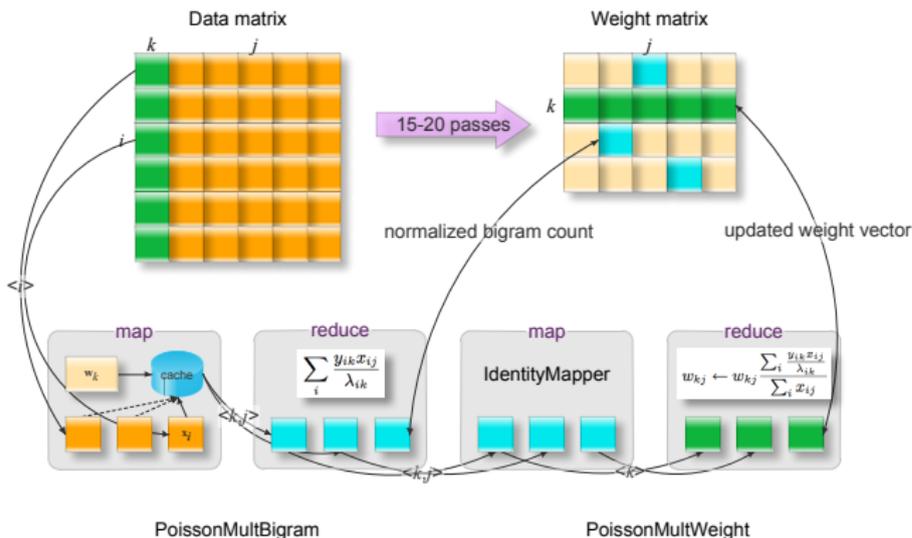
- ▶ Given $D = [Y \ X]$, solve $W^* = \operatorname{argmax}_W \log p(Y^\top | W X^\top)$.
- ▶ An NMF problem $Y^\top \approx W X^\top$ where the quality of factorization is measured by log likelihood.
- ▶ Multiplicative update:

$$w'_j \leftarrow w_j \frac{\sum_i \frac{y_i}{\lambda_i} x_{ij}}{\sum_i x_{ij}}, \text{ where } \lambda_i = \mathbf{w}^\top \mathbf{x}_i.$$

- ▶ Computational bottleneck: $\sum_i \frac{y_i}{\lambda_i} x_{ij}$.
- ▶ Parallel iterative algorithms typically suffer from synchronizing model parameters after each iteration.
- ▶ For BT, the final multiplicative update of \mathbf{w}_k has to be carried out in a single node.

Large-scale implementation: “Fine-grained parallelization”

- ▶ Scalable data structures: $(\mathbf{x}_i, \mathbf{y}_i)$ sparse vectors, \mathbf{w}_k dense vectors.
- ▶ Distribute counting co-occurrences by (k, j) which defines an entry in W .
- ▶ In-memory cache input examples (*not weights*), and retrieve relevant weight vectors on demand.



Legend:

1. Variables: x for feature counts, y for target counts, λ for expected target counts, w for model weights;
2. Indices: i for example, j for feature, k for target;
3. $\langle \text{key} \rangle$: distributing by a single key;

Factor modeling for CTR prediction: Problem definition

(Chen, Kapralov, Pavlov, and Canny, NIPS'09)

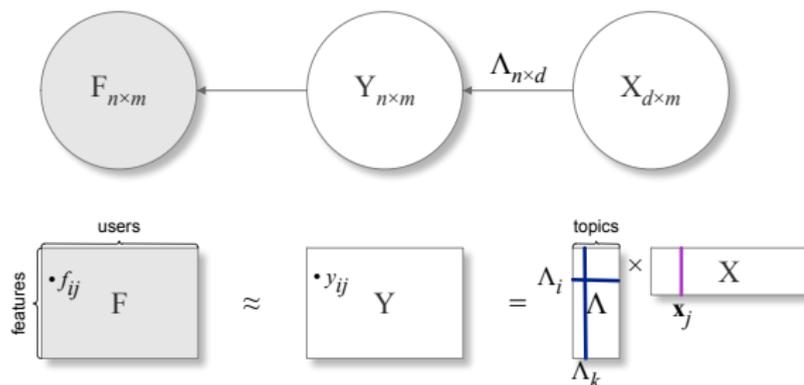
- ▶ Ad targeting $\text{ad}^* = \text{argmax}_{\text{ad}} f(\text{ad}, \text{user}, \mathbf{x})$
 - ▶ To select the ads most relevant to a user.
 - ▶ $y = f(\text{ad}, \text{user}, \mathbf{x})$: typically click-through rate (CTR).
 - ▶ \mathbf{x} : query, page content, user behavior, ad clicks and views.
 - ▶ The count data can be formed as a feature-by-user matrix F .
- ▶ Sponsored search (SS)
 - ▶ To place textual ads alongside algorithmic search results.
 - ▶ $y = p(\text{click}|\text{ad}, \text{user}, \text{query})$.
- ▶ Behavioral targeting (BT)
 - ▶ To select display ads based on historical user behavior.
 - ▶ $y = p(\text{click}|\text{ad}, \text{user}, \text{behavior})$.

