

Mining and Querying Multimedia Data

Fan Guo

CMU-CS-11-133

September 21, 2011

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:

Christos Faloutsos, Chair

Eric P. Xing

William W. Cohen

Ambuj K. Singh, University of California at Santa Barbara

*Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy.*

Copyright © 2011 Fan Guo

This research was sponsored by the National Science Foundation under grant numbers DBI-0640543 and IIS-0970179, and a Google Focused Research Award. The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of any sponsoring institution, the U.S. government or any other entity.

Keywords: data mining, graph mining, Web search

To my parents

Abstract

The emerging popularity of multimedia data, as digital representation of text, image, video and countless other milieus, with prodigious volumes and wild diversity, exhibits the phenomenal impact of modern technologies in reforming the way information is accessed, disseminated, digested and retained. This has iteratively ignited the data-driven perspective of research and development, to characterize perspicuous patterns, crystallize informative insights, and realize elevated experience for end-users, where innovations in a spectrum of areas of computer science, including databases, distributed systems, machine learning, vision, speech and natural languages, has been incessantly absorbed and integrated to elicit the extent and efficacy of contemporary and future multimedia applications and solutions.

Under the theme of pattern mining and similarity querying, this manuscript presents a number of pieces of research concerning multimedia data, to address an array of practical tasks encompassing automatic annotation, outlier detection, community discovery, multi-modal retrieval and learning to rank, in their respective contexts including satellite image analysis, internet traffic surveillance, image bioinformatics, and Web search. A repertoire of extant and novel techniques pertaining to graph mining, clustering analysis, tensor decomposition and probabilistic graphical models has been developed or adapted, which satisfactorily met differing quality and efficiency requisites postulated by specific application settings, best exemplified by the 40 times speed-up in annotating satellite images and the up to 30% performance improvement in predicting web search user clicks, yet without the loss of generality to similar and related scenarios.

Acknowledgments

This manuscript would never be completed without the inspiration, support, trust and humor from my advisor, Christos Faloutsos, to whom I owe the greatest gratitude for the gaiety of graduate study. I'd like to thank Professor Eric Xing for his mentorship that matured my minds on research, Professor William Cohen for his careful review of the thesis and proposal drafts, and Professor Ambuj Singh for his insightful questions and comments.

I'm honored to be a member of the database group, with Jia-Yu Pan, Ippokratis Pandis, Jimeng Sun, Debabrata Dash, Jure Leskovec, Hanghang Tong, Mary McGlohon, Lei Li, Leman Akoglu, Polo Chau, B. Aditya Prakash, U Kang, and Danae Koutra, and enjoying the collaboration with our visitors and collaborators residing on several continents – Michael Buckland, Robson Cordeiro, Tina Eliassi-Rad, Donna Haverkamp, James Horne, Ellen Hughes, Gunhee Kim, Sang-Wook Kim, Lewis Lancaster, David Lo, Koji Maruhashi, Marcela Xavier Ribeiro, Pedro Stancioli, and Tim Tangherlini. Sincere acknowledgements to Edo Airoidi, Wenjie Fu, Lie Gu, Steve Hanneke, Tien-ho Lin, Yan Liu, Pradipta Ray, Yanxin Shi, Kyung-Ah Sohn, Rong Yan for their advise and help earlier in my research endeavors, to Indrayana Rustandi, Yi-Fen Huang and Lei Li for being TA together, to Hui Zhang for being my faculty contact, to Yong Lu for being my student contact, and to Mor Harchol-Balter for the warm words concluding the last Black Friday letter.

The days in Wean Hall and Gates Center would be less fun without another group of my favorite members in the school, most of whom have been here much longer than any student. I could still recall the morning when Sophie Park walked me around the 4200 corridor in Wean, the many check-in's with Sharon Burks and Debbie Cavlovich, an army of packages signed out from Marsey Mauer, Marcella Baker and Gayle Bishop, hilarious emails and relaxing chats with Catherine Copetas, wandering around the 8th floor in Gates to greet Sharon Cavlovich, Michelle Martin, Diane Stidle, Marilyn Walgora and Charlotte Yano, Jim Park sitting after the help desk, Martha Clarke typing in admissions meetings, Karen Lindenfelser speaking in the PDL retreat, picking up hardware from Royal Harvard, and quite a few phone calls to the payroll office made by Karen Olack. A special acknowledgement to my OIE friends, for the foreign-student advisory, and the birthday song that shined through the most rainy April in the city.

The four summers spent in the West Coast are among the most colorful and fruitful chapters of my life as a graduate student. I have been very fortunate to be working with my awesome mentors Suju Rajan, Chao Liu, Monica Rogati, and Yanxin Shi, sharing the time together with my fellow interns Wenjie Fu, Dragomir Yankov, Hong Chen, Yuzhe Jin, Ziqing Mao, Ying-Yi Liang, and Duo Zhang, while receiving a great deal of help also from Zheng Shao, Siying Dong, Anmol Bhasin, Tido Carriero, and Xiaoliang Wei.

Next is a list of friends who had been around since the most challenging season of my stay in Pittsburgh, yet their relentless departure from Carnegie Mellon had made the life of an N-th year student more miserable. Thanks to –

- Yanxin Shi for “refreshing the spirit”, and the MSN message log extending well beyond a dozen megabytes.
- Yi Wu for answering numerous “zen-me-ban-a” questions, and co-authoring part of this acknowledgement.
- Runting Shi for “na-shi”, green tea Frappuccino, DDR demo, and ping-pong games.
- Xi Liu for being my advisor in driving. I should have learned swimming together with you!
- Wenjie Fu for being the first “shi-di”, and the most referenced “big-brother”.
- Kyung-Ah Sohn for the Korean-style dinner treats and the small gift. All the best!
- Vahe Poladian for the numerous “bu-dao” with topics ranging from printer configuration to thesis preparation, from pairs trading to presidential election.
- Monica Rogati and Lucian Lita for strawberry picking, and the lunch from “sheng-jian-bao” – your baby is lovely!
- Minglong Shao for Pretty Good Race and Random Distance Run.
- Jingrui He and Hanghang Tong for being on the same flights.
- Wei Guo for the pick up from the PIT airport on August 9, 2005.
- Lie Gu, Zhenzhen Kou, Han Liu, Mengqiu Wang, Xiaofang Wang, Chenyu Wu, Chuang Wu, Hong Yan, Jun Yang, Yimeng Zhang, Le Zhao, Yi Zhou . . .

And thanks to Ling Li for the multiple times of help while I explored and attempted detours from this long path.

Oversea conference travels aggregate to a unique movement in the melody of my grad-school life; I’m obliged to Robson Cordeiro, Lei Li, Chao Liu, Koji Maruhashi and Christos Faloutsos for making them happen. Besides exploring the unknown, it was always a delight to re-unite with friends on the other side of the Pacific Ocean, who have brought me enjoyment on different occasions with rejoice and peace during these trips. A “Thank-You” to Shiyu Xie, Xin Xin and Bingjun Zhang.

The figure next page features a data-driven approach with a certain degree of randomness to render my appreciation to many other friends.

I’m thankful to my high school classmates Kai Chen, Yongtao Cui, Fei Liu, Jin Su, Chun Wang, and our very-well-respected teacher Guoyong Chen, for being friends for more than 13 years.

With inexpressible gratefulness to Wenxi Cai (cc), Kun Xu (crazyboy), Jian Ying (out_star), and Juntao Ouyang (outfox).

Contents

I	Introduction	1
1	Introduction	3
1.1	Motivation	3
1.2	Mining Multimedia Data	3
1.3	Querying Multimedia Data	4
1.4	Thesis Organization	6
II	Mining Multimedia Data	9
2	<i>QMAS: Querying, Mining And Summarization of Multi-modal Databases</i>	11
2.1	Overview	11
2.2	Related Work	13
2.2.1	Image Labeling	13
2.2.2	Clustering	14
2.2.3	Feature Extraction	14
2.3	Proposed Method	15
2.3.1	Feature extraction	15
2.3.2	Mining and Attention Routing	16
2.3.3	Low-labor Labeling	17
2.4	Experimental Results	19
2.4.1	Experimental Setup	19
2.4.2	Computational Time	20
2.4.3	Non-labor Intensive Labeling	21
2.4.4	Finding Representatives and Outliers	22
2.4.5	Query-by-Example Experiments	24
2.5	Conclusion	24
3	<i>MultiAspectForensics: Pattern Mining on Large-scale Heterogeneous Networks with Tensor Analysis</i>	25
3.1	Introduction	26
3.2	Proposed Algorithm	27

3.2.1	Data Decomposition	27
3.2.2	Spike Detection in Histograms	28
3.2.3	Substructure Discovery	29
3.3	Empirical Results	33
3.3.1	Data and Environment	33
3.3.2	LBNL Traffic Log	34
3.3.3	RTW Knowledge Base	34
3.3.4	BDGP Gene Annotation	35
3.4	Related Work	36
3.4.1	Anomaly Detection	36
3.4.2	Tensor Analysis and Graph Mining	37
3.5	Conclusion	37

III Querying Multimedia Data 39

4 CDEM: Flexible Querying System for Biological Image Databases 41

4.1	Background	41
4.2	Proposed Method	43
4.2.1	Graph Construction	43
4.2.2	Multi-Modal Retrieval	43
4.3	Experimental Evaluation	44
4.4	Related Work	47
4.4.1	Automatic Analysis of Embryo Images	47
4.4.2	Online Databases	47
4.5	Conclusion	48

5 BEFH: Bayesian Exponential Family Harmoniums for Multimedia Retrieval 49

5.1	Introduction	49
5.2	Bayesian EFH	51
5.3	Model Inference and Estimation	55
5.3.1	Algorithm Overview	55
5.3.2	Approximation schemes	56
5.3.3	Approximating the expectations with brief sampling	58
5.3.4	Computing the predictive posterior density	58
5.3.5	Hyperparameter Estimation	58
5.4	Experimental Evaluation	59
5.4.1	Synthetic EFH parameter estimation	59
5.4.2	Classification of Text and Image Data	62
5.5	Conclusion	64

6	Click Models: Leveraging User Feedback for Better Search Experiences	65
6.1	Background	65
6.1.1	Click Position-Bias	66
6.1.2	Basic Hypotheses	66
6.2	Proposed Method	68
6.3	Experimental Evaluation	70
6.3.1	Experimental Setup	71
6.3.2	Evaluation based on Log-Likelihood	72
6.3.3	Evaluation based on Position-Bias Plots	72
6.3.4	Predicting First and Last Clicks	73
6.4	Related Work	78
6.4.1	Click Log Analysis and Learning to Rank	78
6.4.2	User Behavior Study and Click Models	78
6.5	Conclusion and Discussion	79
IV	Conclusion	81
7	Concluding Remarks	83
7.1	Mining Multimedia Data	83
7.2	Querying Multimedia Data	84
7.3	Discussion and Future Directions	85
V	Appendix	87
A	Click Chain Model	89
A.1	Introduction	89
A.2	Background	89
A.3	Click Chain Model	90
A.4	Algorithms	93
A.4.1	Deriving the Conditional Probabilities	93
A.4.2	Computing the Posterior	96
A.4.3	Parameter Estimation	98
A.5	Performance Evaluation	99
A.5.1	Experimental Setup	100
A.5.2	Results on Log-Likelihood	100
A.5.3	Results on Click Perplexity	101
A.5.4	Last-Click Prediction	102
A.5.5	Position-bias of Examination and Click	103
A.6	Conclusion	105
B	RTW Knowledge Base Query Interface	107

List of Figures

1	THANK YOU	ix
1.1	Examples of query interfaces which (a) either explicitly solicits input, (b) or adopts a light-weight manner to infer users’ preferences based on their profiles and impresses results devoid of typing.	5
1.2	A graphical representation for user-page recommendation.	5
1.3	A summary of data sets included in this manuscript.	7
2.1	An illustration of labeling results from the proposed algorithm. Left: the input satellite image of the city of Annapolis, divided in 1,024 (32x32) tiles, only four of which are labeled. Right: suggested labels from <i>QMAS</i> ; yellow indicates outliers which are likely to represent “Bridge”.	12
2.2	Pre-processing applied to multi-band satellite imagery. Best viewed in colors. Left: sample input multi-band image; Right: the resulting 5-band composite image for which features are computed.	15
2.3	GBT structure illustrated. Left: two levels of GBT cells with 343 pixels (Level-3, outlined in white) and 2,401 pixels (Level-4, outlined in red) overlaid on an image. Right: output values were assigned according to the variance of the adjacent lower level of cells (at Level-2, consisting of 49 pixels each). Bright areas have greater variance, dark areas have less.	16
2.4	The knowledge graph for a toy input set. Nodes shaped as squares, circles, and triangles represent images, labels, and clusters respectively.	19
2.5	Graph construction time versus size of the subsets randomly sampled from <i>SATI.5GB</i> . <i>QMAS</i> : red circles; GCap: blue crosses; GCap-ANN: green diamonds. Timing results are averaged over 10 runs; error bars are too tiny to be discerned.	20
2.6	Labeling accuracy versus the number of pre-labeled examples for each labeling class. <i>QMAS</i> : red circles; GCap-ANN: green diamonds. Accuracy values of <i>QMAS</i> are barely affected by the size of the pre-labeled data. Box plots summarize 10 runs over randomly selected inputs, with outliers (typically over 3 standard deviations from the mean) indicated by red circles and green diamonds.	21
2.7	Representatives for the <i>GeoEye</i> dataset, each colored according to the cluster membership.	22
2.8	Top-3 outliers for the <i>GeoEye</i> dataset.	22

2.9	Example “Water”: Labeled Data and Results of Water Query.	23
2.10	Example “Boats”: Labeled Data and Results of Boat Query.	23
3.1	Illustration of the CP decomposition: the input 3-mode tensor on the left is decomposed into R triplets of vectors on the right, reminiscing of the rank- R singular value decomposition of a matrix. The three modes, in a scenario of network traffic analysis, may represent source IP address (red), destination IP address (blue) and port number (green), respectively.	27
3.2	An attribute plot which displays absolute values of eigenscores (y-axis in log-scale) along its elements (indexed by the x-axis). The arrow on the right points to a common score value, illustrating an observation critical to the algorithmic design of <i>MultiAspectForensics</i>	29
3.3	An attribute plot (adopted from Figure 3.2) on the left side-by-side with the corresponding histogram plot with spikes detected indicated by circles.	30
3.4	Examples of generalized star patterns discovered in the LBNL (Lawrence Berkeley National Lab) network traffic data set. Wavy arrows indicate multiple edges between the pair of nodes with a handful of distinct attribute values. (a) 10 source IP addresses (randomly selected out of 172 ones) are sending multiple packets to a server machine with Port# 534, which is a UDP port under the NCP protocol from a network OS for file sharing and printing services; (b) The source IP registered by an Indian ISP is sending packets to a host in LBNL via port numbers (ranging from 2,300 to 2,900) not usually intended for this type of communication, implying a suspicious activity.	31
3.5	Examples of generalized bipartite-core patterns discovered in the LBNL (Lawrence Berkeley National Lab) network traffic data set. Wavy arrows indicate multiple edges between the pair of nodes with a handful of distinct attribute values. (a) 10 source IP addresses (randomly selected out of 119 ones) are sending multiple packets to an array of server machines, including server 131.243.141.187, which also appears in Figure 3.4 as part of a generalized star pattern, over a port used for file sharing and printing services; (b) 10 source IP addresses (randomly selected out of 63 ones) are sending packets over different ports to a multi-purpose server machine.	32
3.6	Two generalized star patterns discovered from the RTW knowledge base: (a) Music artists specialized in punk or one of its sub-genres according to the knowledge base; (b) European destinations of the Ryanair, an Irish low-cost airline.	35
3.7	An attribute plot on the left side-by-side with the corresponding histogram plot for the “gene” mode of the BDGP data set. The largest spike that appeared at the bottom is the set of <i>maternal genes</i> , a special class of genes that play a vital role in early embryo development such as the polarity of the egg, <i>i.e.</i> , which part will become the head and which other part turns into the tail later.	36

4.1	A fruit fly embryo image with all its attributes sampled from the BDGP database. The original filename is <code>insitu65954.jpeg</code>	42
4.2	A tri-partite graph constructed from the BDGP database.	43
4.3	Running time based on different developmental stage ranges (a) for constructing the graphical representation, averaged over 10 repetitions; and (b) for one query using RWR, averaged over 100 random queries, with error bars of one standard deviation.	45
4.4	A typical query result using an embryo image in (a) as the query input. Top 4 similar images other than the query image itself are displayed in (b)-(e).	46
4.5	C-DEM: an online, multi-modal query system for Drosophila embryo databases. Images are adapted from [48].	46
5.1	The graphical model representation for a harmonium with 3 input units (<i>e.g.</i> , binary word occurrences in a document) and 2 hidden units (<i>e.g.</i> , projection in a 2-dim topic space).	53
5.2	A comparison of EFH, LDA and BEFH models over a single document. Circles represent variables, and diamond represents model parameters. (a) EFH. For easy comparison, the hidden unit (<i>i.e.</i> , the topic weight coefficients) $\{H_j\}$ and the input units $\{X_i\}$ are represented as vector valued variables H and X , respectively. For simplicity, only the \mathbf{W} parameter of EFH is explicitly shown. (b) LDA. Note the correspondence between π in LDA and H in EFH, and the fact that β_j 's are random variables rather than parameters. I denotes the length of the document. (c) BEFH. Note that $\mathbf{W} \equiv \{W_j\}$ are now lifted as random variables.	54
5.3	Details of Monte Carlo simulations of the Langevin algorithm, with y -axis corresponds to the value of W_{11} . Three chains of different starting points are shown. The burn-in time to reach convergence is approximately 50 transition.	60
5.4	The estimation versus the number of sampling steps in brief sampling (solid line) compared with the estimation perfect sampling (dash line), with y -axis corresponds to an estimated derivative of log-partition function $\partial \log Z(\mathbf{W})/\partial W_{11}$ averaged over 50 runs. Both sampling schemes generate 100 samples in each run. The standard error bars are scaled by 1.64, indicating 95% significance of the difference in estimation. A single sampling step suffices as it maximizes the program efficiency without increasing bias or variance of the estimation.	61
5.5	The Performance of ML learning and Bayesian inference using the brief Langevin algorithm under two different error measures (a) mean absolute error; (b) mean relative error. The results are averaged over 10 runs; the error bar may be too small to be distinguishable from the figure. The Bayesian approach is subject to less error rate than its ML alternative in both measures.	62

5.6	Histogram of 100 estimations of partition function using a naïve Monte Carlo approximation on (a) synthetic dataset; (b) real-world dataset. Arrows are centered at the mean and indicate an interval of length of 2 times the standard deviation. Each estimation computes the expectation using 1000 samples. On the real-world data set, the estimation is subject to unacceptably high variance. . . .	63
5.7	Classification accuracy versus number of latent topics. Bayesian DWH yields comparable performance to the original DWH approach. Both produce better results over the baseline LSI approach and the GM-LDA approach backed by a directed graphical model.	63
6.1	Comparison of user attention (fixation) and clicks over top 10 ranks between the normal order and the reversed order reveals position bias. Plots are extracted from Figure 4 in [62].	67
6.2	The user model of DCM, in which r_{d_i} is the document relevance of d_i , and λ_i is the user behavior parameter for position i	69
6.3	Log-likelihood per query session on the test data for different query frequencies. The overall average for DCM is -1.327, compared with -1.401 for ICM which reflects a 7% improvement.	73
6.4	Click probabilities for different positions summarized from ICM/DCM samples as well as test data, and examine probabilities implied by DCM. The click distribution implied by DCM matches the ground truth closely.	74
6.5	Examine probabilities implied by DCM for different query frequencies. Queries are grouped into 6 frequency ranges similarly as in Table 6.1. Darker and lower curves correspond to more frequent queries.	75
6.6	Root-mean-square (RMS) errors for predicting (a) first clicked position and (b) last clicked position. Results are averaged over 100 samples per query session.	76
6.7	First click distribution (a) and last click distribution (b) obtained by drawing samples from DCM and ICM given document impression. The overall first/last click distribution of DCM samples matches the empirical distribution in the test set very well, particularly for top 5 positions. Results are averaged over 100 samples per query session.	77
A.1	Our proposed user model of CCM, in which R_i is the relevance variable of d_i at position i , and α 's form the set of user behavior parameters.	91
A.2	The graphical model representation of CCM. Shaded nodes are observed click variables.	92
A.3	Different cases for computing $P(C R_i)$ up to a constant where l indicates the last clicked position. Darker nodes in the figure above represent clicks at these positions, whereas lighter nodes represent skips.	94
A.4	Log-likelihood per query session on test data for different query frequencies. The overall average for CCM is -1.171, 9.7% better than UBM (-1.264) and 14% better than DCM (-1.302).	101

A.5	Perplexity results on test data for (a) for different query frequencies or (b) different positions. Average click perplexity over all positions for CCM is 1.1479, 6.2% improvement over UBM (1.1577) and 7.0% over DCM (1.1590).	102
A.6	Root-mean-square (RMS) errors for predicting the last clicked position. Prediction is based on the SIMulation for solid curves and EXPection for dashed curves.	103
A.7	Examination and click probability distributions over the top 10 positions.	104
B.1	A snapshot of the online query interface.	109
B.2	Illustration of user actions supported by the query interface.	109
B.3	A graphical illustration of the interface.	110

List of Tables

2.1	Summary of Acronyms	13
3.1	A Summary of Data Sets	33
4.1	A summary of graphs constructed of different sizes.	44
4.2	C-DEM query results using the query image shown in Figure 4.4(a).	45
5.1	Summary of Acronyms	49
5.2	Symbol table	52
6.1	Summary of Test Set	71

Part I

Introduction

Chapter 1

Introduction

1.1 Motivation

The vast and sprawling collection of digital multimedia data, on the crescendo of the Internet era, has manifested unprecedented bandwidth and efficacy of information production and transmission, engendering a profound enrichment of our everyday experience. Its proliferating ubiquity could be aptly illustrated by the overwhelming popularity of smart phones, with increasingly powerful capacity and elegant design for Web browsing, image/video recording and playback, location-based services, to name a few, as well as an almost omnipresent social layer upon them, thereby generating and exchanging data flows in a miscellany of modes that extend far beyond those from call making and text messages only a couple of years ago.

Web search, from which the Internet giant Google sprouted and thrived in the previous decade, serves as another vivid example, with most lustrous breakthroughs in this decade to be much likely towards *mobile search*, which renders mobile content more “accessible and useful”, plus *social search*, which “brings friend effect to search”, and also *universal search*, which blends regular Web results with a variety of “verticals” such as image, videos, news articles, microblogging feeds and quite a few others.

These innovations, with gigantic and usually unwieldy multimedia data flows, entail a demand of, and would unexcludingly benefit from, novel computational approaches to navigate and explore the space of information mapped from multi-aspect, and typically structured records. To satisfy the hunger of such algorithmic and heuristic tools, this manuscript presents a set of studies in an attempt to yield knowledge, information and insight into multimedia data from two perspectives – *mining* and *querying*.