The GaP factorization

▶ Notation

- ▶ F is an $n \times m$ matrix of observed counts.
- ▶ Y is an $n \times m$ matrix of expected counts, $F \sim \text{Poisson}(Y)$ element-wise.
- ▶ X is a $d \times m$ matrix where the column \mathbf{x}_j is a low-dimensional representation of user j , i.e., unnormalized $p(k|j)$.
- ▶ Λ is an $n \times d$ matrix where the column Λ_k represents the k th topic as a vector of event probabilities $p(i|k)$, thus $Y = \Lambda X$.

▶ The graphical model



$$f_{ij} \sim \text{Poisson}(y_{ij}) \leftarrow y_{ij} \sim \text{mixture of Multinomial}(p(i|k)) \leftarrow x_{kj} \sim \text{Gamma}(\alpha_k, \beta_k)$$

The generative model

- ▶ To generate an observed event-user count f_{ij} :
 1. Generate $x_{kj} \sim \text{Gamma}(\alpha_k, \beta_k), \forall k$.
 2. Generate y_{ij} occurrences of event i from a mixture of **Multinomial** ($p(i|k)$) with outcome i , i.e., $y_{ij} = \Lambda_i \mathbf{x}_j$ where Λ_i is the i th row vector of Λ .
 3. Generate $f_{ij} \sim \text{Poisson}(y_{ij})$.
- ▶ x_{kj} is given a Gamma as an empirical prior, with pdf

$$p(x) = \frac{x^{\alpha-1} \exp(-x/\beta)}{\beta^\alpha \Gamma(\alpha)} \text{ for } x > 0 \text{ and } \alpha, \beta > 0.$$

- ▶ Given a latent vector \mathbf{x}_j , derive the expected count vector \mathbf{y}_j

$$\mathbf{y}_j = \Lambda \mathbf{x}_j.$$

- ▶ The observed count f_{ij} follows a Poisson with the mean y_{ij}

$$p(f) = \frac{y^f \exp(-y)}{f!} \text{ for } f \geq 0.$$

Parameter estimation

- ▶ The likelihood of a user count vector \mathbf{f}

$$p(\mathbf{f}|\Lambda, \mathbf{x}, \alpha, \beta) = \prod_{i=1}^n \frac{y_i^{f_i} \exp(-y_i)}{f_i!} \prod_{k=1}^d \frac{x_k^{\alpha_k - 1} \exp(-x_k/\beta_k)}{\beta_k^{\alpha_k} \Gamma(\alpha_k)}, \text{ where } y_i = \Lambda_i \mathbf{x}.$$

- ▶ The log likelihood

$$\begin{aligned} \ell = & \sum_i (f_i \log y_i - y_i - \log f_i!) \\ & + \sum_k ((\alpha_k - 1) \log x_k - x_k/\beta_k - \alpha_k \log(\beta_k) - \log \Gamma(\alpha_k)). \end{aligned}$$

- ▶ Given $F = (\mathbf{f}_1, \dots, \mathbf{f}_m)$, we wish to find the MLE of the model parameters (Λ, X) .

$$\text{E-step: } x'_{kj} \leftarrow x_{kj} \frac{\sum_i (f_{ij} \Lambda_{ik} / y_{ij}) + (\alpha_k - 1) / x_{kj}}{\sum_i \Lambda_{ik} + 1 / \beta_k};$$

$$\text{M-step: } \Lambda'_{ik} \leftarrow \Lambda_{ik} \frac{\sum_j (f_{ij} \bar{x}_{kj} / \bar{y}_{ij})}{\sum_j \bar{x}_{kj}}.$$

Rationale for GaP model

- ▶ GaP and LDA are very similar, except for one key difference.
 - ▶ In LDA, the choice of latent factor is made independently word-by-word.
 - ▶ In GaP, several items are chosen from each latent factor, i.e., that topics are locally related.
 - ▶ If x_k are independently distributed $\text{gamma}(\alpha_k, \beta)$ respectively, then the vector $(x_1/s, \dots, x_d/s)$, where $s = \sum_k x_k$, follows a $\text{Dirichlet}(\alpha_1, \dots, \alpha_d)$.
- ▶ Another reason for our preference for GaP is its simplicity.
 - ▶ LDA requires transcendental functions, e.g., the Ψ function in Eq.(8) in (Blei et al., 2003).
 - ▶ GaP requires only basic arithmetic.

Two variants for CTR prediction

- ▶ The standard GaP model fits discrete count data.
- ▶ We derive two variants for predicting CTR.
 1. To predict clicks and views independently, and then to construct the unbiased estimator of CTR, typically with Laplace smoothing:

$$F \approx Y = \Lambda X;$$

$$\widehat{\text{CTR}}_{\text{ad}(i)j} = \frac{\Lambda_{\text{click}(i)} \mathbf{x}_j + \delta}{\Lambda_{\text{view}(i)} \mathbf{x}_j + \eta}.$$

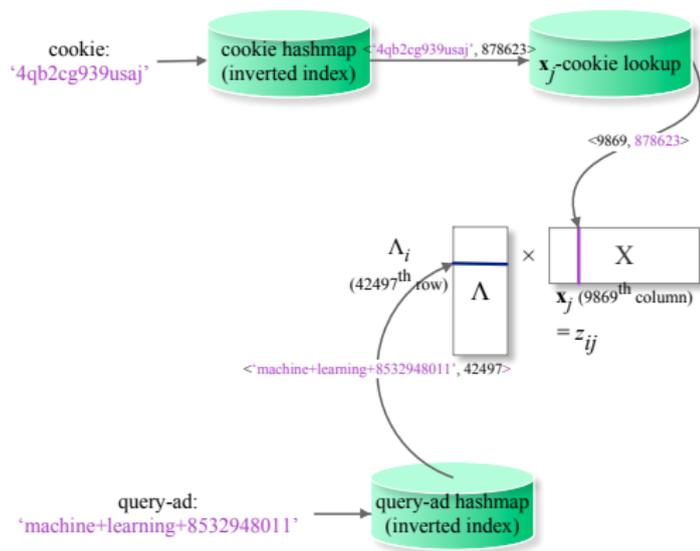
2. To consider the relative frequency of counts in the GaP factorization. Let F be observed clicks, V be observed views, and Z be expected CTRs:

$$F \approx Y = V.Z = V.(\Lambda X);$$

$$\widehat{\text{CTR}}_{\text{ad}(i)j} = z_{ij} = \Lambda_i \mathbf{x}_j.$$

The GaP deployment for sponsored search

- ▶ **Offline training.** Given the observed F and V obtained from a corpus of historical user data, we derive Λ and X using the CTR-based GaP.
- ▶ **Offline user profile updating.** Given the global Λ and the user-local F and V , we update the user profiles X in a distributed and data-local manner, using E-step recurrence only.
- ▶ **Online CTR prediction.** Given a query issued by user j , the global Λ and the local x_j , the predicted CTRs are obtained by a matrix-vector multiplication $z_j^{\text{match}} = \Lambda^{\text{match}} x_j$.



Positional normalization

- ▶ The observed CTR represents a conditional $p(\text{click}|\text{position})$, while we wish to learn a CTR normalized by position $p_{\text{rel}}(\text{click}|\text{examine})$.

- ▶ We assume an examination model with the Markov process:

$$p(\text{click}|\text{position}) = p_{\text{rel}}(\text{click}|\text{examine})p_{\text{pos}}(\text{examine}|\text{position}).$$

- ▶ Apply a GaP factorization with one inner dimension to feature-by-position F and V .
- ▶ Simple and empirically motivated.
 - ▶ Not dependent on the content of ads higher up, as with the cascade or DBN models.
 - ▶ For ads, the probability of clicking any ad link is extremely low.
 - ▶ In this case, the DBN positional prior degrades to a negative exponential function.

Large-scale implementation

- ▶ Recall the multiplicative recurrence

$$\text{E-step: } x'_{kj} \leftarrow x_{kj} \frac{\sum_i (f_{ij} \Lambda_{ik} / z_{ij}) + (\alpha_k - 1) / x_{kj}}{\sum_i v_{ij} \Lambda_{ik} + 1 / \beta_k};$$

$$\text{M-step: } \Lambda'_{ik} \leftarrow \Lambda_{ik} \frac{\sum_j (f_{ij} \bar{x}_{kj} / \bar{z}_{ij})}{\sum_j v_{ij} \bar{x}_{kj}}.$$