1.2 Mining Multimedia Data

Instead of making a pronouncement on the definition of multimedia mining, it might be more informative to commence with a sketch through concrete examples to be covered at length in relevant chapters of the thesis:

Satellite Imagery – Given a collection of high-resolution satellite map images spanning several gigabytes, each of which is divided into small rectangular or hexagonal tiles, with a few of them labeled *a priori* by domain experts using a controlled vocabulary, how to, in an efficient way, suggest labels for all remaining tiles, propose refinements of the labeling vocabulary to better distinguish these tiles, and spot “outlier tiles” that are dissimilar with most others?

Network Traffic Log – As part of a network measurement study, packet traces sent from or received by a several-thousand-host enterprise network during an over-100-hour time span were collected to generate millions of records which log, for each packet, its source, destination, communication port, and time stamp. The task of interest is to automatically and swiftly detect possibly suspicious activities to be reported for further investigation.

Web Knowledge Base – This is produced by an automated system which constantly “reads the web” to extract facts such as (Facebook, HeadquarteredIn, Palo_Alto). Is it possible to distill some knowledge out of these simple and flat facts? Is it possible to further leverage these facts to compose intelligent answers to questions like “tell me something interesting about George Harrison”?

Biological Image Collection – To study the early development process of fruit fly, a total of more than 70,000 embryo images were captured documenting the spatio-temporal expression activities of selected genes during the first 24 hours after fertilization. An ambitious goal is to identify groups of genes that exhibit correlated patterns over time and visualize such relationship *in silico* to assist further scientific verification and discovery.

Each example presented precedingly illuminates a scenario relevant to pattern mining for multimedia data. As diverse as these subjects may be, they do share a single feature: *data-driven problem solving over multiple modes at a non-trivial scale*. And this becomes the working definition of multimedia mining in this monograph, based on which we will strive to understand the data available and the goals set in each context. How was the data set collected, or, put another way, how was the measurement taken for each aspect of the data? What are the connections amongst different data modes, as well as attributes within the same mode? How to transform one or more modes of the data into a compact and viable representation, if at all necessary? How to craft algorithms and heuristics accordingly that attain appropriate trade-off between quality and computational complexity? How to design numerical experiments to evaluate proposed methods with confidence? The second part of the dissertation will be devoted to address the forerunning list of questions.

1.3 Querying Multimedia Data

A substantial part of the information seeking behavior, especially from Internet users, is pertinent to similarity querying, which facilitates the exploration to broaden their scope of knowledge, and



Figure 1.1: Examples of query interfaces which (a) either explicitly solicits input, (b) or adopts a light-weight manner to infer users' preferences based on their profiles and impresses results devoid of typing.

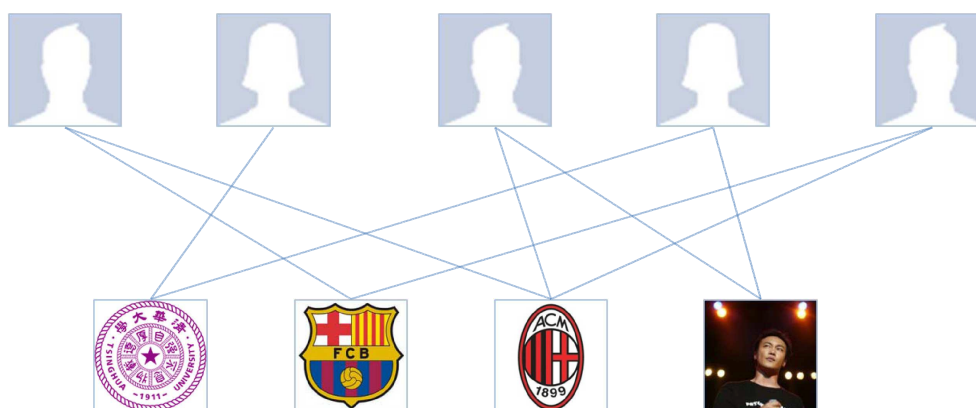


Figure 1.2: A graphical representation for user-page recommendation.

possibly arouses secondary desire and curiosity for more discoveries. The latter has been critical to boost up engagement metrics for web sites like Facebook and Amazon by keeping users onsite, both of which have a rich set of multimedia contents to offer.

A *querying system* for such multi-aspect data provides an interface to retrieve records within and across modes that best match users' information need. Figure 1.1(a) depicts an interface explicitly requesting user inputs to query the Web, whereas Figure 1.1(b) sits at the other end of the gamut which infers users' interests based on their profiles and past activities, not uncommon in nowadays online recommendation applications.

A key algorithmic part of the querying system is the derivation of a quantitative similarity measure. For the page recommendation example, a graphical representation is illustrated in Figure 1.2, with two layers of nodes, users and pages, as well as inter-layer links indicating the user becomes a fan of the page. The intuition about closeness is that when two pages have a larger fraction of overlapping fan bases, *e.g.*, pages "FCB" and "ACM" compared with any other combinations, they are similar to each other and thereby closer to each other's existing fans; iteratively, if a pair of users have quite a few pages of common interest, one's favorite pages could

be recommended to the other if she is not yet a fan. This notion could be reflected by a measure of proximity over graphs by performing random walk with restarts [57] and computing the steady-state probability, which could be obtained efficiently even for million-node graphs [108].

On an additional note, the graph transformation itself could be non-trivial. With regards to the previous user-page graph, intra-layer edges could be introduced between users who are friends in a social network, and edge weights may be further assigned according to, say, the number of mutual friends they share and the frequency they are involved in the same thread of posts and comments. Pages may bear a categorical attribute such as people, sports, movies, and alike, this could be made an additional layer of nodes or be incorporated in the decision to link pairs of nodes within the page layer.

Armed with the foregoing idea to formulate the querying problem into graph search, we present the implementation of an online interface to offer cross-modal search capacity for an annotated biological image database in the opening chapter of the third part of the dissertation. The succeeding chapter, in a different vein, studies multimedia corpora where each querying unit itself is represented as features from more than one modes bearing quite different semantics. A family of probabilistic graphical model is introduced as the solution, which provides both flexibility to accommodate heterogeneous numerical features and interpretability by summarizing concepts or themes from the data and relating them to perceivable entities (*e.g.*, words and imageries), a highly desirable property for practitioners. The last topic of this part is belated to user interaction with a querying system, given the name “user implicit feedback”. Statistical models abstracting users’ behavior as they examine the list of querying results and issue clicks along the way are proposed, with the ambition to improve future ranking for elevated user experience.

1.4 Thesis Organization

The body chapters of the thesis are organized into two parts. The part that comes first consists of the following two chapters addressing mining problems:

- *QMAS*: low-labor labeling, representative finding, and singling out anomalies for satellite images and other image collections. The proposed algorithm yields a 40x speed up over the baseline without loss of labeling quality. This study is presented in Chapter 2 and was published as [30].
- *MultiAspectForensics*: mining heterogeneous network data using tensor decomposition with applications in cyber-security surveillance and pattern discovery in structured knowledge base. Two novel types of subgraph patterns were discovered from data sets across domains. This study is presented in Chapter 3 and was published as [75].

The theme of the later part is querying and it spans over following three chapters:

- *CDEM*: an online query interface for *Drosophila* embryo image databases. It supports cross-modal queries over a large database which consists of genes, embryo images documenting gene expression, and image annotations. This study is presented in Chapter 4 and was published as [48].

- *BEFH*: a Bayesian approach to inference and learning with a family of undirected graphical model for classification and retrieval in multimedia corpora. This study is presented in Chapter 5 and was published as [47].
- *Click Models*: learning to rank from Web search click data by defining and estimating user-perceived relevance of search results. The highlighted model provides an easy and efficient solution to account for the position-bias inherent in the data, and achieves 7% improvement over the baseline as measured by a popular log-likelihood metric, and 30% performance boost when predicting last click positions. This study is presented in Chapter 6 and was published as [51].

Figure 1.3 provides a summary of data sets as well as how they are referred by aforementioned studies.

Chap.	Name	Type	Size	Description
2	GeoEye	Satellite Imagery	17MB	14 high quality images of cities around the world, divided into 14,336 rectangular tiles.
2	SAT1.5GB	Satellite Imagery	1.5GB	3 huge satellite images in GeoTIFF format, divided into 721,408 rectangular tiles.
2	SATLARGE	Satellite Imagery	1.8GB	A 4-band multi-spectral proprietary image, divided into 2.57M hexagonal tiles.
3	LBNL	Network Traffic Log	281K non-zero elements	4-mode data representing packet traces recorded on servers within the Lawrence Berkeley National Lab.
3	RTW	Web Knowledge Base	10K non-zero elements	3-mode triplet data from the NELL system at Carnegie Mellon University such as (pittsburgh, city-located-in-state, pa).
3,4	BDGP	Bioimage Data	38K non-zero elements	3-mode data of images documenting the expression patterns of genes in the early development of <i>Drosophila</i> .
5	TRECVID	Multimedia Corpora	1,078 video clips	Obtained from compiled TRECVID'03 news video collection with a total of 1,894 word features and 166 image features extracted.
6	Click Log	Web Search Click Data	8.8M query sessions	Impressions and clicks for top-10 positions sampled from a major commercial search engine in July 2008.

Figure 1.3: A summary of data sets included in this manuscript.

Part II

Mining Multimedia Data

Chapter 2

QMAS: Querying, Mining And Summarization of Multi-modal Databases

Given a large collection of images, very few of which have labels, how can we guess the labels of the remaining majority, and how can we spot those images that need brand new labels, distinct from the existing ones? Current automatic labeling techniques usually scale super linearly with the data size, and/or they fail when the amount of labeled data is very limited. In this chapter, we introduce *QMAS* (Querying, Mining And Summarization of multi-modal databases), a fast solution to the following problems:

- *low-labor labeling* – given a collection of images, *very few* of which are labeled with keywords, find the most suitable labels for the remaining ones;
- *mining and attention routing* – in a similar setting, find clusters and representative images for each cluster, as well as a set of outlier images.

This chapter is based upon the work published in [30] and [31]. The rest of this chapter is organized as follows: we start with an overview of the proposed approach in Section 2.1, followed by discussion of related work in Section 2.2. Algorithmic details are presented in Section 2.3, and empirical results are covered by Section 2.4. Section 2.5 concludes the chapter.

2.1 Overview

The problem of automatically analysis, labeling and understanding large collections of images appears in numerous fields. Our driving application is related to satellite imagery, involving a scenario in which a topographer wants to analyze the terrains in a collection of satellite images. We assume that each image is divided into smaller tiles (say, 16x16 pixels). Such a user would like to make the effort to create labels (*e.g.*, *Water*, *Concrete*, *Trees*, *etc*) for only a small number of tiles, and then expect an automatic algorithm to generate labels for all the rest. The user would also like to know what pieces of land exist in the analyzed regions look “strange”, not matching any of the known labels, since they may indicate anomalies (*e.g.*, de-forested areas, potential environmental hazards, *etc.*), or errors in the data collection process. Finally, the user would like

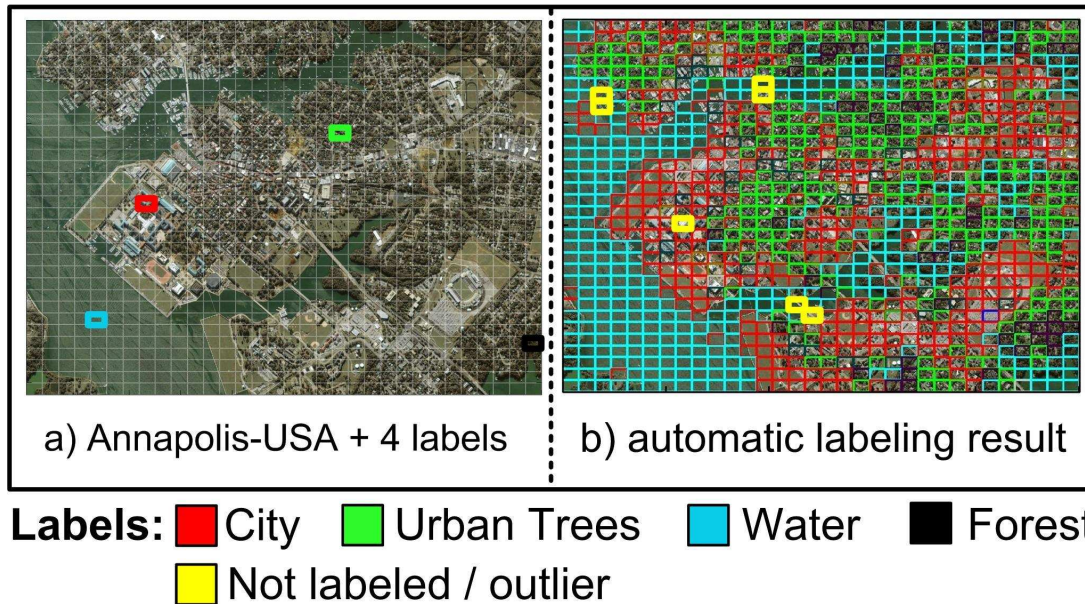


Figure 2.1: An illustration of labeling results from the proposed algorithm. Left: the input satellite image of the city of Annapolis, divided in 1,024 (32x32) tiles, only four of which are labeled. Right: suggested labels from *QMAS*; yellow indicates outliers which are likely to represent “Bridge”.

to have a few tiles that best represent each kind of terrain.

Such requirements are not only limited to satellite image analysis but also arise in several other applications including medical image and biological image pattern analysis. For instance, a doctor wants to find tomographies similar to the images of his/her patients as well as a few examples that best represent both the most typical and the most strange image patterns [32, 66], or a biological expert with a set of imaging data such as the expression pattern in the early development of fruit fly embryos [106] may need a system to answer a similar set of questions.

Our goals are summarized in two research problems:

- *low-labor labeling* – given a collection of images, up to a few of which are labeled *a priori*, find the most appropriate labels for remaining ones.
- *mining and attention routing* – in a similar setting, find clusters, a set of images that best represent the data patterns and another set which consists of top outliers which stand out from existing patterns with labels.

Figure 2.1 illustrates the input and output of the proposed algorithm for *low-labor labeling*. The satellite image, public available at www.geoeye.com, displays part of the city of Annapolis, MD, USA. We split it into 1,024 (32x32) tiles, very few (four) of which were manually labeled as “City” (red), “Water” (cyan), “Urban Trees” (green) or “Forest” (black), as shown on the left figure. *QMAS* automatically produced the label and results are shown on the right. A vast majority of tiles are correctly labeled, and a few outlier tiles, highlighted in yellow, are also picked up, with the implication that they possibly deserve new label(s) of their own. A closer in-

spection in this example concluded that the outlier tiles tend to lie on the border between “Water” and “City”, and are likely to contain a bridge.

For the problem of *mining and attention routing*, we take the approach of finding clusters in the data without the information from the user-provided labels at the beginning. The pure image-based results may be aligned and compared with the evidence from existing labels, possibly leading to suggestions for refinement such as merging too specific labels that are difficult to be differentiated in practice (*e.g.*, “Forest” and “Urban Trees”), and/or splitting too general labels of which tiles are not quite alike (*e.g.*, “Shallow Water” and “Deep Sea” could replace a single label of “Water”). Another advantage it offers is that it enables group labeling – the labeling unit could be a cluster instead of a single tile.

Table 2.1 provides a list of most common acronyms appearing in this chapter.

Table 2.1: Summary of Acronyms

Acronym	Explanation
ANN	Approximate Nearest-Neighbor, an algorithm for fast nearest-neighbor searching
GBT	Generalized Balanced Ternary, a hexagonal mathematical system for feature extraction
GCap	Graph-based automatic image captioning, the baseline image annotation algorithm
MrCC	Multi-resolution Correlation Cluster detection, a scalable subspace-clustering method[32]
RWR	Random Walk with Restart for establishing proximity between pairs of nodes in graph
ViVo	Visual Vocabulary, an algorithm to group image tiles into a set of visual terms

2.2 Related Work

2.2.1 Image Labeling

There is an extensive body of work on the classification of unlabeled regions from partially labeled images in the field of computer vision, such as image segmentation and region classification [46, 69, 98, 110]. The conditional random fields (CRF) and boosting approach in [98] shows the competitive accuracy for multi-class classification and segmentation, but it is relatively slow and requires a lot of training examples to get started. The random walk segmentation method in [46] is closely related to our work, but scalability is beyond the scope of that work since it is concerned with the segmentation of a single image. The KNN classifier in [110] may be the fastest way for region labeling, but it may be sensitive for outliers. The empirical Bayes approach in [69] is able to learn contextual information from unlabeled data. However, it may be difficult to apply to satellite image sets.

Graph-based methods provide a flexible tool for automatic image captioning. Images and caption keywords are represented by multiple layers of nodes in a graph. Image content similarities are captured by edges between image nodes, and existing image captions become links between corresponding images and keywords. Such techniques have been previously used in GCap [84], in which a tri-partite graph was built based on captioned images, further segmented

into regions. Given an image node of interest, the Random Walk with Restart (RWR) algorithm, which resembles semi-supervised label propagation on graphs [120], was used to perform proximity query to automatically find the best annotation keyword for each region. RWR is usually computed using the power iteration method, which converges in a few iterations in most cases. Another alternative algorithm for labeling is the transductive support vector machine, which has been shown to be efficient and accurate for data which comes with very high dimension and sparse features like word counts [59]. In the satellite imagery we studied in this chapter, the number of dimensions does not go beyond a few dozen and certain features may be irrelevant to the labeling class.

To create edges between similar image nodes, most previous work searches for nearest neighbors in the image feature space. However, this operation is super-linear even with the speed up offered by many approximate nearest-neighbor finding algorithms (e.g., the ANN Library [80]). Given millions of image tiles in satellite image analysis, greater scalability is almost mandatory.

2.2.2 Clustering

Most clustering algorithms assume the following cluster definition: a cluster is a region in the feature space in which the objects are dense. This region may have an arbitrary shape, and the points inside it may be arbitrarily distributed. k -means like methods start by picking k points in the metric space as cluster centers, or centroids, through a random process or by applying some specific heuristics for this task. The clustering is made possible by an iterative process that assigns objects to their closest centroids, and iteratively improving the centroids according to the objects assigned to each cluster. The computation stops when a quality criterion is satisfied or when a maximum number of iterations is achieved. An example of this approach is K-Harmonic Means [117]. The main drawbacks of this approach are that it assumes that the clusters have hyper-spherical shapes in the data space and that the number k of clusters should be determined by the user *a priori*.

The Visual Vocabulary (ViVo) [14] method is particularly useful for our work. ViVo is a novel approach, proposed for the analysis of biomedical images, that applies Independent Component Analysis (ICA) to group image tiles into a set of visual terms, avoiding subtle problems, such as non-Gaussianity.

2.2.3 Feature Extraction

Feature extraction is generally considered to be a low-level image processing task and is closely related to feature detection. Histogram-based features are perhaps the simplest and most popular type of features. Texture-based features such as wavelets and fractals are able to capture more subtle spatial variations such as repetitiveness. Local feature descriptors such as SIFT [74] and SURF [12] have also been widely used, as well as the Generalized Balanced Ternary (GBT) [44], a hexagonal mathematical system that allows feature extraction. A recent example of GBT's usage in target recognition is found in [45]. The choice of candidate features is usually domain-specific and may also be subject to scalability constraints in large scale analysis. The feature

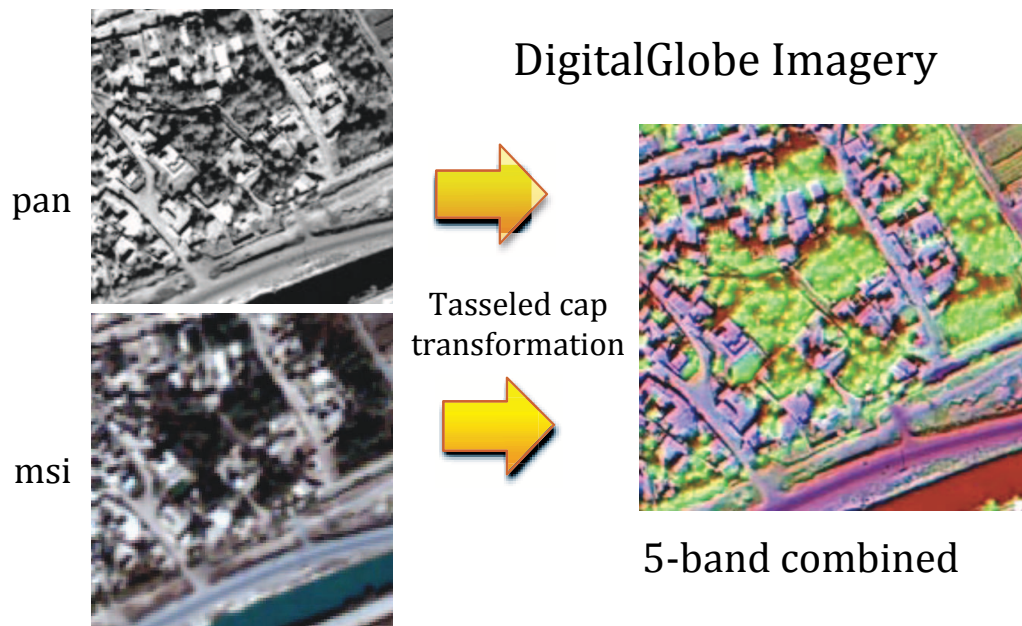


Figure 2.2: Pre-processing applied to multi-band satellite imagery. Best viewed in colors. Left: sample input multi-band image; Right: the resulting 5-band composite image for which features are computed.

extraction procedures applied in our study will be introduced in the next section.

2.3 Proposed Method

In this section we discuss algorithmic details of *QMAS*, starting with the feature extraction procedure to obtain a compact yet informative representation of image pixels. Then procedures for mining and attention routings are introduced, which include clustering, representative identification and outlier discovery. The low-labor labeling technique is presented in sequel.

2.3.1 Feature extraction

Two approaches feature extraction were employed for different data sets. For public available image collections, we obtained Haar wavelets in two resolution levels, plus the mean value of each band of the images. For proprietary image collections, we applied an alternative approach: first, there was a pre-processing step resulting in a 5-band composite image as illustrated in Figure 2.2. The first four bands are the 4-band tasseled cap transformation (TCT) of 4-band multi-spectral data, and the fifth band is the panchromatic band.

Feature generating following this second approach utilizes a variety of characteristics, including statistical measures, gradients, moments, and textual measures. For multi-scale image characterization, which is crucial for finding patterns at various resolutions, we adopted Generalized Balanced Ternary (GBT). We mapped the raster pixel data into the GBT space and calculated

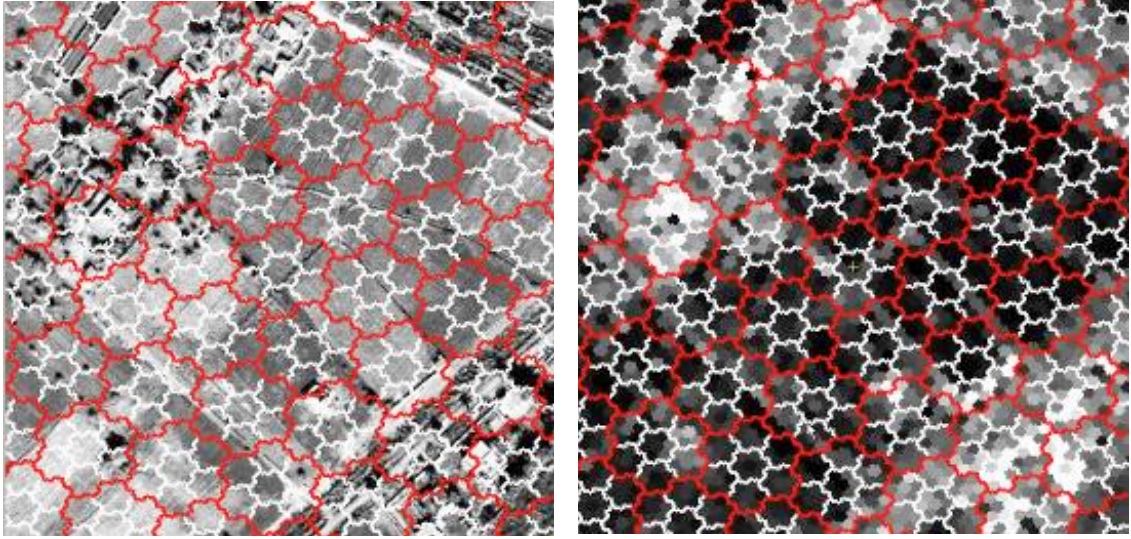


Figure 2.3: GBT structure illustrated. Left: two levels of GBT cells with 343 pixels (Level-3, outlined in white) and 2,401 pixels (Level-4, outlined in red) overlaid on an image. Right: output values were assigned according to the variance of the adjacent lower level of cells (at Level-2, consisting of 49 pixels each). Bright areas have greater variance, dark areas have less.

a set of moments-based features over the multi-scale hierarchy of GBT cells. The GBT structure is such that any cell or aggregate at a given layer in the hierarchy contains seven hexagonally grouped aggregates or hexagons (if at the pixel level) in the layer below it. The cells form a hexagonal tiling of the pixels at a variety of scales, effectively describing the image in multiple resolutions. A sample of GBT structure and simple computations is shown in Figure 2.3.

Image features such as mean, variance, and GBT texture are calculated for GBT aggregates in each of the five bands of data. The final feature set comprises a 30-dimensional feature vector per aggregate: mean, variance, and GBT texture of the Level- n aggregate in each of the five data bands plus the mean, variance, and GBT texture of the Level- $n + 1$ aggregate centered at that Level- n position in each of the five data bands.

Following this feature extraction, we adapted the ViVo method [14] to group image tiles into a set of visual terms, with a slight modification to incorporate GBT aggregate features. If a tile cannot be represented by the vocabulary already known to ViVo, then it will automatically derive a new type of tiles (represented by a new visual term), as needed. These new types represent natural groupings of tiles in the feature space and indicate where new labels could potentially improve the accuracy of *QMAS*.

2.3.2 Mining and Attention Routing

Clustering

We start by clustering image tiles and subsequently determine representatives and outliers based on the output. The clustering step over the set of images I is performed by a modified version of

the MrCC algorithm. As described in Section 2.2, MrCC is a fast subspace clustering algorithm designed to look for clusters in large collections of medium-dimensionality data. The original MrCC algorithm is composed of three main steps: (i) data structure construction; (ii) identification of initial clusters, named β -clusters, which are axis-parallel hyper-rectangles with high data densities; and (iii) overlapping β -clusters merging. Here we bypass the third step to obtain “soft” clusters, where a single tile can belong to one or more clusters, such as “Trees” and “Water”.

Finding Representatives

Now we focus on the problem of selecting a set of elements R , of size N_R specified *a priori*, to represent a given set of images I . The set of representatives R should have the following property: for every image I_i in I there should be a representative R_r from R that is mostly similar to I_i . Assuming that these images are already embedded in a metric space, a straightforward optimization goal is to minimize the sum of squared distances between each image I_i and its closest representative R_r . The solution is simply the popular clustering algorithm K-means. However, the method is known to be sensitive to skewed distributions, data imbalance, *etc*, which is not uncommon for our use case in studying the satellite imagery. Here we propose to optimize the following dis-similarity function instead:

$$E_{QMAS}(I, R) = \sum_{I_i \in I} \frac{N_R}{\sum_{j=1}^{N_R} \frac{1}{\|I_i - R_j\|^2}} \quad (2.1)$$

Therefore, instead of focusing on the minimum of distance between the target image and representatives, here the harmonic mean is the concern, which is usually more robust to extreme data distributions and unfortunate initializations. The solution to this metric is known as K-harmonic means [117].

Finding Outliers

The goal in this part is to find an ordered list of potential outliers, images of I that diverge most from main data patterns. We take the representatives found in the previous section as the basis for the outliers definition, *i.e.*, assuming that a set of representatives R is a good summary of I , the N_O images from I worst represented by R are said to be the top- N_O outliers. Formally, *QMAS* uses the harmonic mean of the squared distances between an image I_i and each one of the representatives in R to measure the quality of the representation of I_i , therefore the top outlier is identified by

$$\arg \max_i \frac{N_R}{\sum_{j=1}^{N_R} \frac{1}{\|I_i - R_j\|^2}} \quad (2.2)$$

2.3.3 Low-labor Labeling

Our approach is to represent input images and labels, together with the image clusters found before, in a graph G , known as the *knowledge graph*. A random walk-based algorithm is applied

over G to find the most appropriate labels for the unlabeled images. Algorithm 1 shows a sketch of our solution, and details are given in the remainder of this subsection.

Algorithm 1 : *QMAS labeling*.

Input: collection of images I ;

collection of known labels L ;

restart probability c ;

clustering output C .

Output: full set of labels LF .

- 1: use I , L and C to build the graphical representation G ;
 - 2: **for** each unlabeled image $I_i \in I$ **do**
 - 3: random walk over graph G from vertex $V(I_i)$, and with probability c , restart the walk from $V(I_i)$ again;
 - 4: compute the affinity between each label of L and I_i using power iteration method implementing a revised random walk with restart algorithm;
 - 5: let L_l be the one with the largest affinity value in set in L , then $LF_i \leftarrow L_l$;
 - 6: **end for**
 - 7: **return** LF ;
-

G is a tri-partite graph that consists of the vertex set V and the edge set E . V is made up of three layers corresponding to the input images I , the clusters of images C , obtained with algorithms described in Section 2.3.2, and the set of image labels L . Vertices in G that represent image I_i and label L_l are denoted by $V(I_i)$ and $V(L_l)$, respectively. It is self-evident that with clustering results in hand, the construction process of such a graph G is linear in time and space. Figure 2.4 exemplifies a toy graph with seven images, two distinct labels, and three clusters. Image I_1 is pre-labeled with L_1 , while I_4 and I_7 are both pre-labeled with L_2 . Note that under the soft clustering scheme we adopted, an image node may be linked with more than one nodes representing clusters, *e.g.*, I_3 is connected to both C_1 and C_2 .

Given an unlabeled image I_i , we apply the following algorithm over the graph G in order to find an affinity score for each possible label with respect to I_i ; it is imitating the well-known random walk with restart algorithm with a minor modification on selecting the random neighbor to land on. The random walker starts from vertex $V(I_i)$ initially. At each time step, the walker always takes one of the following two choices: (1) go back to $V(I_i)$, with probability c ; (2) walks to a neighboring vertex, with probability $1 - c$. Under the latter case, the probability of choosing a neighboring vertex is proportional to the degree of that vertex, *i.e.*, the walker favors smaller clusters and more specific labels. The value of c is usually set to an empirical value (*e.g.*, 0.15), or determined by cross-validation. The affinity score for L_l with respect to I_i is given by the steady state probability that our random walker will find himself at vertex $V(L_l)$, always restarting at $V(I_i)$. The label with the largest score becomes the recommended label for I_i .

The intuition behind this procedure is that similar images that belong to the same cluster should share similar labels. This is consistent with our graph proximity measure which favors multiple short paths between the two vertices of interest. For instance, consider image I_6 in

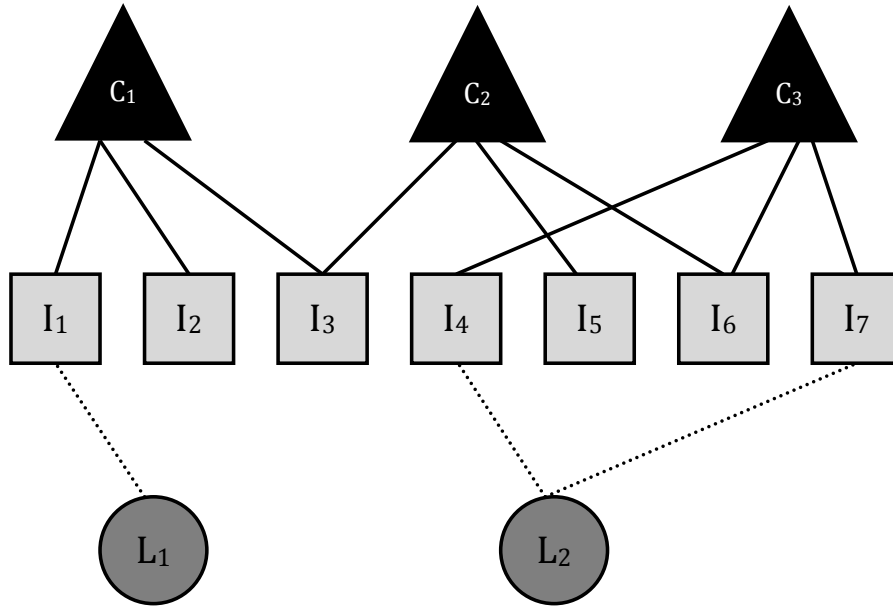


Figure 2.4: The knowledge graph for a toy input set. Nodes shaped as squares, circles, and triangles represent images, labels, and clusters respectively.

Figure 2.4. It belongs to clusters C_2 and C_3 . The other two images in C_3 bear label L_2 , whereas none of the images in C_2 is labeled. Hence there is a higher probability that a random walker starting from $V(I_6)$ will reach $V(L_2)$ than $V(L_1)$, in that there are two shortest paths of length 3 linking $V(I_6)$ and $V(L_2)$, whereas the only shortest path connecting $V(I_6)$ to $V(L_1)$ takes as many as 5 steps. Moreover, the affinity score for L_2 could be higher if I_6 were associated with C_3 only. Thus, for larger graphs, in which it is not untypical that an image belongs to multiple clusters, the membership with a smaller cluster takes more weight than that with a larger one.

2.4 Experimental Results

2.4.1 Experimental Setup

We first introduce three data sets of satellite images that serve at the test beds in this section:

- *GeoEye* – 14 high quality satellite images in JPEG format of a number of cities around the world. The total size of these images is 17 MB. We divided each image into equal-sized rectangular tiles and yielded a total of 14, 336 tiles. Figure 2.1 includes a snapshot of 1, 024 tiles.
- *SAT1.5GB* – the data set is made up of three satellite images in the GeoTIFF format, each of size ~ 500 MB. The total number of equal-sized rectangular tiles is 721, 408.
- *SATLARGE* – this proprietary data set contains a pan QuickBird image of size 1.8 GB, and its matching 4-band multi-spectral image of size 450 MB each. These images were

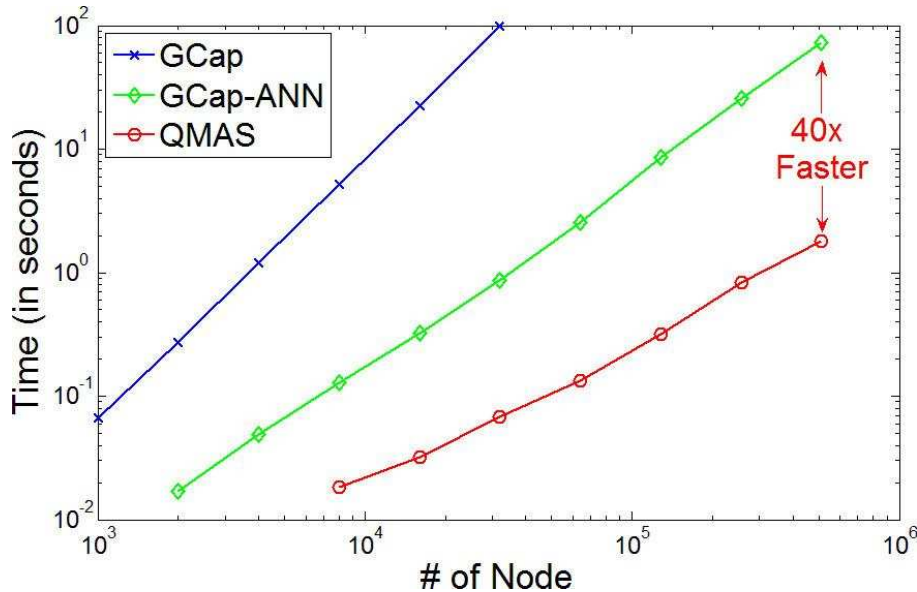


Figure 2.5: Graph construction time versus size of the subsets randomly sampled from *SATI.5GB*. *QMAS*: red circles; *GCap*: blue crosses; *GCap-ANN*: green diamonds. Timing results are averaged over 10 runs; error bars are too tiny to be discerned.

combined as described previously in Section 2.3.1, and 2,570,055 hexagonal tiles were generated.

The experiment is designed to demonstrate the performance of *QMAS* in terms of computational time, non-labor intensive labeling, and identification of representatives as well as outliers. We also highlight results from a set of *query-by-example* tests performed over the proprietary data; such queries exemplify practical retrieval tasks in satellite image analysis. The baseline algorithm to be compared against for performance evaluation is the Graph-based automatic image Captioning method [84], with two different approaches of nearest neighbor finding in the graph construction: either the basic quadratic algorithm (*GCap*) and or the approximate nearest neighbors using the ANN Library (*GCap-ANN*). The number of nearest neighbors is set to seven. All three approaches share the same implementation of random walk algorithms with the restart parameter set to $c = 0.15$. They were executed on a LINUX server using a single 2.8GHz CPU core and 4GB RAM available.

2.4.2 Computational Time

Figure 2.5 compares the elapsed time for graph construction using the *SATI.5GB* data set and smaller randomly sampled subsets. On the entire *SATI.5GB* data set, *QMAS* is 40 times faster than *GCap-ANN*, while running *GCap* will take hours (not shown). *QMAS* scales almost linearly with the input data size, while the slope of log-log curves are 2.1 and 1.5 for *GCap* and *GCap-*

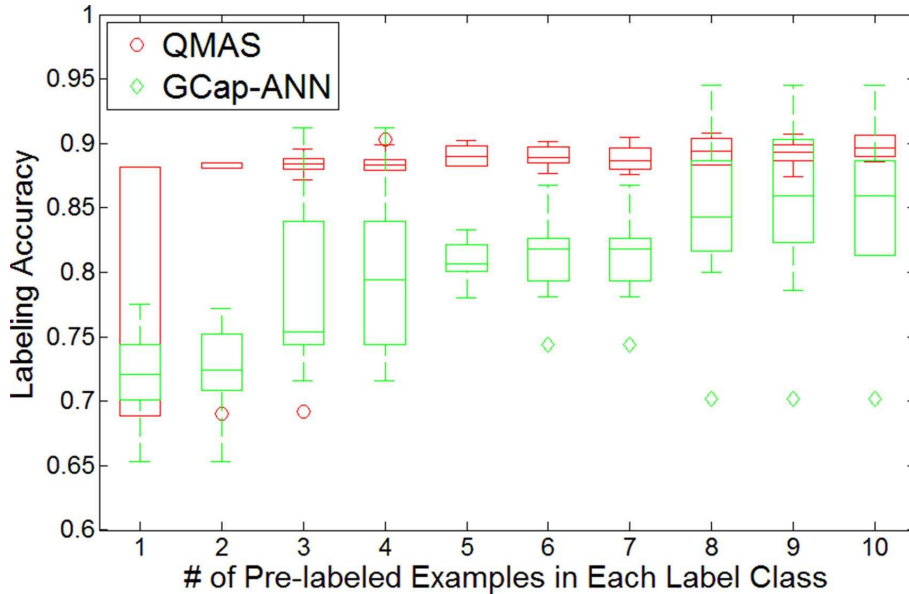


Figure 2.6: Labeling accuracy versus the number of pre-labeled examples for each labeling class. *QMAS*: red circles; *GCap-ANN*: green diamonds. Accuracy values of *QMAS* are barely affected by the size of the pre-labeled data. Box plots summarize 10 runs over randomly selected inputs, with outliers (typically over 3 standard deviations from the mean) indicated by red circles and green diamonds.

ANN, respectively. Instead of performing nearest neighbor searches, which is super-linear even with a state-of-the-art approximation algorithm, *QMAS* employs a linear-time clustering algorithms to leverage the content similarity in satellite image tiles, and achieves superior scalability.

2.4.3 Non-labor Intensive Labeling

We manually labeled 256 tiles in the *SATI.5GB* data set as the ground truth. By randomly choosing a small number of these labels as input and leaving out remaining ones for evaluation, we compare the labeling accuracy of the three approaches from 10 repetitive runs and display quality results as box plots in Figure 2.6. *QMAS* does not sacrifice quality for faster computational time when compared with *GCap-ANN*, and it actually performs even better when the size of the pre-labeled data is limited. Additional experiments have shown given 10 pre-labeled examples for each class, even under the optimal performance-speed trade-off for *GCap-ANN*, where the number of nearest neighbors set to three, *QMAS* is still 1.75 times faster and around 10% more accurate. Note that the accuracy of *QMAS* is barely affected by the number of the pre-labeled examples in each label class. The fact that it can still extrapolate from tiny sets of pre-labeled data ensures its non-labor intensive capability.

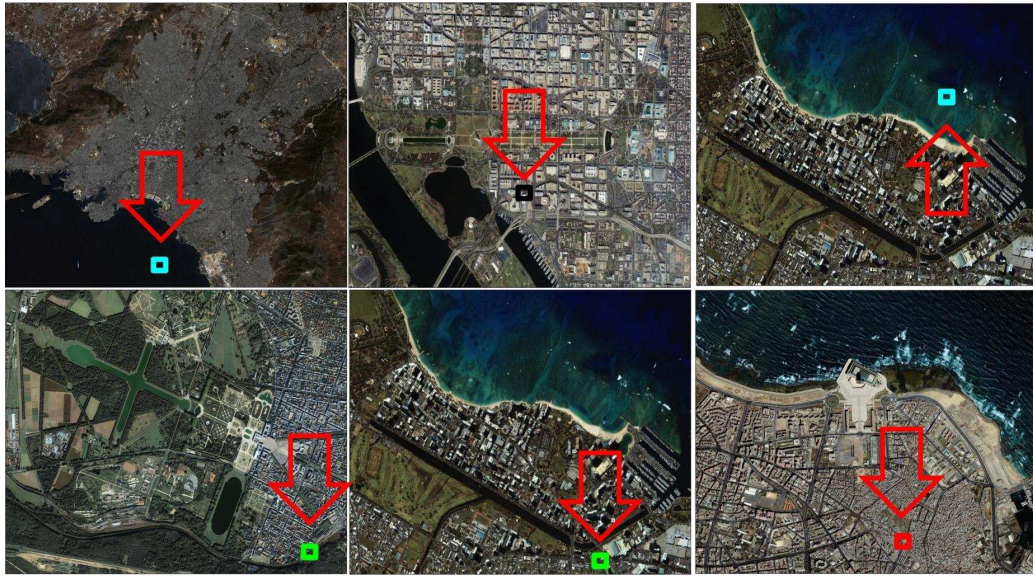


Figure 2.7: Representatives for the *GeoEye* dataset, each colored according to the cluster membership.



Figure 2.8: Top-3 outliers for the *GeoEye* dataset.

2.4.4 Finding Representatives and Outliers

Figure 2.7 shows data representatives obtained on the *GeoEye* data set. A total of 6 representatives are displayed, which are colored according to their clusters. Note that a large cluster, such as “Water”, may have multiple representatives.

Figure 2.8 presents the top-3 outliers on the same data set. Closer inspection found that these outlier tiles tend to be on the border of areas like “water” and “city”, where a bridge usually appears. These 3 outlier tiles, together with 6 representatives, compactly summarize the *GeoEye* data set, which contains more than 14 thousand tiles.

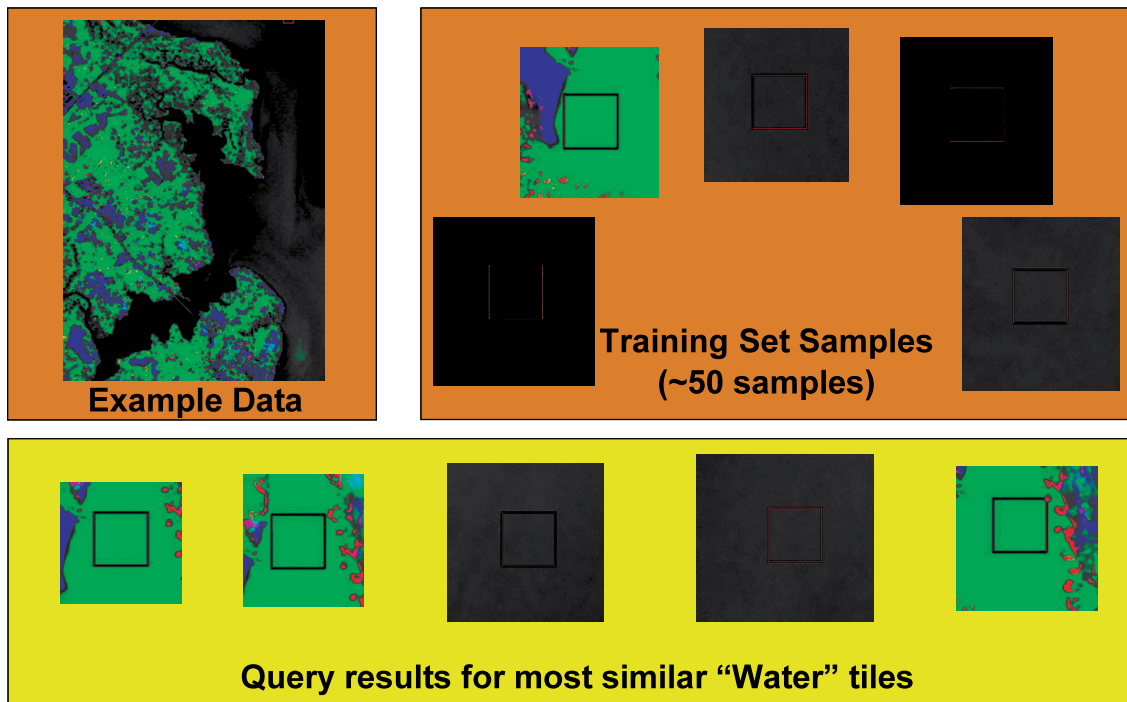


Figure 2.9: Example “Water”: Labeled Data and Results of Water Query.