- ▶ Data locality
 - ▶ Updating X in a distributed and data-local manner.
 - ▶ Training Λ by alternating 10 successive E-steps with one M-step.
 - ▶ For M-step summing over all users ($f_{ij} \bar{x}_{kj} / \bar{z}_{ij}$ and $v_{ij} \bar{x}_{kj}$) incrementally.
- ▶ Data sparsity
 - ▶ Only computing z when the corresponding f is non-zero (f_{ij} / z_{ij}).
 - ▶ Let N_c be the total number of non-zero f 's, N_v be the total number of non-zero v 's, and r be the number of EM iterations. Typically $N_v \gg N_c \gg m > n \gg d$, the complexity of offline training is $O(N_v dr)$.
- ▶ Scalability
 - ▶ Assuming $O(N_v dr) \approx 4N_v dr$, $m = 10M$, $d = 10$, $N_v = 100 \times m$, $r = 15, \dots, 20$.
 - ▶ We have achieved 100 Mflops by a single-machine implementation with sparse matrix arithmetics.
 - ▶ Thus it takes 1.6-2.2 hours to train a model.

Remarks on scaling

- ▶ We observed that the running time on a 250-node cluster is no less than a single-node yet highly efficient implementation, after accounting for the different factors of users $4\times$ and latent dimension $2\times$, with a similar 100 Mflops.
- ▶ To deal with scaling, implementation issues (such as cache efficiency, number of references, data encapsulation) still cause orders-of-magnitude differences in performance and can more than overwhelm the additional nodes.
- ▶ The right principle of scaling up should start with a single node and achieve above 100 Mflops with sparse arithmetic operations.

Recommendation at long tail: Problem definition

(Chen and Canny, SIGIR'11)

- ▶ Recommendation for online marketplaces (e.g., eBay)
 - ▶ Items are ad-hoc listings, without product or catalog taxonomy.
 - ▶ Transactional data is very sparse (30-fold sparser than the Netflix data).
 - ▶ No user-item ratings, and transactional counts do not follow Gaussian.
- ▶ From items to products
 - ▶ Most items are unique, no links between user behaviors.
 - ▶ Map items to products, a clustering problem.
 - ▶ Topic models are not suitable.
 1. An item has 10 terms, little can be learned by projecting to lower-dim
(e.g., Apple iPhone 4 - 32GB - Black (Unlocked) Smartphone).
 2. Item title terms are highly independent.
 3. The remaining term dependencies are entirely local (e.g., red iPhone?).

Generative clustering

- ▶ An item \mathbf{x} is a 3-tuple of vectors: $\mathbf{x} = (\mathbf{b}, \mathbf{c}, \mathbf{g})$.
 1. For binary variables: $b_v \sim \text{Binom}(p_v), \forall v \in \{1, \dots, V\}$;
 2. For categorical variables: $c_u \sim \text{Mult}(\theta_u), \forall u \in \{1, \dots, U\}$;
 3. For continuous variables: $g_s \sim \mathcal{N}(\mu_s, \sigma^2), \forall s \in \{1, \dots, S\}$.
- ▶ Given a latent product \mathbf{z}_k , the likelihood of an item \mathbf{x}_i is:

$$p(\mathbf{x}_i | \mathbf{z}_k) = \prod_{v: b_{iv}=1} p_{kv} \prod_{v: b_{iv}=0} (1 - p_{kv}) \prod_u \theta_{ku} \\ \times \prod_s \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(g_{is} - \mu_{ks})^2}{2\sigma^2}\right).$$

Parameter estimation

- ▶ Given a set of items $I = \{\mathbf{x}_i\}_{i=1}^n$, we wish to learn a smaller set of latent products $P = \{\mathbf{z}_k\}_{k=1}^m$.

$$(\mathbf{z}_1^*, \dots, \mathbf{z}_m^*) = \operatorname{argmax}_{(\mathbf{z}_1, \dots, \mathbf{z}_m)} \sum_k \sum_i \gamma_{ik} \ell(\mathbf{x}_i | \mathbf{z}_k),$$

where γ_{ik} is an indicator variable for item-product membership.