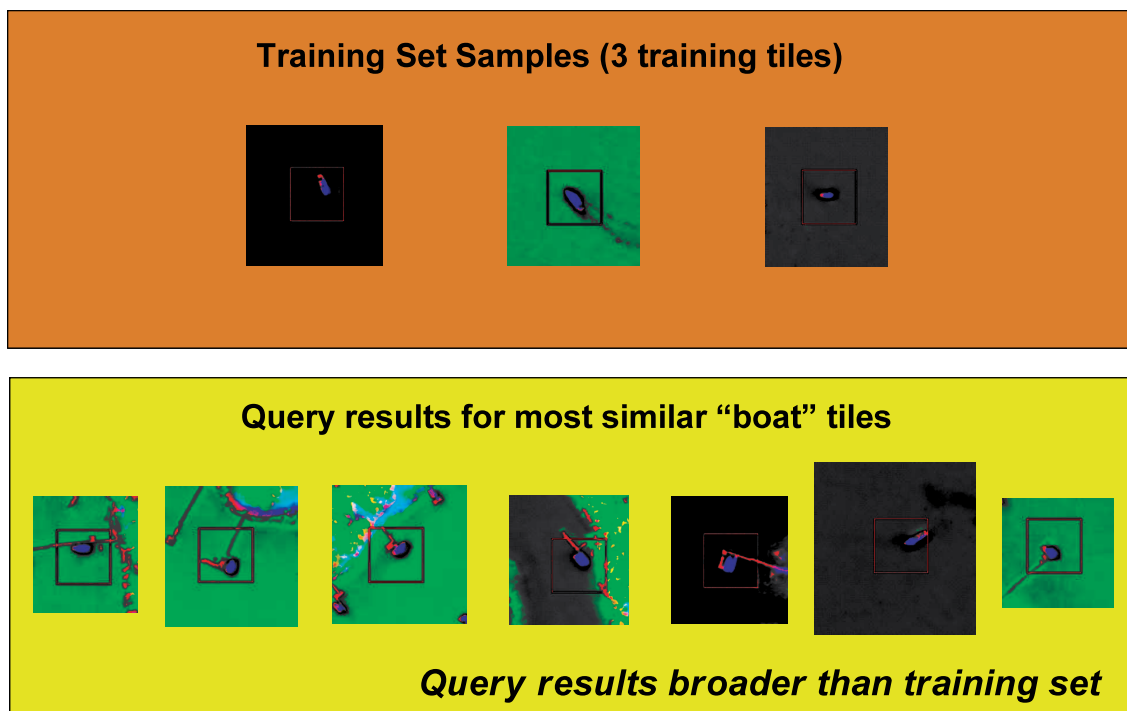


Figure 2.10: Example “Boats”: Labeled Data and Results of Boat Query.

2.4.5 Query-by-Example Experiments

The query-by-example experiments were carried out for the proprietary *SATLARGE* data set by domain experts in satellite image analysis. Given a small set of tiles as *examples*, they would like to find most similar tiles over the entire data set. To apply *QMAS*, the given tiles are assigned with a single label, and we performed random walk based algorithms to find tiles, other than those already given, that are mostly similar to this label. For instance, Figures 2.9 and 2.10 exemplify the results obtained for “Water” and “Boats”, respectively. We also varied the size of the pre-labeled data in these experiments between one and fifty, to observe how the system responded to these changes. In general, labeling only small numbers of examples (*even less than five*) still leads to accurate results; when the number was reduced to 2, we began to see the negative effects of having an exceedingly small input set.

Notice that correct returned results often look very different from the given samples, *i.e.*, the system is able to extrapolate from the given examples to other, correct tiles that do not have significant resemblance to the pre-labeled set. Clearly, this is not a typical automated target recognition (ATR) approach. There are no “templates” and no specific object shapes, orientations, sizes, or patterns that are learned. Unlike a traditional ATR that typically fails when it encounters an object that does not fit the specified description, *QMAS* is able to correctly label an object that has a somewhat different appearance from the “known” set.

2.5 Conclusion

In this chapter we proposed *QMAS*, a fast solution to low-labor labeling, mining and attention routing for multi-modal databases. We carried out experiments in the scenario of satellite image analysis to evaluate its performance. *QMAS* scales linearly over the size of the data set, being multiple times faster than an alternative algorithm in graph construction. At the same time, it provides high quality labeling results, even with tiny sets of pre-labeled data as inputs. It could also spot top representatives and outliers and offered a compact summarization of a large data set. The implementation was also employed to perform a set of practical queries over a proprietary data set by domain experts and it yielded quite positive results – *QMAS* is able to correctly label an object where the traditional automated target recognition (ATR) approach may fail.

Future directions include leveraging the locality within images, *i.e.*, the fact that image tiles that are neighbors are more likely to share similar labels, and generalizing the method to handle an ontology of labels.

Chapter 3

***MultiAspectForensics*: Pattern Mining on Large-scale Heterogeneous Networks with Tensor Analysis**

Modern applications such as web knowledge base, network traffic monitoring and online social networks have made available an unprecedented amount of network data with rich types of interactions carrying multiple attributes, for instance, port number and timestamp in the case of network traffic. The design of algorithms to leverage this structured relationship with the power of computing to assist researchers and practitioners for better understanding, exploration and navigation of this space of information has become a challenging, albeit rewarding, topic in social network analysis and data mining. The constantly growing scale and enriching genres of network data always demand higher levels of efficiency, robustness and generalizability where existing approaches with successes on small, homogeneous network data are likely to fall short.

MultiAspectForensics, introduced in this chapter, is a handy tool to automatically detect and visualize novel subgraph patterns within a local community of nodes in a heterogeneous network, such as a set of vertices that form a dense bipartite graph whose edges share exactly the same set of attributes. We apply the proposed method on three data sets from distinct application domains, present empirical results and discuss insights derived from these patterns discovered. Our algorithm, built on scalable tensor analysis procedures, captures spectral properties of network data and reveals informative signals for subsequent domain-specific study and investigation, such as suspicious port-scanning activities in the scenario of cyber-security monitoring.

This chapter will be structured as follows: we first motivate the discussion in Section 3.1, and then elaborate on *MultiAspectForensics* procedures step-by-step in Section 3.2. Empirical results are presented in Section 3.3. And related literatures are briefly sketched in Section 3.4. Lastly, Section 3.5 concludes the chapter and highlights future directions. Most of the work described subsequently is based on the material presented in [75].

3.1 Introduction

Modern applications in the Internet era, either data-informed or data-driven, have contributed to the boom of network data arising from a spectrum of domains, such as web knowledge base, network traffic monitoring and online social networks. A glowing trend in the accumulation and analysis of such data is the emergence of heterogeneous interactions between nodes in the network, for which a vivid depiction is offered by the Facebook friendship page, with multiple page elements ranging from wall posts, comments, and photos, to mutual friends, shared interests and common networks between a pair of users. Browsing and navigation over such a space of information, despite its overwhelming scale and complexity, has been a challenging task common encountered in many fields. Yet the rather recent availability and popularity of these data, in addition to practical requirements over the efficiency, robustness and generalizability of the solution, has rendered the topic of pattern mining for heterogeneous network data a relatively underexplored one, where even the definition of interesting or abnormal *patterns* could become a non-trivial problem itself.

Many of pioneering studies on pattern discovery for graph and network data focused on frequent substructure mining, with heuristics motivated by information theory [29], mathematical graph theory [67, 116], inductive logic programming [35], etc. An intimately related problem is the detection of rare event and anomalous behavior, which has attracted wide interests thanks to its many well-recognized applications concerned with security, risk assessment, and fraud analysis. Noble and Cook [82] were among the first to address this challenge on structured network data by providing solutions based on the minimal description length principle to search for abnormal subgraphs. And many alternative approaches are now available to spot anomalous nodes [6], edges [24], or both [38], with further elaboration adapted to bipartite graphs [103], and time-evolving graphs [109]. This piece of work, by revealing two classes of patterns in the context of heterogeneous graphs, resembles a novel attempt to explore this relatively young realm of multi-aspect network data for state-of-the-art discoveries and developments.

We resort to a tensor-based representation for heterogeneous network data and employ off-the-shelf decomposition algorithms [64] as a starting point of the analysis. Previous research along this line has paid a great deal of attention on individual nodes, which play a central role in similarity ranking [41], personalized recommendation [118], etc. The major finding in our study is that, for multiple heterogeneous network data across diverse application domains, we could always observe groups of elements with similar connections along one or more data modes, as implied by nearly-identical decomposition scores, which transform to quite visible spikes in histogram plots. While algorithms in aforementioned studies mostly look for elements with top eigenscores, our heuristic distinguishes itself by *being able to capture patterns formed by less well-connected nodes in the network*, which do not necessarily stand out in the eigenspace and are often ignored by other extant techniques.

In summary, *MultiAspectForensics* starts with a data decomposition step for input heterogeneous networks, features a spike detection heuristic to reveal non-trivial substructure patterns, and also includes programs to automatically visualize the findings. We demonstrate its effectiveness and efficiency by executing *MultiAspectForensics* on three data sets from distinct appli-

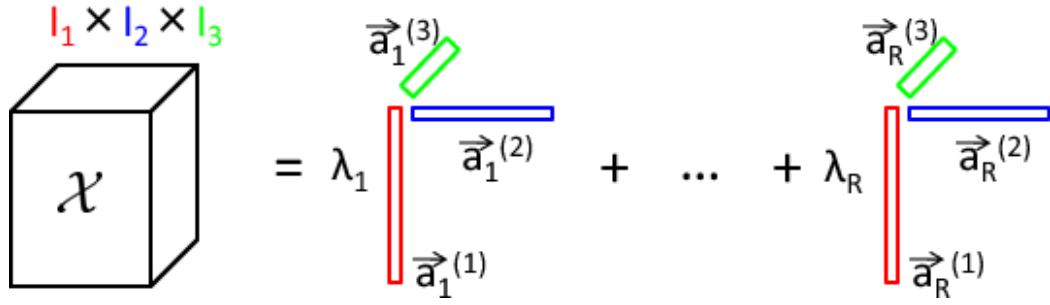


Figure 3.1: Illustration of the CP decomposition: the input 3-mode tensor on the left is decomposed into R triplets of vectors on the right, reminiscing of the rank- R singular value decomposition of a matrix. The three modes, in a scenario of network traffic analysis, may represent source IP address (red), destination IP address (blue) and port number (green), respectively.

cation scenarios, present empirical results and investigate the discovered patterns, which could be leveraged to suggest suspicious activities from network traffic logs such as port-scanning and denial-of-service attack, extract interesting facts from a web knowledge base such as punk musicians or low-cost airline destinations, and report gene function groups in a developmental biology study consistent with established theories.

3.2 Proposed Algorithm

MultiAspectForensics, in a nutshell, consists of the following steps:

- *Data Decomposition*: take the input heterogeneous network as a tensor and perform tensor decomposition to obtain an eigenscore vector along each data mode.
- *Spike Detection in Histograms*: iterate over all data modes to obtain histograms and apply the spike detection algorithm.
- *Substructure Discovery*: identify the induced subgraph/subtensor for each spike and summarize patterns discovered.
- *Visualization*: create attribute plots and histogram plots with detected spikes highlighted.

The above procedure just makes use of the strongest component after data decomposition. If the contribution of the top one eigen-component is not as large, the latter three steps should be carried out over multiple strongest components in a similar fashion. For brevity, we subsequently elaborate on three algorithmic steps with only the first component taken into consideration, and the visualization step is illustrated by resulting figures intermixed with the rest of the discussion.

3.2.1 Data Decomposition

We first introduce a few definitions. A *tensor* can be represented as a multi-dimensional array of scalars. Its *order* is the dimensionality of the array, while each dimension is known as one *mode*, of which the value ranges over the set of *elements* for the specific mode. Thus, vectors

are tensors of order one, and matrices are tensors with two modes. In Section 3.3 we will use *measure* to denote the unit of each *entry* in the multi-dimensional array.

To transform a heterogeneous network into a tensor, every edge becomes a non-zero entry in the multi-dimensional array, where edge attributes, together with edge source and destination, make up different modes of the tensor. Edge weights naturally stay as entry values for weighted networks. Node attributes could also be incorporated by taking a Cartesian product over two end points of an edge, for instance, if a directed network contains nodes with 7 different colors, we could have an edge attribute whose arity is $7^2 = 49$.

Tensor decomposition leverages multi-linear algebra to the analysis of high-order data. The canonical polyadic (CP) decomposition we applied in this chapter generalizes the singular value decomposition (SVD) for matrices. It factorizes a tensor to the weighted sum of outer products of mode-specific vectors, as illustrated in Figure 3.1 for a 3-order tensor. Formally, for an M -mode tensor \mathcal{X} of size $I_1 \times I_2 \times \dots \times I_M$, its CP decomposition of rank R yields

$$\begin{aligned} \mathcal{X}(i_1, \dots, i_M) &\approx \sum_{r=1}^R \lambda_r \left(\overrightarrow{a_r^{(1)}} \times \dots \times \overrightarrow{a_r^{(M)}} \right) \\ &= \sum_{r=1}^R \lambda_r \prod_{m=1}^M a_{ri_m}^{(m)} \end{aligned} \quad (3.1)$$

Similar to SVD, the approximation becomes closer as R enlarges, and would be exact if it equals the rank of the tensor (see [53] for details).

3.2.2 Spike Detection in Histograms

Now that we have transformed complex structured data into a set of more manageable vectors, the next step is to spot common patterns from these vectors. As a starting point, we visualize each vector by creating an attribute plot, which displays absolute values of eigenscores (y-axis) along its elements (indexed by the x-axis). An example of such plots is given in Figure 3.2. Note that the y-axis should be in *log* scale to emphasize the relative difference. The arrow on the right indicates a score value shared by many elements, *i.e.*, a number of entries in the eigenvector have exact the same values, which is not uncharacteristic in other dimensions and across different data sets. *This key observation* enables us to create effective heuristics to extract spikes from histograms and subsequently examine subgraph patterns they imply in the next subsection. And the fact that many spikes do not appear at the very top of the figure with most significant eigenscore values makes it more difficult for many alternative methods to be effective.

Prior to applying the spike detection heuristics, we obtain histogram data by equally dividing the range of eigenscores in log scale. The detection algorithm just needs to sort and traverse the histogram data until one of the following conditions is satisfied: (1) the energy as measured by sum of square values covered is equal or more than a fraction of s , and the magnitude of the spike is less than a fraction of r than the largest one; (2) there are already K spikes. Parameter values are empirically set to $s = 90\%$, $r = 50\%$, $K = 20$, where small variations lead to little perturbation of the output. The pseudo-code of the algorithm is listed in Algorithm 2 above.

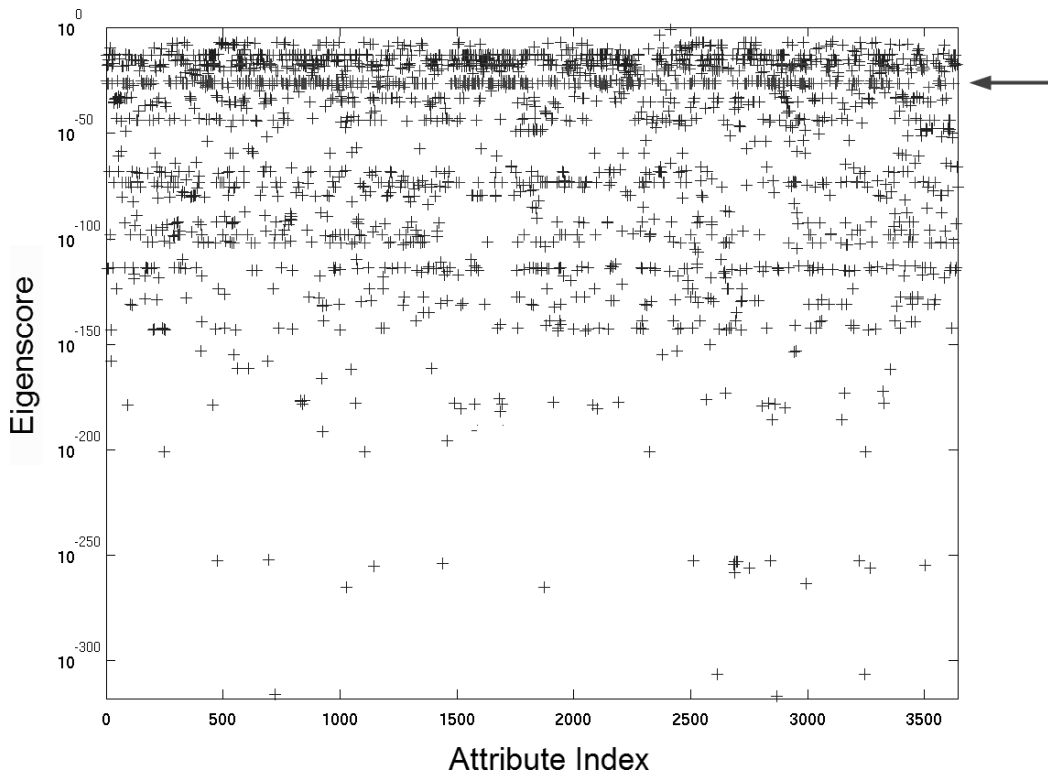


Figure 3.2: An attribute plot which displays absolute values of eigenscores (y-axis in log-scale) along its elements (indexed by the x-axis). The arrow on the right points to a common score value, illustrating an observation critical to the algorithmic design of *MultiAspectForensics*.

Application of this algorithm to the data vector in Figure 3.2 yields Figure 3.3, where we put attribute plot on the left side-by-side with histogram plot on the right, highlighting every spike in red.

3.2.3 Substructure Discovery

Having extracted sets of elements that form histogram spikes from each data mode, we head back to the input network data to examine corresponding local subnetworks to complete the final step of pattern discovery. The running example in this subsection comes from a snapshot of network traffic log which consists of packet traces in an enterprise network [70]. Each trace in the log is a triplet of (*source-IP*, *destination-IP*, *port-number*), which could be represented as a directed network of machine IP addresses with the only edge attribute “port number” and number of packets as edge weights. Patterns derived from *MultiAspectForensics* could be summarized into the following two categories:

generalized star

A subnetwork which consists of conterminous edges that differ only in one data mode. For instance, a group of source IP addresses sending packets to a single destination server using the

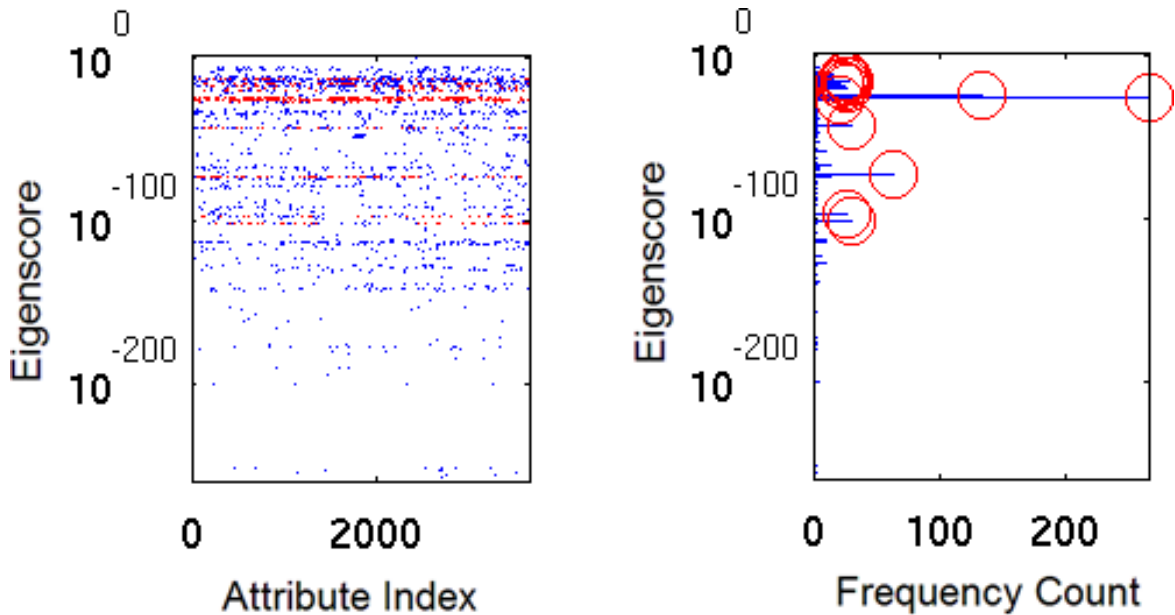


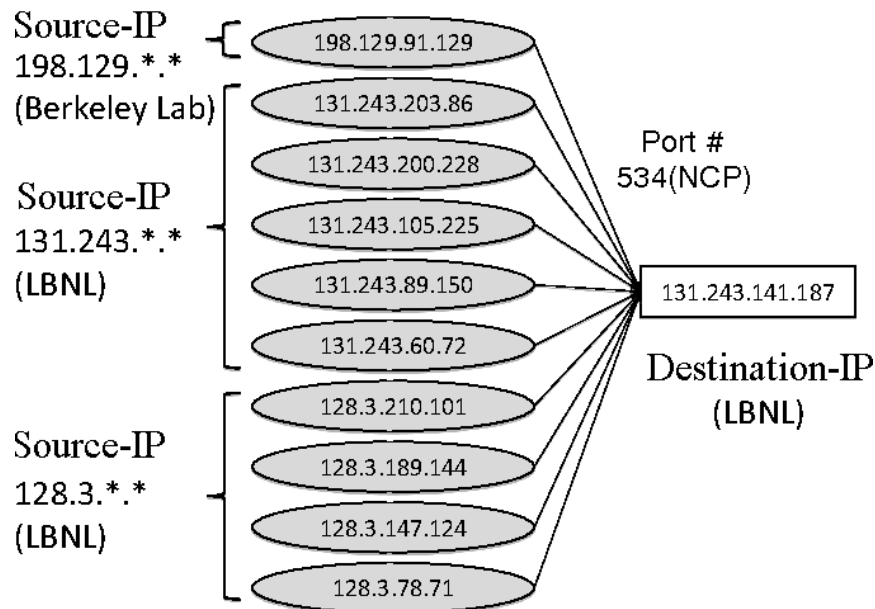
Figure 3.3: An attribute plot (adopted from Figure 3.2) on the left side-by-side with the corresponding histogram plot with spikes detected indicated by circles.

same port. It generalizes the star pattern in two dimensional graphs, and makes up a continuous block along one dimension in the adjacency tensor, if elements along that dimension are ordered carefully. Note that in a heterogeneous network, this category of patterns also includes multiple edges between one pair of nodes with differing attribute values, *e.g.*, a good many port numbers in our running example, in which case the source machine may be either an administrator performing port screening or a suspect trying to exploit a vulnerable port. Figure 3.4 provides an illustration of these patterns.

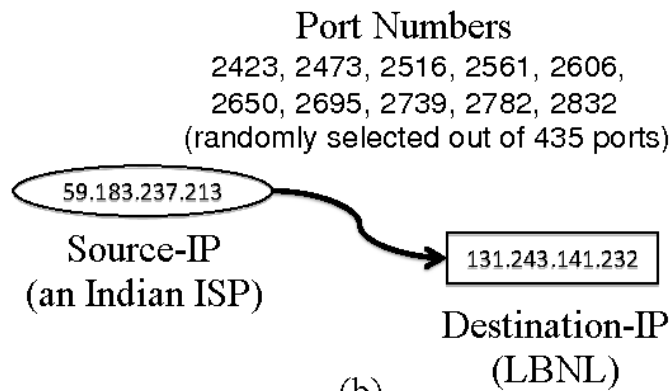
generalized bipartite-core

A subnetwork that represents a dense bipartite structure similar to the bipartite-core pattern in regular graphs, and is akin to association rules as well. More generally, it can be viewed as a continuous block along two dimensions in higher-order tensors under specific element orders. For instance, a group of source IP addresses sending packets to multiple destination servers with the same port. Note that in a heterogeneous network, this category of patterns also includes, written in the language of network forensics, multiple source IP addresses sending packets over different port numbers to the same server. This is likely to happen during a DDoS (Distributed Denial-of-Service) attack, a typical scenario of network intrusion, in which source IPs play the role of malicious hosts sending huge volumes of packets to the target server as the victim. Figure 3.5 provides an illustration of these patterns.

As a final remark, the statement that both patterns are related to a block along one or two dimensions in the high-order tensor only holds when elements of their respective data modes are ordered in specific ways. And the complexity to search for such an order is generally exponential, which reflects, in some sense, the power of the proposed approach.



(a)



(b)

Figure 3.4: Examples of generalized star patterns discovered in the LBNL (Lawrence Berkeley National Lab) network traffic data set. Wavy arrows indicate multiple edges between the pair of nodes with a handful of distinct attribute values. (a) 10 source IP addresses (randomly selected out of 172 ones) are sending multiple packets to a server machine with Port# 534, which is a UDP port under the NCP protocol from a network OS for file sharing and printing services; (b) The source IP registered by an Indian ISP is sending packets to a host in LBNL via port numbers (ranging from 2,300 to 2,900) not usually intended for this type of communication, implying a suspicious activity.

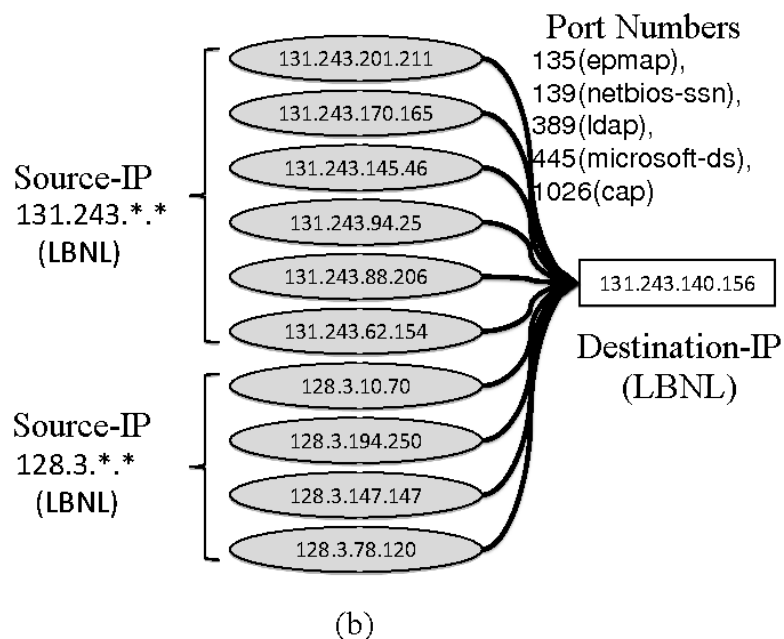
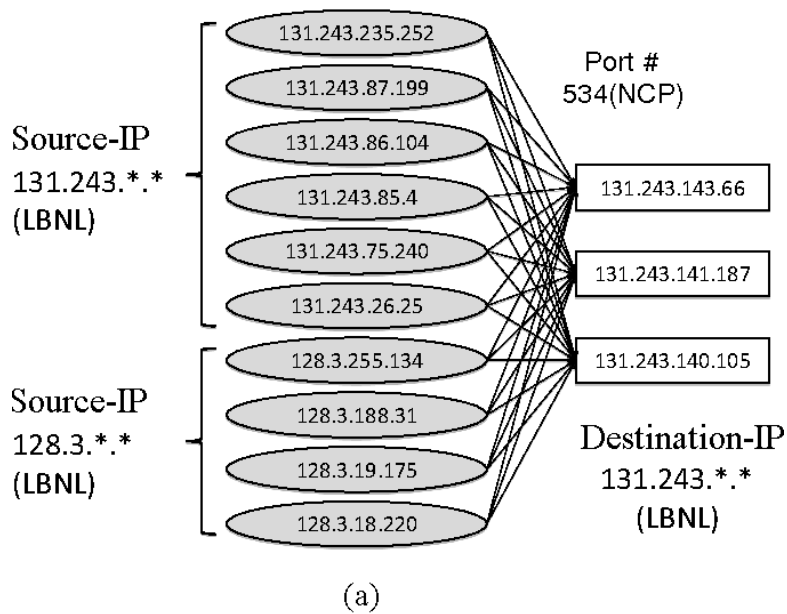


Figure 3.5: Examples of generalized bipartite-core patterns discovered in the LBNL (Lawrence Berkeley National Lab) network traffic data set. Wavy arrows indicate multiple edges between the pair of nodes with a handful of distinct attribute values. (a) 10 source IP addresses (randomly selected out of 119 ones) are sending multiple packets to an array of server machines, including server 131.243.141.187, which also appears in Figure 3.4 as part of a generalized star pattern, over a port used for file sharing and printing services; (b) 10 source IP addresses (randomly selected out of 63 ones) are sending packets over different ports to a multi-purpose server machine.

Algorithm 2 *SDA* (Spike Detection Algorithm)

Input: Eigenscore histogram vector H of size N **Output:** The set indicating spikes detected S

```
1:  $S = \phi$ 
2: sort the histogram to obtain an ordered vector  $H_o$  s.t.  $H_{o_1} \geq H_{o_2} \geq \dots \geq H_{o_N}$ 
3:  $Q_{SUM} \leftarrow \sum_{n=1}^N H_n^2$ 
4:  $Q \leftarrow 0$ 
5: for  $k = 1, \dots, K$  do
6:    $S \leftarrow S \cup \{o_k\}$ 
7:    $Q \leftarrow Q + H_{o_k}^2$ 
8:   if  $Q/Q_{SUM} \geq s$  and  $H(o_k)/H(o_1) < r$  then
9:     break
10:  end if
11: end for
12: return  $S$ 
```

Data set	# modes	Dimensions	Measure	# non-zero elements
LBNL	4	2,345 source IPs, 2,355 dest IPs, 6,055 port #'s, 3,610 timestamps	# packets	281K
RTW	3	3,641 subjects, 3,929 objects, 98 verbs	binary	10K
BDGP	3	4,491 genes, 248 terms, 6 stages	binary	38K

Table 3.1: A Summary of Data Sets

3.3 Empirical Results

We commence this section with the description of data sets as well as experimental environment. It is followed by the discussion of respective patterns discovered by *MultiAspectForensics* in each of the three data sets.

3.3.1 Data and Environment

Data sets are acquired from three dissimilar application domains: network traffic monitoring, knowledge networks, and bioinformatics. A summary is highlighted in Table 3.1.

LBNL The network traffic log is made available through a research effort to study the characteristics of traffic for Internet enterprises [86]. The measurement was taken on servers within the Lawrence Berkeley National Lab (LBNL) from thousands of internal hosts over time, with millions of packet traces recorded. Each packet trace includes four data modes:

source IP, destination IP, port number, and a timestamp in second. For privacy reasons, lower 16 bits were randomly permuted to anonymize the host identity, whereas upper 16 bits were kept intact for proper identification of the location and service provider [87]. We borrowed a subset of this data set within 1-hour time span in this section.

RTW This online knowledge base is the outcome of the NELL (Never-Ending Language Learning) system at Carnegie Mellon University [20]. It employs natural language processing and machine learning techniques to constantly and automatically crawl web pages and extract facts [21]. Each fact is a triplet of (subject, verb, object) such as (*pittsburgh*, *city-located-in-state*, *pennsylvania*), which could be represented as a directed graph made up of entities like *pittsburgh* or *pennsylvania*, edges with attributes like *city-located-in-state*. For better quality of results, we applied our algorithm on a preprocessed subset after manual noise removal (by courtesy of Bryan Kisiel at Carnegie Mellon University).

BDGP The data set is collected from the Berkeley Drosophila Genome Project (BDGP) to study the spatial-temporal patterns of gene expression during the early development of fruit fly [106, 107]. We selected three data modes from the database dump available at [13], which consists of 4,491 genes, 248 functional annotation terms from a specialized vocabulary, and 6 different developmental stages.

MultiAspectForensics was implemented in the MATLAB language, and all following experiments were performed on a Unix machine with four 2.8GHz cores, and 16GB memories. For every of these data sets, the wall-clock time was no more than 2 minutes to carry out the computation and generate attribute plots and histogram plots along all modes.

3.3.2 LBNL Traffic Log

We have already discussed patterns discovered from a snapshot of this data set in Section 3.2.3, illustrated in Figures 3.4, 3.5. With the additional mode of timestamp, we found two dominating spikes in its histogram plot. Upon closer examination, we reported the following activities: the first spike is a generalized bipartite-core pattern related to the HTTP traffic on port 80 between four servers in LBNL and three remote hosts in Chinese academic institutions, possibly executing scripts to crawl/download web pages. The second spike represents a generalized star pattern between one of the local HTTP server and the same remote host at India aforementioned. We traced further in time and found that the remote host never sent packets back to acknowledge the connection, suggestive of suspicious activities to be reported to domain experts.

3.3.3 RTW Knowledge Base

Recall that each item in the knowledge database could be represented as a (subject, verb, object) triplet. *MultiAspectForensics* detected spikes mostly on data modes representing subjects and objects.

Figure 3.6(a) illustrates a subgraph discovered revealing a generalized star pattern. The music artists/bands listed here are specialized to punk music or its sub-genres (not shown in the

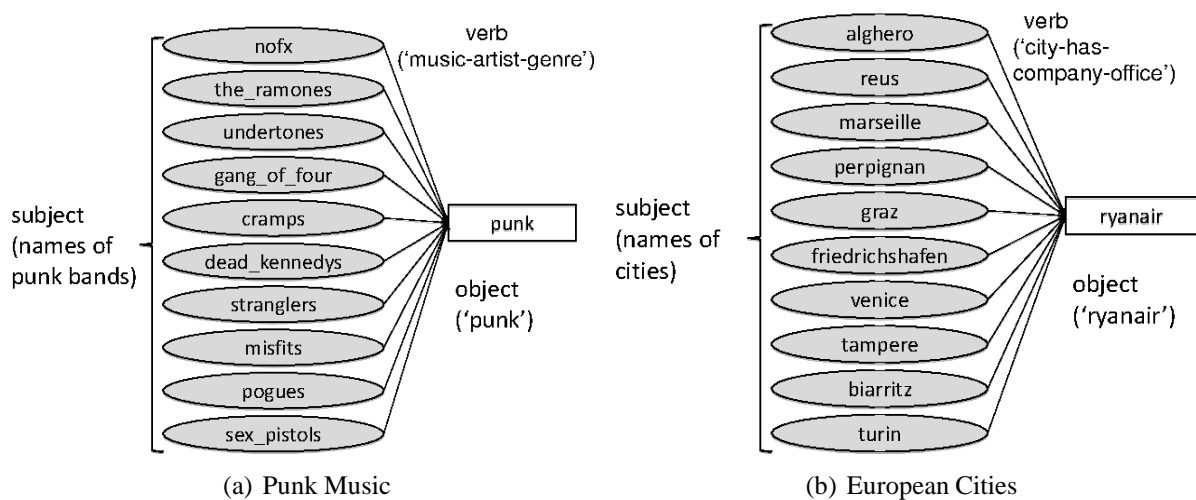


Figure 3.6: Two generalized star patterns discovered from the RTW knowledge base: (a) Music artists specialized in punk or one of its sub-genres according to the knowledge base; (b) European destinations of the Ryanair, an Irish low-cost airline.

figure) according to the knowledge base, whereas their more versatile peers will not be favorably selected by *MultiAspectForensics*.

Figure 3.6(b) displays another generalized star pattern between European cities and an Irish low-cost airline which flies to many regional or secondary airports to reduce cost, following a different business model and choice of destination from industrial giants.

The evidence here and many others alike could also be leveraged in a variety of graph mining tasks on this knowledge base such as clustering entities or creating an ontology between them, given the fact that nodes within the same spike tend to behave similarly and specifically. Moreover, as a sanity check, since node names are ordered alphabetically in this data set, the pattern does not make a continuous block in the tensor without non-trivial permutation.

3.3.4 BDGP Gene Annotation

In this data set *MultiAspectForensics* spots a set of genes known to be responsible for the *maternal effect* in the early development of fruit fly (Figure 3.7), which also provides hints to study other higher organisms including *Homo sapiens*. Products of such maternal effect genes, in the form of either protein or mRNA, play a critical role in the very early stage of embryo development, such as the first few cell divisions. For instance, four of such genes, including *bicoid*, *caudal*, *hunchback*, and *nanos*, is mostly responsible for the determination of anterior-posterior axis – which side of the embryo will be the future head and which other side will be the future tail [68].

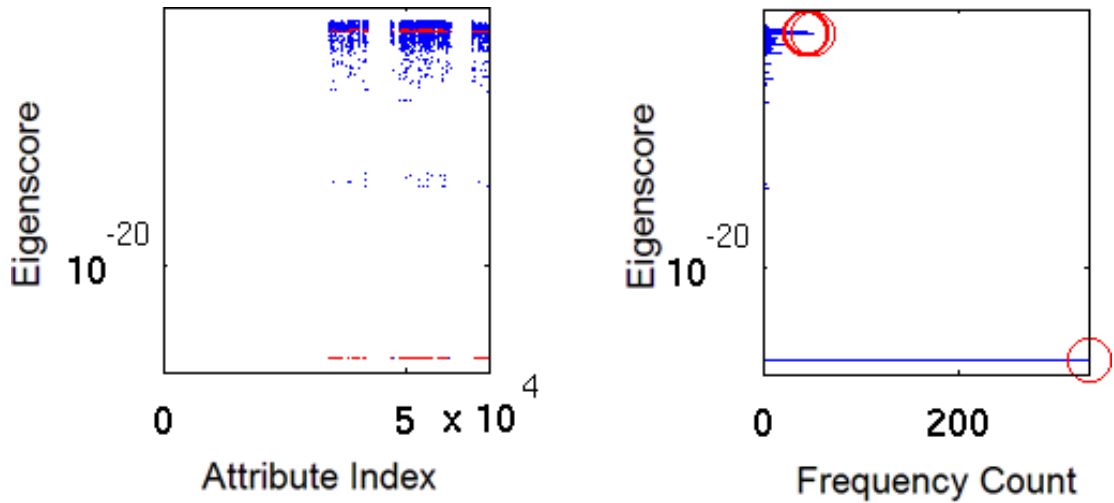


Figure 3.7: An attribute plot on the left side-by-side with the corresponding histogram plot for the “gene” mode of the BDGP data set. The largest spike that appeared at the bottom is the set of *maternal genes*, a special class of genes that play a vital role in early embryo development such as the polarity of the egg, *i.e.*, which part will become the head and which other part turns into the tail later.

3.4 Related Work

3.4.1 Anomaly Detection

Outlier detection, despite its wide interest across many application domains, is usually a challenging problem, as reflected in the fact that even a formal definition is not easy to make. A classical one was given by Hawkins in [54]: “an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism”.

Outlier detection methods can be categorized into two sets: parametric, statistical-based approaches, and non-parametric, model-free approaches. A common characteristic of methods in the former category is the existence of statistical assumptions about the underlying data distribution [11]. The latter category usually makes the call by resorting to distance computation [63] or density estimation [19, 58]. Besides, projection-based methods [2] have been introduced for high-dimensional data. Moreover, clustering algorithms may output outlier labels as a by-product (*e.g.*, [26]).

Compared to outlier detection, anomaly detection in structured data has only gained recent attention [25], where we have reviewed relevant studies in the introductory section and claimed that there is no other attempt, to the extent of our knowledge, to discover similar patterns in heterogeneous network data as *MultiAspectForensics*.

3.4.2 Tensor Analysis and Graph Mining

Tensor decomposition has been a basic technique well studied and applied to a wide range of disciplines and scenarios. An informative survey on tensor decompositions is presented by Kolda and Bader [64] with many further references. Recent researches have further generalized the CP decomposition to handle incomplete data [1], or to produce non-negative components [97]. Tucker decomposition, as the other well-known approach, is more flexible, although its application is usually limited by its limited scalability and vulnerability to noise. Notably, recent work on scalable alternatives such as [111] may open up the venue to enhance the *MultiAspectForensics* methodology with more powerful decomposition algorithms.

Quite a few popular implementations of tensor decomposition algorithms for academic researchers have been made publicly available. Examples are the N-way toolbox by Andersson and Bro [7] and the more recent MATLAB Tensor Toolbox by Bader and Kolda [9].

Tensor analysis has also been applied to study the dynamics of graphs and networks [104]. They commonly start by analyzing graph/tensor snapshots within each timestamp, and take the output for subsequent time-series analysis. *MultiAspectForensics*, instead of focusing on the evolution between adjacent timestamps, treats timestamp as another data mode to allow better discovery of global patterns in this trade-off.

3.5 Conclusion

We presented *MultiAspectForensics*, a handy and effective tool to automatically detect and visualize a category of novel patterns, including generalized star and generalized bipartite-core patterns, within a local community of nodes in heterogeneous networks, even if they exist among less-well connected nodes which are more likely to be ignored by many extant methods. Empirical results exhibited valuable insights derived from pattern discovered, across multiple application domains such as network traffic monitoring, knowledge networks, and bioinformatics. These successes could be attributed to the fact that we resorted to a tensor-based representation to facilitate data decomposition, reached a key observation leading to spike patterns in histogram plots, and revealed typical substructures reflecting spectral properties of heterogeneous data. Hence *MultiAspectForensics* realizes an early attempt to research substructure patterns commonly existing in heterogeneous network data, and a reasonable use case of tensor analysis, despite the simplicity of heuristics resided.

An important problem beyond the scope of this manuscript is the design of an objective and quantitative evaluation framework of discovered patterns, especially for large-scale networks for which it is prohibitive to label every interesting pattern. Although it may be relatively to define precision and recall by exhaustively searching for subgraphs bearing the specified pattern such as generalized star or generalized bipartite core, the definition of quality, or value of these patterns from automated discovery, is usually domain and context specific – even may not be losslessly quantified. This would also shed lights on a principle way of optimizing parameters, yet we found that results were usually not sensitive to parameter values when they vary within reasonable

ranges. Meanwhile, it's our plan to open-source the *MultiAspectForensics* tool based on the generic *boost graph library* [99] to make it more accessible and usable by industrial practitioners and academic researchers, and collect feedbacks for possible future developments.

Part III

Querying Multimedia Data

Chapter 4

CDEM: Flexible Querying System for Biological Image Databases

Given a large collection of images documenting the spatial patterns of gene activities on football-shaped embryos, can we perform content-based retrieval to find similar patterns for an existing or new input? Can we leverage this similarity to group together genes that display correlated patterns, which suggests a greater likelihood for them to participate in the same biological process in the development of the embryo? Can we construct a network of genes to visualize such relationships known as co-expression? Moreover, most of the genes bear a handful of labels, manually curated by domain experts using anatomical terms to indicate the body-part (*e.g.*, “embryonic hindgut”) with most significant expression activity, can we employ this additional semantic information to improve the retrieval results, or automatically assign such labels to new and upcoming images?

This chapter and the accompanying online query interface is an initial attempt to address these questions. Part of the work described subsequently is based on the material presented in [48].

4.1 Background

How an organism develops from a single cell, one of the great mysteries of life, has always been an intriguing problem for biologists. To uncover the genetic foundation of animal design, extensive *in vitro* and *in vivo* studies have been carried out to decode the early development of *Drosophila melanogaster*, or fruit fly, with the expectation that understanding gained in this organism model may apply to other species, not excluding human beings, as well. Thanks to the recent advancement in biomedical imaging technologies, it is now possible to make high-resolution image recording of 2D or even 3D spatial signals capturing gene expression in cells and living organisms. As a popular type of data characterizing complex biological systems, many of these images are digitally stored into multimedia databases and made publicly accessible via various online and offline interfaces for querying and browsing. For instance, the Berkeley Drosophila Genome Project (BDGP) [106, 107] provides an online database of two-dimensional


	Gene
	CG 15141
	Annotation Terms
	brain, central nervous system, glia, neurons, ventral nerve cords
	Developmental Stage
	13-16

Figure 4.1: A fruit fly embryo image with all its attributes sampled from the BDGP database. The original filename is `insitu65954.jpe`.

fruit fly embryo images, which includes three modes of data: more than 70,000 images of size up to 1520 by 1080 pixels, around 3,000 genes, and several hundred annotation terms. Figure 4.1 illustrates one image from the database together with all its attributes.

Every image is associated with a single gene, of which the expression is documented digitally. There are up to a few annotation terms from a controlled vocabulary as a textual description of the pattern, indicating parts of the body that have darker colors and more significant expressions. Image-based analysis is still indispensable since subtle patterns may not be captured by the vocabulary of limited size. Also, each image has a time stamp which is labeled using one of the six predefined developmental stage ranges. Patterns under different stages are not directly comparable due to the drastic morphological changes in the developmental process. This also leads to the difference in the set of annotation terms eligible for each stage ranges.

We present a general framework in this chapter on which a number of interesting tasks around this multi-modal data collection of genes, image patterns, and text annotations. For example,

- *Multi-modal Retrieval*: given one or more images/genes/terms, what are the most relevant images/genes/terms? This could assist biologists to find similarities with each data modal or perform cross-modal queries such as annotation term suggestions for images.
- *Multi-modal Clustering*: find groups of images/genes/terms that members are more closely linked to each other than with non-members. The hope is that each group may correspond to one or more biological functions so that subsequent biological analysis could be more focused on a smaller set of genes.
- *Network Construction*: draft a network between a subset of genes where links between genes represent spatial co-expression and may provide hint on meaningful biological interactions.

The basic idea of our approach is constructing a graph of multiple types of nodes representing different mode of data. Graph-based algorithms such as random walk with restart could be adapted, and the infrastructure and the algorithm employed are readily scalable to handle a relatively large volume of data. An online interface is made available accordingly to assist biologists to browse and navigate the data set.