- ▶ We thus derive the following EM algorithm:

$$\text{E-step: } \gamma_{ik} \leftarrow \begin{cases} 1 & k = \operatorname{argmax}_{k'} \ell(\mathbf{x}_i | \mathbf{z}_{k'}), \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

$$\text{M-step: } \mathbf{z}_k \leftarrow \mathbb{E}(\mathbf{x}_i | \gamma_{ik} = 1). \quad (2)$$

- ▶ Smooth the local parameters \mathbf{z}_k by the background probabilities \mathbf{q} :

$$\mathbf{z}_k \leftarrow (1 - \lambda)\mathbf{z}_k + \lambda\mathbf{q}.$$

Efficient inference

- ▶ The inferential task given a model trained:

$$k^* = \operatorname{argmax}_k \ell(\mathbf{x}' | \mathbf{z}_k).$$

- ▶ Let us define $p_a = (1 - \lambda)p_{kv} + \lambda q_v$ and $p_b = \lambda q_v$,

$$\begin{aligned} \ell(\mathbf{x}_i | \mathbf{z}_k) = & \sum_{v:p_{kv}>0, b_{iv}=1} \log \left(\frac{p_a(1-p_b)}{(1-p_a)p_b} \right) + \sum_{v:p_{kv}>0} \log \left(\frac{1-p_a}{1-p_b} \right) \\ & + \sum_v \log(1-p_b) + \sum_{v:b_{iv}=1} \log \left(\frac{p_b}{1-p_b} \right). \end{aligned}$$

Naïve Bayes: Product-level recommendation model

- ▶ Recommendation can be made by naïve Bayes for ranking:

$$y^* = \operatorname{argmax}_y p(y|x) \propto \operatorname{argmax}_y p(x, y).$$

- ▶ The product-to-product preference probability:

$$p(y|x) = \frac{\alpha_1 C_{yx}^{\text{bb}} + \alpha_2 C_{yx}^{\text{pp}} + \alpha_3 C_{yx}^{\text{cb}} + \zeta}{C_{yx}^{\text{vb}} + \zeta/p(y)},$$

where the baseline popularity:

$$p(y) = \frac{\beta_1 C_y^{\text{p}} + \beta_2 C_y^{\text{b}} + \beta_3 C_y^{\text{c}}}{\max(C_y^{\text{v}}, \varepsilon)}.$$

Here C denotes co-occurrences, e.g., C_{yx}^{vb} is the number of users who bid on x and viewed y . We consider four types of co-occurrence patterns: (1) bid-bid (bb), (2) purchase-purchase (pp), (3) click-bid (cb), and (4) view-bid (vb).

Counting co-occurrences

- ▶ Implement as matrix multiplication, and exploit sparseness:

$$C^{vb} = D_v D_b^T, \forall (x, y) \text{ where } C_{yx}^{cb} > 0.$$

- ▶ Use a $t \times w$ sliding window to count co-occurrences.
- ▶ Impose an empirical positional prior to normalize (multiply) view count:

$$p(r; \eta, \phi) = \frac{1}{Z_1} r^{-\eta}, r = 1, 2, \dots, \phi.$$

Here η is a positive real number implying the rate at which a prior decreases as the positional rank r moves down in a search result page, ϕ is the lowest rank the prior covers, and $Z_1 = \sum_{r=1}^{\phi} r^{-\eta}$ is a normalizing constant.

Item-level ranking model

- ▶ We wish to have a probabilistic scoring function:

$$j^* = \operatorname{argmax}_j p(j|i),$$

where i and j are seed and candidate items, respectively.

- ▶ The item-to-item recommendation score can be factorized as:

$$p(j|i) = \sum_{x,y} p(j|y)p(y|x)p(x|i)$$

where x and y denote latent products, correspondingly.

- ▶ Incorporate auction end time:

$$p(j, \Delta h(j)|i) = p(j|y)p(y|x)p(\Delta h(j)),$$

where $\Delta h(j)$ is the remaining auction time, and $p(\Delta h(j))$ is a smoothed and normalized exponential decay function.

Objective functions

- ▶ Ranking recommendations directly in purchase probability shall optimize number of purchases.
- ▶ But the probabilistic co-preference score can be extended to maximize other utilities.

$$\mathbb{E}(f(u_j)) = f(u_j)p(j, \Delta h(j)|i),$$

where u_j is the unit price of the target item j .

- ▶ To optimize revenue,

$$f_{\text{rev}}(u) = a_1 \min(u, b_1) + a_2 \max(0, (\min(u, b_2) - b_1)) \\ + a_3 \max(0, (u - b_2)).$$

- ▶ To optimize user satisfaction,

$$f_{\text{usr}}(u) = 1 + \log(\max(1, u)).$$

Summary

- ▶ Computational advertising is an emerging scientific subdiscipline that matters substantially to Internet monetization today.
- ▶ Recommendation system is a classic problem, yet becomes active as a formulation for ad targeting, and generally matching items to users.
- ▶ Industrial problems are rich, yet challenging in data scale, sparsity, real-time response time, and temporal dynamics.
- ▶ Often times, the deployment success relies as much on engineering excellence as algorithmic elegance.