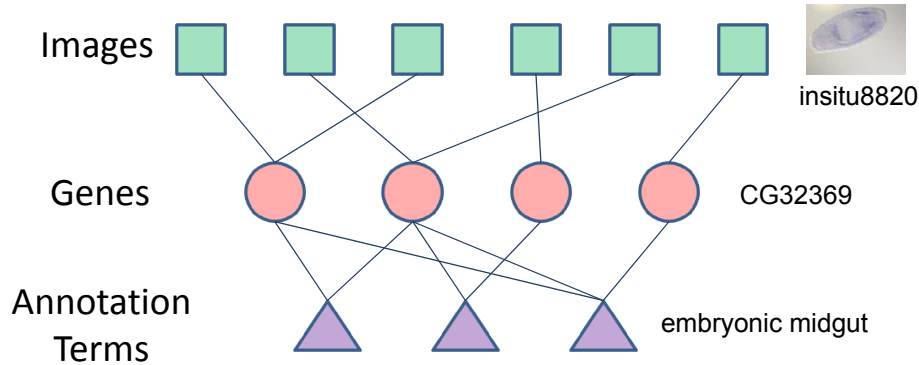


Figure 4.2: A tri-partite graph constructed from the BDGP database.

4.2 Proposed Method

4.2.1 Graph Construction

As a prerequisite for all the subsequent algorithmic development, we construct a heterogeneous graph which consists of multiple layers of nodes, each of which represents one of the data modes (genes, images, or annotation terms). The connection between different data modes are abstracted as edges between corresponding nodes. For instance, as illustrated in Figure 4.2, gene node CG32369 is connected to an image node *insitu8820* and a term node *embryonic midgut*. This indicates that the image with the label *insitu8820* documents the expression pattern for gene CG32369. And such spatial pattern, as well as those recorded in other images of the gene in the same developmental stage, could be (partially) located at the *embryonic midgut*, *i.e.*, the part of embryo from which most of the intestines are derived. Note that annotation nodes are connected nodes representing genes rather than image nodes because based on the data source available, given a particular developmental stage, images from the same gene always share the same set of annotation terms.

However, an important part of information is ignored by the aforementioned tripartite graph – content similarity between images, which is the essential pattern we would like to capture in the image-based analysis. Consequently, we propose to obtain a feature-based representation of expression images, and connect pairs of images that are close to each other in the feature space. Feature values are borrowed from the triangulated images in [42], which employed a number of image processing techniques to align embryos of different sizes, shapes, positions and orientations, as well as to remove certain imaging artifacts. And we find 3~10 nearest neighbors for each image node to create intra-layer edges.

4.2.2 Multi-Modal Retrieval

With the complete graph of images, genes and terms as well as links between them, we are ready to derive the proximity measure for the retrieval task. Here we employ random walk with restart (RWR) on graphs, which is also known as personalized Pagerank [83]. Given a query node i , the

Stages Included	13-16	11-16	9-16	7-16	4-16	1-16
# of Nodes	10,868	23,008	28,568	34,141	42,319	49,261
# of Edges	99,113	199,035	237,555	274,415	336,354	385,515
Avg Node Degree	9.2	8.7	8.3	8.0	7.9	7.8

Table 4.1: A summary of graphs constructed of different sizes.

proximity from i to every other node in the graph can be derived from the steady-state probability of the following discrete-time Markov process: at each time tick, the random walker makes one of two possible choices:

1. with probability $(1 - c)$, randomly pick one of the neighbors of the current node, and walk to that node,
2. with probability c , jumps back to the query node i ,

where c is known as the restart probability and empirically set to 0.1. If the graph has weighted edges, the chance of picking a random neighbor under the first choice is proportional to the weight of the connecting link. Denote the corresponding steady-state probability vector by \mathbf{r}_i , and the graph adjacency matrix by W , then we have

$$\mathbf{r}_i = (1 - c)W\mathbf{r}_i + c\mathbf{e}_i \quad (4.1)$$

where W_{jk} equals the probability of going from node k to node j under the first choice, and \mathbf{e}_i is a vector of which the i th element is 1 and other ones are 0. Eq. 4.1 can be solved directly to obtain $\mathbf{r}_i = c(I - (1 - c)W)^{-1}\mathbf{e}_i$. However, the cost matrix inversion would be prohibitive for a moderately large W . An iterative power method is applied instead, of which the complexity is linear to the number of edges for a sparse graph. More elaborate techniques such as [108] could be employed if scalability is concerned.

The RWR algorithm applies naturally to the multi-modal query setting, as relevance scores could be normalized within each layer of node respectively. It also generalizes smoothly to multiple query inputs, by letting \mathbf{e}_i have multiple non-zero entries, each of which corresponds to one of the candidate nodes under the random walker’s second choice.

4.3 Experimental Evaluation

To shed light on the scalability of the proposed approach, we create a total of 6 graphs of varying sizes by putting in additional data from other stage ranges. Statistics are summarized in Table 4.1. The database dump and feature representation of images were the same as [42] and downloaded from the BDGP website. Images are linked to each other in the feature space if their feature vectors have a correlation value greater than 0.7. The number of nearest neighbors linked to each image is constrained to be between 3 and 10.

We measure the elapsed time of the graph construction algorithm on a Linux machine with 2.8GHz cores with MATLAB 2009b installed. Figure 4.3(a) plots the average number over 10 repetitions against the number of nodes in the graph, whereas Figure 4.3(b) reports the running

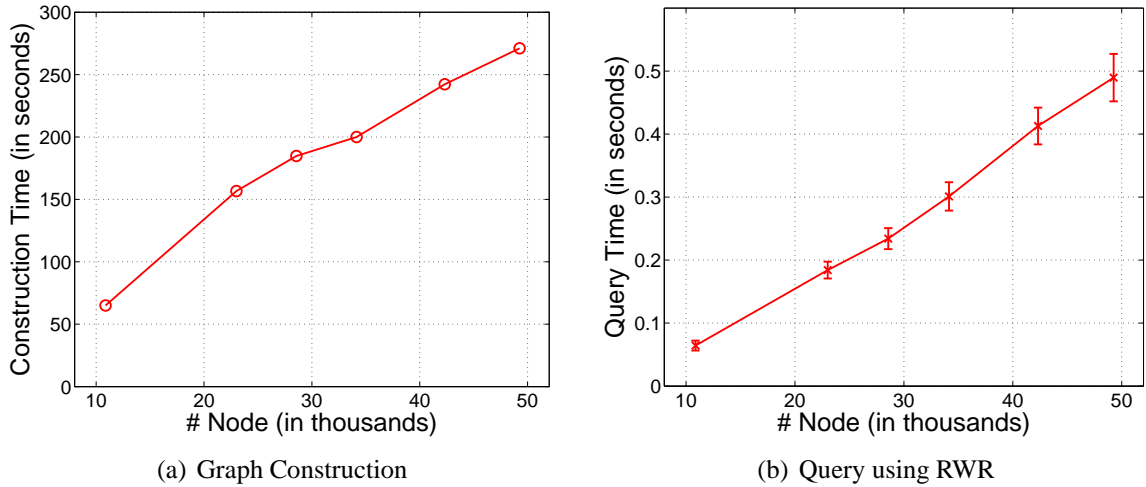


Figure 4.3: Running time based on different developmental stage ranges (a) for constructing the graphical representation, averaged over 10 repetitions; and (b) for one query using RWR, averaged over 100 random queries, with error bars of one standard deviation.

Rank	Image Results	Genes (Synonyms)	Annotation Terms
1	insitu67039 (Fig. 4.4(a))	CG10498 (<i>cdc2c</i>)	ventral nerve cord
2	insitu64954 (Fig. 4.4(b))	CG15141	brain
3	insitu28800 (Fig. 4.4(c))	CG5581 (<i>Ote</i>)	central nervous system
4	insitu67041 (Fig. 4.4(d))	CG10212 (<i>SMC2</i>)	midgut
5	insitu35317 (Fig. 4.4(e))	CG1245 (<i>MED27</i>)	neurons

Table 4.2: C-DEM query results using the query image shown in Figure 4.4(a).

time for executing query from a random node of graphs constructed. Both curves show a linear trend as graph sizes go up, and for the largest graph containing almost 50,000 nodes, the time needed for graph construction and querying are no more than 5 minutes and 0.5 seconds, respectively.

Table 4.2 and Figure 4.4 provide top five query results for each modal using an image with visible expression patterns in the anterior (near the head) and ventral (near the belly) part of the embryo as the query input. All five images display similar patterns to the query. The corresponding gene for each of these images is also in the list of most relevant genes. Three of the top five genes (*cdc2c*, *Ote*, *SMC2*) have been identified as mitotic genes, which is related to the mitosis (M) phase in the cell-division cycle, in an independent study [102]. Relevance scores for top annotation terms are 0.15, 0.15, 0.12, 0.07 and 0.04, respectively. Top two terms (ventral nerve cord and brain) are part of the annotation of every image in the top-five list, whereas the third term (central nervous system) is shared by all the images but *insitu28800*. This may lead to a specific annotation suggestion of central nervous system for the image *insitu2880*. And we would like to automate this task in our future work.

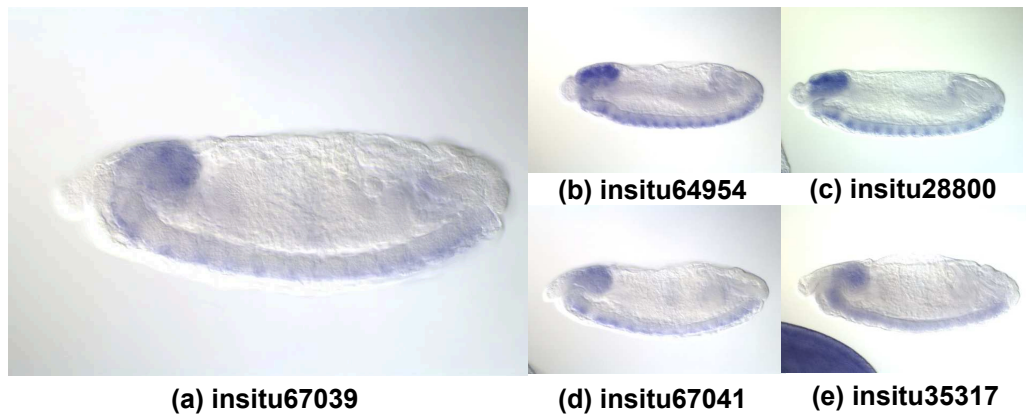
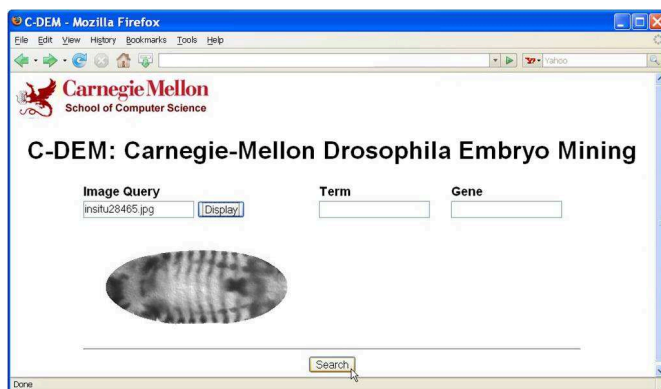
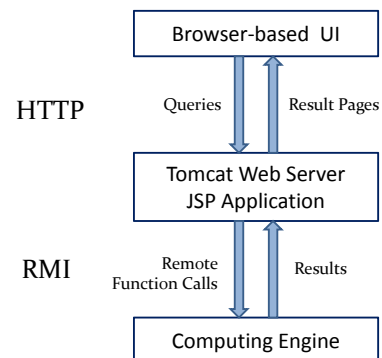


Figure 4.4: A typical query result using an embryo image in (a) as the query input. Top 4 similar images other than the query image itself are displayed in (b)-(e).



(a) Online Interface



(b) System Architecture

Figure 4.5: C-DEM: an online, multi-modal query system for Drosophila embryo databases. Images are adapted from [48].

Figure 4.5(a) illustrates the online interface of the C-DEM query system. The query input could be an image, and/or a gene, and/or an annotation term. The software architecture of C-DEM in Figure 4.5(b) de-associates the front-end web server and back-end computing engine with a clear and stable API. They are deployed on separate machines for better performance by distributing the workload. Detailed discussion on how to query and browse the database using C-DEM is given in [48].

4.4 Related Work

4.4.1 Automatic Analysis of Embryo Images

A first analysis of the data set came from [106, 107] by the BDGP group: [106] performed co-clustering of the gene-annotation matrix; [107] incorporated the microarray data for gene clustering and applied a fuzzy clustering algorithm which allowed a gene to belong to multiple clusters. Both of them only made indirect use of image data through the manual annotation results. Moreover, [107] provided a network representation of collapsed annotation terms which reflect tissue relatedness.

[119] developed algorithms to determine the stage of an image and perform automatic annotation. Since an image may have multiple annotation terms, annotation was treated as a series of simple bi-class classification problems. Image features were obtained using 2D wavelet discrete transform and a feature selection algorithm from previous work [89]. The same author introduced in [90] additional features based on Gaussian mixture model (GMM) and principle component analysis (PCA) to characterize local and global patterns. Their proposed algorithm performed very well in the task of automatically finding the developmental stage given an expression image (>99% accuracy), however, automatic annotation turned out to be a much more challenging task, and even finding an intuitive, objective evaluation scheme seems to be nontrivial.

[52] argued that the pure visual feature based retrieval method cannot be applied to find correspondence between images in different developmental stages. The correspondence should be established through the annotation terms which are from the same controlled vocabulary independent of developmental stages. A max-margin based algorithm designed for multi-modal data mining was applied to obtain empirical evaluation on the BDGP database. The algorithm outperformed another state-of-the-art multi-modal mining algorithm in precision-recall for most of the retrieval tasks evaluated, and scaled well with the size of the dataset. However, more biological evidence need to be provided to support this special treatment of “across-stage retrieval”, and this framework may limit some global image features.

The FEMine system [85] derived two sets of features by applying PCA and ICA (Independent Component Analysis) respectively, and performed classification, clustering and retrieval tasks based on these features. [85] provided a detailed discussion on image preprocessing, which will be adapted as an important component in the current project. The major concern comes from the experimental evaluation where images were hand-picked from the BDGP dataset and the size of the experimental data need to be significantly.

Random walk and related methods have many successful applications, of which the most well known one is PageRank [83]. [84] applied random walk with restart to a simple automatic caption setting.

4.4.2 Online Databases

The database created by BDGP provides a query interface where users provide gene name, developmental stage, and/or annotation information and search engine returns corresponding images.

The `FlyExpress` database [65] offers a content based image retrieval function named “find similar patterns”, where users first choose an extracted pattern of an image from a small set of candidates, then the search engine returns a list of images with similarity score. Both web sites refer/link to the well known `Flybase` [112] for detailed information about gene function, synonym etc.

4.5 Conclusion

We presented an online interface to assist biological researchers to perform flexible querying and exploration over a large database which consists of embryo images, image annotations, as well as genes whose expression patterns are illustrated by these images. Given an input query from any data modal, image or text, the system could automatically and efficiently search over the entire database and output an ordered list of most similar images or formatted attributes. The underlying proximity measure is derived via random walk with restart algorithm over graphs.

Chapter 5

***BEFH*: Bayesian Exponential Family Harmoniums for Multimedia Retrieval**

The vast size of the text and multimedia information available from digital libraries and World Wide Web, and large amount of knowledge contained therein, creates a need to organize and summarize topical contents of these data. In recent years, there is a growing volume of research on applying probabilistic graphical models to develop automatic information distillation systems that can explore and exploit real-world data from diverse sources, such as texts, images and biological sequences. This chapter presents a Bayesian approach to inference and learning with the recently proposed exponential family harmonium (EFH) models and their variants for posterior latent semantic projection of multimedia documents for subsequent data mining tasks such as classification and retrieval.

We first provide a table listing common acronyms in this chapter for reference.

Table 5.1: Summary of Acronyms

Acronym	Explanation
EFH	Exponential Family Harmonium, a family of undirected graphical model
BEFH	Bayesian extension of EFH
GB-EFH	A special case of EFH with variable distributed in binary/Gaussian
DWH	Dual-Wing Harmonium, a generalization of EFH for multi-modal data
LSI	Latent Semantic Indexing, a classical method for topic discovery
pLSI	Probabilistic Latent Semantic Indexing, a classic probabilistic topic model
LDA	Latent Dirichlet Allocation, a generative Bayesian graphical model for topic discovery

5.1 Introduction

Probabilistic graphical models provide a compact description of complex stochastic relationships among random variables, which can correspond to both perceivable entities (*e.g.*, words,

imageries) and abstract concepts (*e.g.*, topics, themes). Such a formalism often facilitates flexible statistical reasoning and query answering based on appropriate computational algorithms. Inspired by the classical approach of *latent semantic indexing* [34], at the beginning of this century there have been important advances in developing latent semantic graphical models for large text corpora and multimedia data, based on either a Bayesian network or a Markov random field (MRF) formalism. For instance, the *probabilistic latent semantic indexing* (pLSI) [56] method models each document as an admixture of topic-specific distributions of words. The more recent *latent Dirichlet allocation* (LDA) technique [18] employs a hierarchical Bayesian extension of pLSI, treating both the document-specific topic-mixing coefficients and the topic-specific word probabilities as random variables, under appropriate conjugate priors. LDA can be extended to multimedia collections by assuming that the unobserved “topics” are correlated with both image variables and word variables [10, 16]. Recently, Welling et al. [113] proposed another class of latent semantic graphical models known as the exponential family harmonium model (EFH), which can be understood as an undirected, and non-Bayesian counterpart of the LDA model. Subsequently, [114] extended EFH to a *dual-wing harmonium model* (DWH) for joint modeling of text and image. Also, Gehler et al. [43] proposed the *rate adapting Poisson* (RAP) model which follows the general architecture of EFH model and use conditional Poisson distributions to model observed count data. And McCallum et al. [78] proposed a training criterion called *multiple-conditional learning* (MCL) for MRFs and EFHs. Unlike the directed graphical models such as pLSI and LDA, EFH does not employ auxiliary latent variables (*i.e.*, the imaginary topic indicators for every word) to facilitate topic mixing and simulate data generation; and it allows a more flexible representation of the latent topic aspects for documents (*i.e.*, as a point is a Euclidean space rather than in a simplex).

An important advantage of the directed latent-topic models such as LDA is that they can be straightforwardly embedded in a Bayesian framework, and can undergo Bayesian training, smoothing and inference. To date, the MRF-based models such as EFH and DWH have been largely limited to a maximum likelihood (ML) learning framework, which is prone to undesirable effects such as overfitting over a relatively smaller data set, high variance in sampling-based inference and parameter estimation, and indifference to prior knowledge. These limitations restrict their utilities in many realistic data mining scenarios where data are sparse and spurious. The ML framework also makes it difficult to fully exploit the modeling power of MRF in latent topic distillations and to develop future extensions. The unavailability of a Bayesian version of EFH is partly due to the remarkable technical difficulties one must overcome when working under such a formalism. It is well-known that statistical learning of EFH models from data, even under an ML framework, is technically non-trivial. As discussed in [81] and [91], Bayesian learning for general MRF, is even more challenging, particularly in cases that involve latent variables as in EFH. In this chapter, we attempt to address some of this challenges: endowing EFH with a simple Bayesian prior, and presenting a sampling-based algorithm for Bayesian inference and learning.

In summary, we present Bayesian EFH (BEFH), in which a multivariate Gaussian prior is introduced for the weight matrix that couples the latent topics with observed attributes in EFH

(and also in DWH). As detailed subsequently, it is illuminative to view the weight matrix of EFH as the matrix of word probabilities under all topics in LDA. Under this analogy, our prior corresponds to the Dirichlet priors for the word probabilities in LDA. It is well-known that methods for Bayesian inference and learning in directed graphical models such as LDA does not apply to the undirected graphical models concerned here, because of the intractability and non-conjugacy arising from the generally intractable partition function. In this chapter, we present the Langevin algorithm conjoint with a MCMC sampling scheme for posterior inference under BEFH. We also propose an empirical Bayes method based on the Langevin algorithm for unsupervised estimation of the BEFH hyper-parameter given training data. Finally we show comparisons of ML and Bayesian approaches on a synthetic dataset with known parameters and a dataset provided by TRECVID 2003 [101] with both text and image data.

5.2 Bayesian EFH

In this section, we outline the basic structure of a Bayesian EFH in the context of a simple instantiation of EFH for latent topic modeling of text corpora.

Prior to delving into technical discussion, we provide a summarization of symbols to be referred multiple times in Table 5.2.

We commence the discussion with a brief recap of the basic EFH, as described in [113]. Consider an undirected graphical model defined on a complete bipartite graph containing two layers of nodes (Fig 5.1). Let $\mathbf{H} = \{H_j\}$ denote the set of *hidden units* in such a graph, and let $\mathbf{X} = \{X_i\}$ denote the set of *input units*. An EFH defines the following Markov random field:

$$p(\mathbf{x}, \mathbf{h}) \propto \frac{1}{Z} \exp \left\{ \sum_{ia} \theta_{ia} f_{ia}(x_i) + \sum_{jb} \lambda_{jb} g_{jb}(h_j) + \sum_{ijab} W_{ia}^{jb} f_{ia}(x_i) g_{jb}(h_j) \right\}, \quad (5.1)$$

where $\{f_{ia}(\cdot) : \forall a\}$ denotes the set of potential functions (or features) defined on each of the input units (indexed by i) in the model, and likewise $\{g_{jb}(\cdot) : \forall b\}$ for the hidden units; $\Theta = \{\theta_{ia}\} \cup \{\lambda_{jb}\} \cup \{W_{ia}^{jb}\}$ denotes the “weights” of the corresponding potentials or potential pairs; and Z stands for the partition function, which is a function of Θ .

The bipartite topology of the harmonium graph suggests that nodes within the same layer are conditionally independent given all nodes of the opposite layer. Specifically, from Eq. 5.1, we have the following factored form for the between-layer conditional distribution functions: $p(\mathbf{x}|\mathbf{h}) = \prod_i p(x_i|\mathbf{h})$, $p(\mathbf{h}|\mathbf{x}) = \prod_j p(h_j|\mathbf{x})$, and each of the singleton conditional has a simple exponential family form:

$$p(x_i|\mathbf{h}) = \exp \left\{ \sum_a \hat{\theta}_{ia} f_{ia}(x_i) - A_i(\{\hat{\theta}_{ia}\}) \right\}, \quad (5.2)$$

$$p(h_j|\mathbf{x}) = \exp \left\{ \sum_b \hat{\lambda}_{jb} g_{jb}(h_j) - B_j(\{\hat{\lambda}_{jb}\}) \right\}, \quad (5.3)$$

where $A_i(\cdot)$ and $B_j(\cdot)$ denote the respective log-partition functions; and the “shifted” parameters

Symbol	Definition
\mathbf{X}	observed units (random variables) in the harmonium model
X_i	the i th observed unit
\mathbf{x}	a vector of observed values as an instantiation of \mathbf{X}
x_i	the i th entry of \mathbf{x}
I	the total number of observed units
\mathbf{H}	hidden units (random variables) in the harmonium model
H_j	the j th hidden unit representing topics
\mathbf{h}	a vector of hidden values as an instantiation of \mathbf{H}
h_j	the j th entry of \mathbf{h}
J	the total number of hidden units
f_{ia}	the a th potential function associated with the i th observed unit
θ_{ia}	the parameter associated with f_{ia}
$\boldsymbol{\theta}$	the vector of such parameters when each observed unit has a single potential function
g_{jb}	the b th potential function associated with the j th hidden unit
λ_{jb}	the parameter associated with g_{jb}
$\boldsymbol{\lambda}$	the vector of such parameters when each hidden unit has a single potential function
W_{ia}^{jb}	the a, b th potential function associated with the i th observed unit and j th hidden unit
\mathbf{W}	the matrix of such parameters when each unit has only one potential function
$\boldsymbol{\mu}$	columns of \mathbf{W} follows <i>iid</i> multivariate Gaussian of dim- I ; this is the mean vector
$\boldsymbol{\theta}^2$	the vector of non-zeros elements in the diagonal covariance matrix
μ_i, σ_i^2	the mean and variance parameter for W_{i1}, \dots, W_{ij}
Z	the intractable partition function
ϵ^2	the discretization size of the Langevin algorithm in Section 5.3.2
N	the size of the data set, or the number of independent observations available
m	# samples obtained by repeatedly doing brief sampling as described in Section 5.3.3
V	a $I \times I$ matrix equivalent to WW^T

Table 5.2: Symbol table

$\hat{\theta}_{ia}$ and $\hat{\lambda}_{jb}$ are defined as:

$$\hat{\theta}_{ia} = \theta_{ia} + \sum_{jb} W_{ia}^{jb} g_{jb}(h_j),$$

$$\hat{\lambda}_{jb} = \lambda_{jb} + \sum_{ia} W_{ia}^{jb} f_{ia}(x_i),$$

where the shifts, *i.e.*, the second term in each of the equation above, are induced by the total couplings between units in the input and hidden layers. As seen from the above definition, since all the parameters in the joint distribution under EFH can be identified from the local conditional distributions, one can determine an EFH *using a bottom-up strategy* by directly specifying the often easily comprehensible local conditionals. For instance, as our running example in this

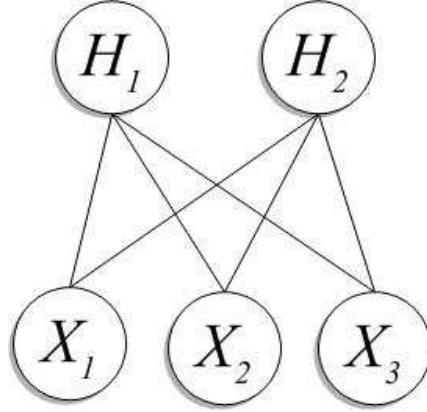


Figure 5.1: The graphical model representation for a harmonium with 3 input units (*e.g.*, binary word occurrences in a document) and 2 hidden units (*e.g.*, projection in a 2-dim topic space).

chapter, we define the following Gaussian-Bernoulli EFH (GB-EFH) for text:

$$p(x_i|\mathbf{h}) = \text{Bernoulli}(x_i|\text{logit}(\theta_i + \sum_j h_j W_{ij})), \quad (5.4)$$

$$p(h_j|\mathbf{x}) = \mathcal{N}(h_j|\sum_i x_i W_{ij}, 1), \quad (5.5)$$

where $\text{logit}(\alpha) = (1 + e^{-\alpha})^{-1}$ is the logistic function, and the shift of the logit-transformed Bernoulli rate θ_i is induced by a weighted combination of the latent units \mathbf{h} . It can be shown that under this construction, we obtain an EFH with the joint:

$$p(\mathbf{x}, \mathbf{h}) \propto \exp \left\{ \boldsymbol{\theta}^T \mathbf{x} - \frac{1}{2} \mathbf{h}^T \mathbf{h} + \mathbf{x}^T \mathbf{W} \mathbf{h} \right\}. \quad (5.6)$$

The GB-EFH models text (represented by variables \mathbf{x}) as binary occurrences of words, which is suitable for sparse, short text such as video captions. When modeling long articles, one may want to directly model word counts; and in this case one can replace Eq. 5.4 with, *e.g.*, a Binomial distribution. It is interesting to examine side-by-side the GB-EFH and the LDA model as displayed in Fig. 5.2. Note that, when treating each hidden unit h_j as a representative of a latent topic aspect, Eq. 5.4 can be understood as a likelihood function of an observed attribute, such as a word occurrence, induced by a combination of topics. Thus, the coupling matrix $\mathbf{W} = \{W_1, \dots, W_J\}$ in GB-EFH is reminiscent of the word probability matrix $\mathbf{B} = \{\beta_1, \dots, \beta_J\}$ in LDA, where β_j denotes the M -dimensional vector (M denotes the size of the vocabulary) of multinomial word probabilities under topic j . In GB-EFH each M -vector W_j represents the set of “contributions” topic j as on each word in a vocabulary. Although structurally similar, it is noteworthy that the topic mixing mechanism of GB-EFH is very different from that of the LDA model. In LDA the topic mixing is achieved by marginalizing out the auxiliary topic indicator variables for each word occurrence – as illustrated in Fig. 5.2(b), the LDA likelihood of a word x_w , given topic mixing coefficient θ and the probabilities of this word under all J topics, $\{\beta_{w,1}, \dots, \beta_{w,J}\}$ can be

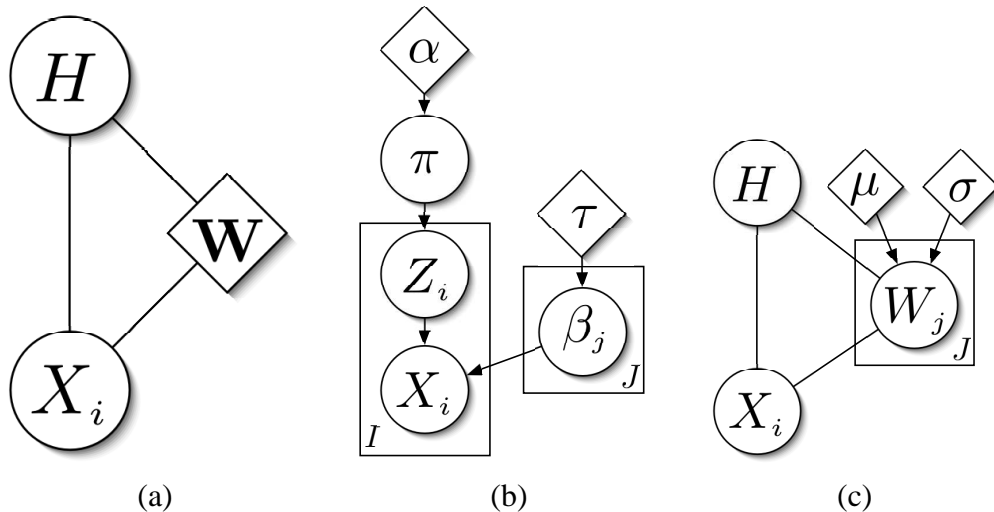


Figure 5.2: A comparison of EFH, LDA and BEFH models over a single document. Circles represent variables, and diamond represents model parameters. (a) EFH. For easy comparison, the hidden unit (*i.e.*, the topic weight coefficients) $\{H_j\}$ and the input units $\{X_i\}$ are represented as vector valued variables H and X , respectively. For simplicity, only the \mathbf{W} parameter of EFH is explicitly shown. (b) LDA. Note the correspondence between π in LDA and H in EFH, and the fact that β_j 's are random variables rather than parameters. I denotes the length of the document. (c) BEFH. Note that $\mathbf{W} \equiv \{W_j\}$ are now lifted as random variables.

written as $p(x_w|\theta) = \sum_z p(z|\theta)p(x_w|\mathbf{B}, z) = \sum_j \theta_j \beta_{w,j}$ – whereas it can be shown that in EFH the expected rates of all words are directly determined by the weighted sum of topic specific contributions $\sum_j h_j W_j \equiv \mathbf{W}h$. In this regard EFH is closer to the classical LSI principle in which the observed rates of all words can be expressed as a weighted combinations of the eigen-topics (*i.e.*, orthonormal topic-specific word rate vectors).

Empirically, it was noted that the performance of EFH and variants on latent semantic modeling is comparable, and sometimes superior, to LDA [113, 114]. But as shown in Fig. 5.2, structurally EFH is not yet a full undirected counterpart of LDA, which employs an elegant hierarchical strategy to incorporate priors for both the word probabilities \mathbf{B} and topic mixing coefficients π . We expect that, as is the case for LDA, it is possible for EFH to also leverage on the possible extra modeling power endowed by a Bayesian formalism.

Now we propose a Bayesian EFH that exploits the proclaimed benefits. To maintain exchangeability between hidden units $\{h_j\}$, we place column-*iid* prior on \mathbf{W} , that is, each column of \mathbf{W} follows a multivariate Gaussian, which is a common choice for modeling continuous parameters without any additional assumption:

$$p(\mathbf{W}) = \prod_{j=1}^J p(W_j) = \prod_{j=1}^J \mathcal{N}(W_j|\mu, \Sigma). \quad (5.7)$$

A full covariance matrix in the above prior would have size M^2 , which is prohibitively expensive for modeling large vocabulary. For simplicity, we consider a further simplification where: $\Sigma =$

$\text{diag}(\boldsymbol{\sigma})$, i.e. $\Sigma_{ij} = \sigma_i \sigma_j \delta(i, j)$. This means that each element of \mathbf{W} follows an independent normal distribution. Note that although we omit correlations between the topic-word coupling coefficients, the expressiveness of this prior is comparable to the Dirichlet prior for columns in the \mathbf{B} matrix of LDA, which captures little correlation behavior of the word-probabilities sampled from a simplex.

Now we are left with two remaining sets of parameters of EFH: θ and λ . It turns out that in many practical settings (e.g., GB-EFH and DWH), λ is vacuous, i.e., $\lambda = 0$, which essentially “centers” the conditional distribution $p(\mathbf{h}|\mathbf{x})$ at the shifts induced by the input units. For θ , in EFH it lacks an intuitive semantics, such as being a prior for topic coefficients as in LDA. Therefore we choose to leave θ as *fixed* parameters to be estimated via a maximum-likelihood principle or cross-validation techniques.

Now, putting things together, we arrive at a Bayesian EFH model with the following joint density function

$$p(\mathbf{x}, \mathbf{h}, \mathbf{W} | \boldsymbol{\theta}, \boldsymbol{\mu}, \boldsymbol{\sigma}) = p(\mathbf{W} | \boldsymbol{\mu}, \boldsymbol{\sigma}) p(\mathbf{x}, \mathbf{h} | \boldsymbol{\theta}, \mathbf{W}). \quad (5.8)$$

The hyperparameters in the model are $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$, which we treat as fixed quantities presumably known or to be estimated.

5.3 Model Inference and Estimation

5.3.1 Algorithm Overview

Given the prior distribution on \mathbf{W} with presumably known hyperparameters and a collection of N *iid*-sampled data $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_N)$, also assume that parameters $\boldsymbol{\theta}$ are known or already estimated, we need to compute or approximate the posterior

$$p(\mathbf{W} | \mathbf{X}) \propto p(\mathbf{X} | \mathbf{W}) p(\mathbf{W}) = \frac{1}{(Z(\mathbf{W}))^N} \tilde{p}(\mathbf{X} | \mathbf{W}) p(\mathbf{W}) \quad (5.9)$$

and the predictive posterior density over hidden variables

$$p(\mathbf{h} | \mathbf{x}, \mathbf{X}) = \int_{\mathbf{W}} p(\mathbf{h} | \mathbf{x}, \mathbf{W}) p(\mathbf{W} | \mathbf{X}) d\mathbf{W}, \quad (5.10)$$

where $\tilde{p}(\cdot)$ in Eq. 5.9 represents the unnormalized density function corresponding to $p(\cdot)$.

We can take a Monte Carlo approach to obtain a set of m samples $\{\mathbf{W}_1, \dots, \mathbf{W}_m\}$ by simulating an ergodic Markov chain whose stationary distribution is the posterior $p(\mathbf{W} | \mathbf{X})$. The difficulty here is due to the presence of an intractable term $(1/Z(\mathbf{W}))^N$ in the posterior distribution, which is a *function* of the target random parameters \mathbf{W} . Therefore, unlike simple posterior inference settings in which there is a normalization constant that will be canceled out by computing the ratio of two posterior densities or taking the derivative, in Bayesian inference with MRFs using MCMC we have to seek an efficient approximation of the intractable random partition function in posterior distribution.

In the following, we investigate two MCMC approximation schemes and show that in both cases the intractable term can be written as expectations under the data distribution $p(\mathbf{x}|\mathbf{W})$. Then we show that these terms can be approximated efficiently by minimizing the contrastive divergence (CD) [55], or equivalently, by performing Gibbs sampling for only very few steps starting from data (the empirical distribution). The derivation is in parallel with that in [81]; here we provide a more detailed discussion on the comparison of the two schemes.

5.3.2 Approximation schemes

Metropolis-Hasting algorithms

Consider simulating a Markov chain using a Metropolis-Hasting algorithm with the proposal distribution $q(\mathbf{W}'|\mathbf{W})$. The acceptance probability of the transition $\mathbf{W} \rightarrow \mathbf{W}'$ is

$$\rho(\mathbf{W}, \mathbf{W}') = \min \left(\frac{p(\mathbf{W}'|\mathbf{X})}{p(\mathbf{W}|\mathbf{X})} \frac{q(\mathbf{W}|\mathbf{W}')}{q(\mathbf{W}'|\mathbf{W})}, 1 \right) \quad (5.11)$$

Suppose the proposal distribution is easy to draw sample from and is tractable, then the only difficulty in implementing Metropolis-Hasting algorithms is to approximate the intractable term $\left(\frac{Z(\mathbf{W})}{Z(\mathbf{W}')}\right)^N$, where N is the size of the data set. The ratio of two partition functions can be written as an expectation over the data distribution $p(\mathbf{x}|\mathbf{W}')$.

$$\begin{aligned} \frac{Z(\mathbf{W})}{Z(\mathbf{W}')} &= \sum_{\mathbf{x}} e^{\frac{1}{2}\mathbf{x}^T(\mathbf{W}\mathbf{W}^T - \mathbf{W}'\mathbf{W}'^T)\mathbf{x}} \frac{e^{\boldsymbol{\theta}^T\mathbf{x} + \frac{1}{2}\mathbf{x}^T\mathbf{W}'\mathbf{W}'^T\mathbf{x}}}{Z(\mathbf{W}')} \\ &= \left\langle \exp \left\{ \frac{1}{2}\mathbf{x}^T (\mathbf{W}\mathbf{W}^T - \mathbf{W}'\mathbf{W}'^T) \mathbf{x} \right\} \right\rangle_{p(\mathbf{x}|\mathbf{W}')} \end{aligned} \quad (5.12)$$

The Langevin algorithm

We also investigate the Langevin algorithm as an alternative approximate MCMC scheme. The Markov chain simulated by the Langevin algorithm is characterized by the following stochastic transition equation

$$\mathbf{W}' = \mathbf{W} + \frac{\epsilon^2}{2} \nabla \log p(\mathbf{W}|\mathbf{X}) + \epsilon N_{\mathbf{W}} \quad (5.13)$$

where $N_{\mathbf{W}}$ are randomly generated from $\mathcal{N}(\mathbf{0}, I_{|\mathbf{W}|})$. This is a discrete version of the Langevin diffusion and ϵ^2 corresponds to the discretization size. A diffusion is a continuous time process which can be defined by a stochastic differential equation. The Langevin diffusion is characterized by

$$d\mathbf{W}(t) = \frac{1}{2} \nabla \log p(\mathbf{W}(t)|\mathbf{X}) dt + dB(t) \quad (5.14)$$

where $B(t)$ is a $|\mathbf{W}|$ -dimensional Brownian motion. The Markov chain converges when ϵ is reasonably small and has the desired density $p(\mathbf{W}|\mathbf{X})$ as $\epsilon^2 \rightarrow 0$. The gradient of the posterior is

the sum of three terms

$$\nabla \log p(\mathbf{W}|\mathbf{X}) = \nabla \log p(\mathbf{X}, \mathbf{W}) = \nabla \log p(\mathbf{W}) + \nabla \log \tilde{p}(\mathbf{X}|\mathbf{W}) + (-N \nabla \log Z(\mathbf{W})) \quad (5.15)$$

where in the GB-EFH model

$$\{\nabla \log p(\mathbf{W})\}_{ij} \triangleq \frac{\partial \log p(\mathbf{W})}{\partial W_{ij}} = \frac{\partial \log p_{ij}(\mathbf{W})}{\partial W_{ij}} = -\frac{1}{\sigma_i^2}(W_{ij} - \mu_i) \quad (5.16)$$

and $\nabla \log \tilde{p}(\mathbf{X}|\mathbf{W})$ is also tractable

$$\nabla \log \tilde{p}(\mathbf{X}|\mathbf{W}) = \sum_i \nabla \log \tilde{p}(\mathbf{x}_i|\mathbf{W}) = \sum_i \mathbf{x}_i \mathbf{x}_i^T \mathbf{W} = \mathbf{X} \mathbf{X}^T \mathbf{W}. \quad (5.17)$$

Hence, the only intractable term involved in the Langevin algorithm is $N \nabla \log Z(\mathbf{W})$, in which $\nabla \log Z(\mathbf{W})$ can be written as an expectation over the data distribution $p(\mathbf{x}|\mathbf{W})$

$$\begin{aligned} \nabla \log Z(\mathbf{W}) &= \frac{1}{Z(\mathbf{W})} \sum_{\mathbf{x}} \nabla \tilde{p}(\mathbf{x}|\mathbf{W}) = \frac{\tilde{p}(\mathbf{x}|\mathbf{W})}{Z(\mathbf{W})} \sum_{\mathbf{x}} \nabla \log \tilde{p}(\mathbf{x}|\mathbf{W}) \\ &= \sum_{\mathbf{x}} p(\mathbf{x}|\mathbf{W}) \nabla \log \tilde{p}(\mathbf{x}|\mathbf{W}) = \left\langle \mathbf{x} \mathbf{x}^T \mathbf{W} \right\rangle_{p(\mathbf{x}|\mathbf{W})} \end{aligned} \quad (5.18)$$

Discussion on the two schemes

The straightforward approach of estimating $Z(\mathbf{W})$ itself often fails to provide reliable estimates. To provide some intuition of the nature of this difficulty, we give a brief illustration as follows: with some mathematical manipulation, the partition function in the BG-EFH model equals the expectation of the following random variable

$$Z(\mathbf{t}) = \prod_i (1 + t_i) \quad (5.19)$$

under the multivariate lognormal distribution of \mathbf{t}

$$\mathbf{t} \sim \text{LogNormal}(\boldsymbol{\theta}, \mathbf{W} \mathbf{W}^T)$$

Thus under the Bayesian framework in which \mathbf{W} is considered a random matrix, we should expect $Z(\mathbf{W})$ to have *exponential* mean and variance.

Thus, we put more emphasis on variance in the bias-variance tradeoff of estimators in approximate Bayesian learning. Compare the approximations in the Langevin algorithm to update \mathbf{W} as in Eq. 5.13 and in Metropolis-Hasting algorithms to compute the acceptance probability as in Eq. 5.11

$$\epsilon^2 \nabla \log p(\mathbf{W}|\mathbf{X}) = -\epsilon^2 N \nabla \log Z(\mathbf{W}) + C \quad (5.20)$$

$$\left(\frac{Z(\mathbf{W})}{Z(\mathbf{W}')} \right)^N \approx e^N e^{(\mathbf{W} - \mathbf{W}') N \nabla \log Z(\mathbf{W}')} \quad (5.21)$$

where $C = \epsilon^2 (\nabla \log p(\mathbf{W}) + \nabla \log \tilde{p}(\mathbf{X}|\mathbf{W}))$ can be computed exactly, and Eq. 5.21 is obtained by first-order Taylor expansion. We expect the latter approximation has *exponential* variance compared to the former one. Therefore, we choose the Langevin algorithm conjoint with the MCMC scheme for posterior inference on BEFH model.

5.3.3 Approximating the expectations with brief sampling

$\nabla \log Z(\mathbf{W})$ in Eq. 5.18 can be estimated using a “sampling very few steps from the data” technique. It is first proposed in [55] under the name of minimizing contrastive divergence (CD) and suggested by [81] for approximate Bayesian inference in MRF in which it is named *brief sampling*. Brief sampling in GB-EFH runs multiple chains starting from the data X . Each chain performs l full step of Gibbs sampling. A total of N samples are obtained, denoted by $X_l = (\mathbf{x}_1^{(l)}, \dots, \mathbf{x}_N^{(l)})$. Then $\nabla \log Z(\mathbf{W})$ is approximated as an expectation over the empirical distribution of X_l . This whole procedure of brief sampling is illustrated as follows where we set $l = 1$ to perform just a single iteration:

- Draw $\mathbf{h}_k^{(1)} \sim \mathcal{N}(\mathbf{W}^T \mathbf{x}_k, I_J)$ for $k = 1, \dots, N$;
- Draw $\mathbf{x}_k^{(1)} \sim \text{Bernoulli}(\text{logit}((\boldsymbol{\theta} + \mathbf{W}\mathbf{h}_k^{(1)})))$ for $k = 1, \dots, N$;
- $\nabla \log Z(\mathbf{W}) \approx \frac{1}{N} \sum_k \mathbf{x}_k^{(1)} (\mathbf{x}_k^{(1)})^T \mathbf{W} = \frac{1}{N} \mathbf{X}_1 \mathbf{X}_1^T \mathbf{W}$

Brief sampling has been previously shown to provide low variance estimation with a small bias in ML learning [22]. The intractable term in ML learning of MRF is just the same term $\nabla \log Z(\mathbf{W})$, therefore we expect similar low-variance behavior of brief sampling estimation in the Langevin algorithm. Fig. 5.4 in the experiment section provides an empirical demonstration.

5.3.4 Computing the predictive posterior density

Given m samples $\{\mathbf{W}_1, \dots, \mathbf{W}_m\}$ obtained by the Langevin algorithm with brief sampling described above, the predictive conditional distribution is approximated by

$$p(\mathbf{h}|\mathbf{x}, \mathbf{X}) = \frac{1}{m} \sum_{k=1}^m p(\mathbf{h}|\mathbf{x}, \mathbf{W}_k). \quad (5.22)$$

More specifically, in GB-EFH we are interested in the conditional expectation of \mathbf{h} given \mathbf{x} , this is computed as

$$E(\mathbf{h}|\mathbf{x}, \mathbf{X}) = \frac{1}{m} \sum_{k=1}^m E(\mathbf{h}|\mathbf{x}, \mathbf{W}_k) = \frac{1}{m} \sum_{k=1}^m \mathbf{W}_k^T \mathbf{x}. \quad (5.23)$$

5.3.5 Hyperparameter Estimation

Now we briefly outline how to compute the maximum likelihood estimates of the hyperparameters μ and σ of BEFH from training data, based on an empirical Bayes principle. We employ a Monte Carlo EM scheme. In the “E”-step, we impute the hidden variables in BEFH, specifically, \mathbf{W} , from its posterior distribution; and in Section 5.3.2 we have developed the Langevin algorithm for this step. Given a set of K imputed \mathbf{W} from iteration t , we proceed to the “M”-step, in which now we are essentially back to the standard ML learning scenario for fully observed MRF,

and compute an estimate of the hyperparameters as follows:

$$\begin{aligned}
(\mu^{(t+1)}, \sigma^{(t+1)}) &= \arg \max_{\mu, \sigma} \sum_{k=1}^K \log p(X, \mathbf{W}_k^{(t)} | \mu, \sigma) \\
&= \arg \max_{\mu, \sigma} \sum_{k=1}^K (\log p(\mathbf{W}_k^{(t)} | \mu, \sigma) + \log p(X | \mathbf{W}_k^{(t)})) \\
&= \arg \max_{\mu, \sigma} \sum_{k=1}^K \log p(\mathbf{W}_k^{(t)} | \mu, \sigma),
\end{aligned} \tag{5.24}$$

where $\mathbf{W}_k^{(t)}$ denotes the k -th imputed sample at iteration t .

It can be shown that, the ML estimate of each element of μ and σ is:

$$\mu_i^{(t+1)} = \frac{1}{JK} \sum_j \sum_k W_{ij,k}^{(t)} \tag{5.25}$$

$$\sigma_i^{(t+1)} = \sqrt{\frac{1}{JK-1} \sum_j \sum_k (W_{ij,k}^{(t)} - \mu_i^{(t+1)})^2} \tag{5.26}$$

where $W_{ij,k}^{(t)}$ denotes the ij -th element of $\mathbf{W}_k^{(t)}$.

To initialize the EM procedure, we can make use of the ML estimate of \mathbf{W} , denoted by \mathbf{W}^{MLE} , and let

$$\mu_i^{(0)} = \frac{1}{J} \sum_j W_{ij}^{MLE} \tag{5.27}$$

$$\sigma_i^{(0)} = \sqrt{\frac{1}{J-1} \sum_j (W_{ij}^{MLE} - \mu_i^{(0)})^2} \tag{5.28}$$

This concludes the algorithmic section.

5.4 Experimental Evaluation

5.4.1 Synthetic EFH parameter estimation

The dataset is generated for a GB-EFH model with $\theta = 0$. The model contains $I = 100$ observed variables and $J = 10$ hidden variables, so the number of parameters in W is $I \times J = 1000$. We vary the size of the training dataset from 25 to 200 and compare the performance of ML estimation via gradient ascent and the Langevin algorithm proposed in the previous section.

Generating *iid* samples from a general MRF is known to be non-trivial. However, for a GB-EFH model exact samples can be generated fairly efficiently by employing the perfect sampling technique [28] when all the elements of the matrix $V = WW^T$ are non-negative. To ensure this property, we first generate an $M \times M$ matrix whose elements are uniformly distributed in the

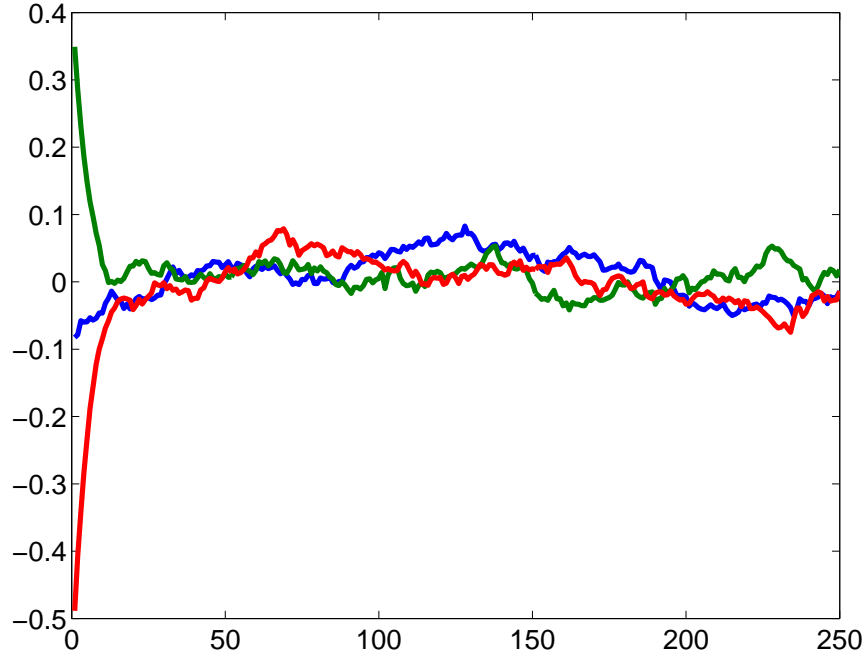


Figure 5.3: Details of Monte Carlo simulations of the Langevin algorithm, with y -axis corresponds to the value of W_{11} . Three chains of different starting points are shown. The burn-in time to reach convergence is approximately 50 transition.

$[0, 0.1]$ interval. Then W is determined by performing an SVD on this matrix so that V is the best rank- J approximation.

There is an un-identifiability issue here because the data distribution

$$p(\mathbf{x}|\mathbf{W}) = \frac{1}{Z} \exp\left(\frac{1}{2}\mathbf{x}^T \mathbf{W} \mathbf{W}^T \mathbf{x}\right) \quad (5.29)$$

is a function of V and is invariant if W is right-multiplied by an orthogonal matrix Q because $(WQ)(WQ)^T = WW^T$. Also it can be shown the prior of W defined in Eq. 5.7 is also invariant under this transformation. Therefore our evaluation criteria are based on the matrix V instead of W . We define two error measures: mean averaged error (mae) and mean relative error (mre) to evaluate an estimate \hat{V}

$$mae = \frac{1}{M^2} \sum_i \sum_j |V_{ij} - \hat{V}_{ij}| \quad (5.30)$$

$$mre = \frac{1}{M^2} \sum_i \sum_j \frac{|V_{ij} - \hat{V}_{ij}|}{\max\{|V_{ij}|, |\hat{V}_{ij}|\}} \quad (5.31)$$

Fig. 5.4 shows the estimate of the gradient using brief sampling versus the number of sampling steps l . We also generate the same number of samples using the perfect sampling technique to provide an approximately correct version for comparison. Brief sampling provides biased estimation compared to the exact sampling approach, but the bias is relatively small considering

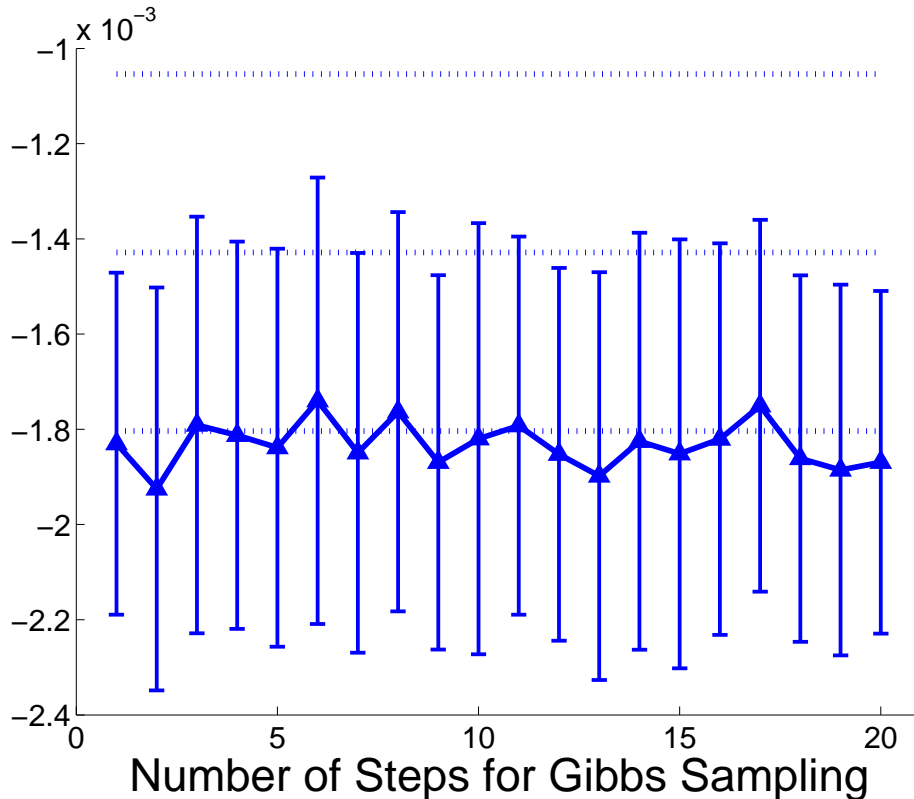


Figure 5.4: The estimation versus the number of sampling steps in brief sampling (solid line) compared with the estimation perfect sampling (dash line), with y -axis corresponds to an estimated derivative of log-partition function $\partial \log Z(\mathbf{W}) / \partial \mathbf{W}_{11}$ averaged over 50 runs. Both sampling schemes generate 100 samples in each run. The standard error bars are scaled by 1.64, indicating 95% significance of the difference in estimation. A single sampling step suffices as it maximizes the program efficiency without increasing bias or variance of the estimation.

the difficulty of dealing with intractable partition function. Note that the bias is not decreased by increasing l . The variance of the estimation, on the other hand, is minimized when $l = 1$. Therefore, we let $l = 1$ in subsequent experiments.

Two tunable parameters in the Langevin algorithm are yet to be determined: the step size ϵ in Eq. 5.13 and the number of steps l to sample from data in brief sampling. We choose an appropriate ϵ by investigating the evolution of a number of elements of W during the simulation of the Markov chain. Under a too large step size the chain goes to infinity in a few steps, and under a too small one the burn-in time is undesirably long. Fig 5.3 shows a simulation of the Langevin algorithm using the step size we choose.

In Fig. 5.5 we compare the performance of ML estimation via gradient ascent and the Bayesian approach using the Langevin algorithm. The Langevin algorithm consistently achieves lower errors under both measures and with different sizes of the training set. As more data are available, the performance of ML estimation improves little; it appears that the gradient ascent procedure

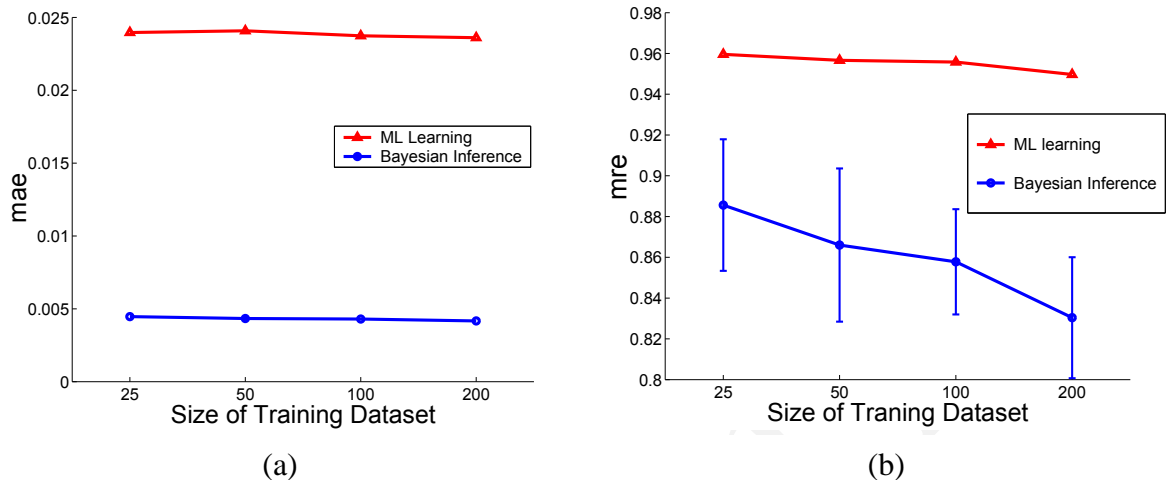


Figure 5.5: The Performance of ML learning and Bayesian inference using the brief Langevin algorithm under two different error measures (a) mean absolute error; (b) mean relative error. The results are averaged over 10 runs; the error bar may be too small to be distinguishable from the figure. The Bayesian approach is subject to less error rate than its ML alternative in both measures.

gets stuck into a local minimum. On the other hand, the Langevin algorithm does benefit from more data, which is possibly the consequence of the uninformative prior we placed for this problem by setting $\mu_i = 0, \sigma_i = d = 0.1$ for $i = 1, \dots, M$. The estimation by both methods has a non-negligible bias from the true value, and we conjecture that it is due to the sparsity of the data. We also observe that the performance difference of ML estimation and the Langevin algorithm is much larger as measured by *mean absolute error* than *mean relative error*, which suggests that the latter algorithm provides better estimates for parameters with larger values.

5.4.2 Classification of Text and Image Data

The data set is from the compiled TRECVID'03 news video collection in [114]. It contains 1078 video shots with captions, each of which can be treated as a document and belongs to one of five pre-defined categories. 1894 binary word occurrence features and 166 continuous features for key images are extracted from each document. We extend the dual-wing harmonium (DWH) developed in [114], which was previously trained by ML estimation, to Bayesian DWH (BDWH) in which column-*iid* multivariate normal priors are placed on the coupling matrices for word and image features respectively. The hyperparameters in the priors are estimated using the empirical Bayes method developed in Section 5.3.5.

To give a hint on the difficulty of performing Bayesian learning in a real dataset discussed in Section 5.3.2, we implement the naïve Monte Carlo estimation of the partition function in Eq. 5.19 for both GB-EFH with synthetic dataset and DWH with real world dataset. The histograms of the estimated Z over 100 runs are shown in Fig. 5.6. In the synthetic dataset the estimated values approximately fit to a normal distribution. However, in the real dataset, there

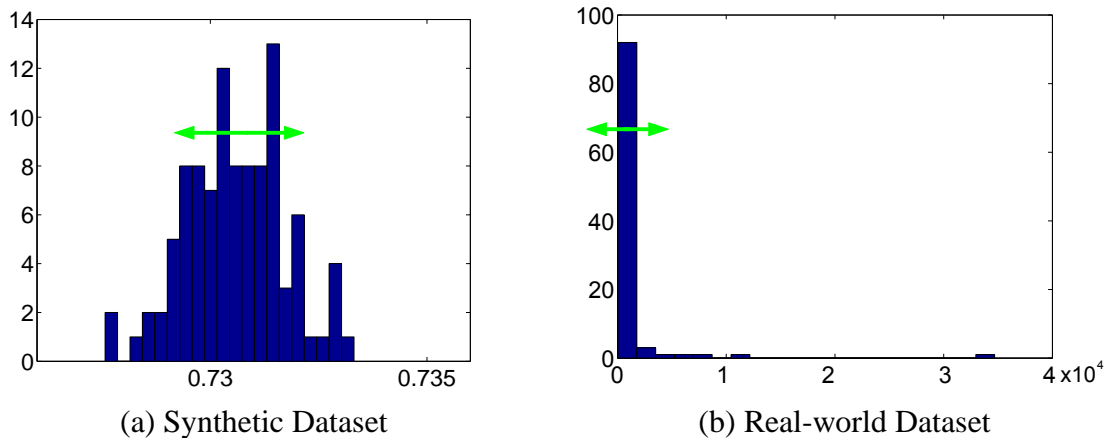


Figure 5.6: Histogram of 100 estimations of partition function using a naïve Monte Carlo approximation on (a) synthetic dataset; (b) real-world dataset. Arrows are centered at the mean and indicate an interval of length of 2 times the standard deviation. Each estimation computes the expectation using 1000 samples. On the real-world data set, the estimation is subject to unacceptably high variance.

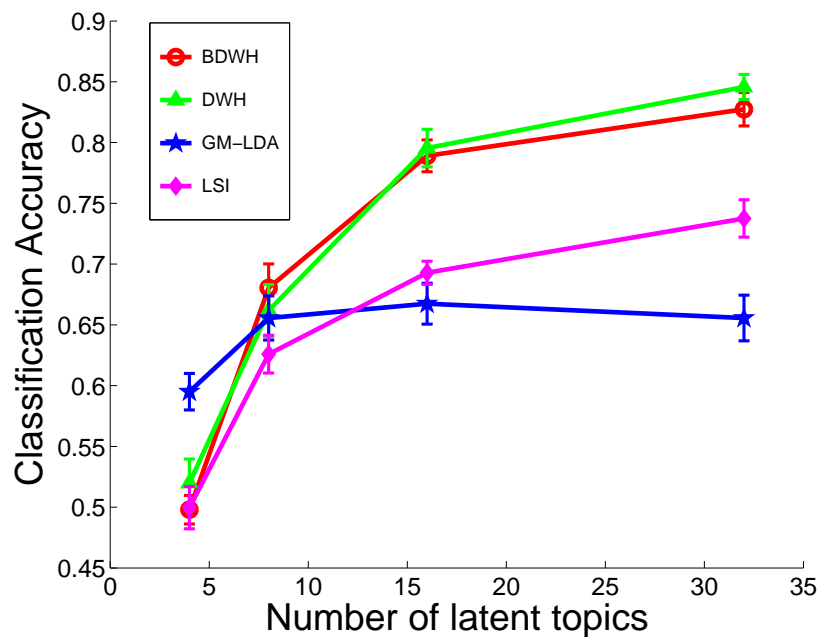


Figure 5.7: Classification accuracy versus number of latent topics. Bayesian DWH yields comparable performance to the original DWH approach. Both produce better results over the baseline LSI approach and the GM-LDA approach backed by a directed graphical model.

are a few spurious outliers, which shift the mean estimated values over all the runs significantly, leading to generally biased, high variance estimates. In Fig. 5.6(b) the variance of the estimation is three times as large as the estimated mean.

We evaluate the performance of four different models LSI [34], GM-LDA, DWH and BDWH for classification task on the news video collection. For each algorithm, the parameters are estimated using all data, without reference to their labels. Once the model are learned, every document in the data are projected into the lower-dimensional latent semantic space. The data are then randomly split to a training set and a testing set with the same size. We show the result of using one nearest neighbor (1-NN) classifier to predict the category of each test data given the training data.

Fig. 5.7 compares the performance obtained at different dimensions of latent semantic space, or equivalently different numbers of latent topics ranging from 4 to 32. BDWH and DWH achieve comparable classification accuracy consistently, and outperform LSI and GM-LDA with a good margin when the number of latent topics are 16 and 32. LSI, DWH and BDWH all get better performances in higher dimensional semantic space with less dimensionality-reduction. In contrast, GM-LDA outperforms other methods when the number of latent topics are 4 but the performance curve goes down when the number of latent topics increases from 16 to 32, which may reflect a low-dimensionality bias from the modeling.

5.5 Conclusion

We have proposed a new Bayesian formalism of EFH model and variants for latent semantic modeling of text and multimedia data. The Langevin algorithm conjoint with an MCMC scheme was applied to carry out approximate posterior inference, and an empirical Bayes method is also developed for estimating the parameters. The Bayesian approach achieves superior performance of parameter estimation on a synthetic data set and comparable classification accuracy on a real dataset of both text and image data.

EFH models differ from singular value decomposition in that there is a freedom to choose different exponential family distribution to model the input data, which could be either discrete or continuous. Compared with a linear ICA model, hidden topics in EFH are *not* assumed to be statistically independent. Instead, they are conditionally independent given the observed layer. This reflects the assumption that topics could be semantically different but correlated, such as “science” and “technology”. Similar assumptions could also be spotted in other studies [5, 17].

Our experiments presented in this chapter focus on binary occurrences of words which is suitable for short texts. A good future direction is to build an BEFH to directly model word counts. Also, the independent Gaussian prior we used can be replaced by an more informative one, while the inference and learning algorithm can straightforwardly apply to the new formalism. Finally, the discretization scheme in the Langevin algorithm can be more elaborate, such as incorporating the idea suggested in [96].

Chapter 6

Click Models: Leveraging User Feedback for Better Search Experiences

In the era of World Wide Web, search engines have become essential tools for browsing and navigating over vast amounts of information on the internet. After a query submission, the user interacts with the search engine via examining through the snippets from web documents in the ranked list of query results and following the hyper-link with a click if she would like to find more about a particular one. Such events are usually logged to track user activities and provide insights about the performance of the search engine. It is probably the most extensive, albeit indirect, surveys on user experience, especially in the event that explicit user feedback is either expensive or less likely to be collected, which is usually true for any query system with a moderate or large user base.

In this chapter, we study how to leverage user click data to obtain a similarity score, known as user-perceived relevance, for any query term and result document pair. Such scores form an important feature to adapt future ranking for improved user experience. Although much of the material is presented in the context of web search, it should be applicable to other search problems as well, including the task of querying multimedia databases. Most of the work described subsequently is based on the material presented in [49, 50, 51, 72].

6.1 Background

We first introduce definitions and notations that will be used throughout this chapter. A web search user initializes a *query session* by submitting a *query* to the search engine. We regard re-submissions and reformulations of the same query as distinct query sessions. Snippets from *web documents* are presented in a ranked order in the first result page, where a document in a higher *position*, or *rank*, appears above those in lower positions. Such appearance is also known as *impression* in the web search community. The identity of a document impressed at the i th position is denoted by d_i , where the value of i ranges between 1 and M , the latter of which is the maximum number of results displayed in the page.

6.1.1 Click Position-Bias

If a document is impressed and clicked, it indicates a positive feedback from the user regarding the relevance of the document. On the other hand, if there is only impression without any click, it possibly links to a negative feedback. This may seem a reasonable conclusion at first sight, however, empirical studies such as [62] have proved that this straightforward idea for leveraging click data is subject to heavy bias favoring documents that appear at higher positions to those lower ones, regardless of their snippets or contents. This could be (partially) attributed to the fact that users are accustomed to examine over search results in a roughly linear order from the top to the bottom, with the possibility of an early termination, reflecting varying degrees of trust on the ranking algorithm of the search engine tailored to their preferences. Since a typical user may not go over every document in the page, for a document that appears at the bottom, even if it mostly satisfied the user's information needs, it may still receive much less user attention and clicks than the top ranked one.

This *position-bias* in observing clicks over different positions can be portrayed by examining Figure 6.1 obtained from an eye-tracking study in [62]. Both plots display how the eye fixation, a measure of user attention, and the number of clicks vary over the top-10 search results. The difference between the two is in the ranking of search results - the top plot assumes the default ranking, whereas the bottom plot corresponds to a fully reverse one, switching the 1st position with the 10th, the 2nd with the 9th, and so on. If there were no position-bias at all, we would expect the count of clicks would be mirrored in the bottom plot as a result of such switching. However, the top position in the bottom figure still receives most user attention, and much more clicks than bottom ones, which implies that position should not be excluded when assessing chances of clicks over search results.

Position-bias has become a key challenge in the accurate interpretation of user clicks and inspires the proposal of a number of hypotheses to provide formal description, followed by the development of click models to offer more principled solutions.

6.1.2 Basic Hypotheses

The *examination hypothesis*, proposed first in [94], characterizes user interaction with two types of probabilistic events: examination and click over a document. The insight is to impose examination as a prerequisite for click over the same document, and link the relevance of the document to the conditional probability of a click given that it has been already examined. Formally, for a given query session, we use binary random variables E_i and C_i to represent the examination and click events of the document at position i , respectively, and denote corresponding examination and click probabilities by $P(E_i = 1)$ and $P(C_i = 1)$. The *examination hypothesis* can be summarized as follows:

$$\begin{aligned} P(C_i = 1 | E_i = 0) &= 0, \\ P(C_i = 1 | E_i = 1) &= r_{d_i}, \end{aligned}$$

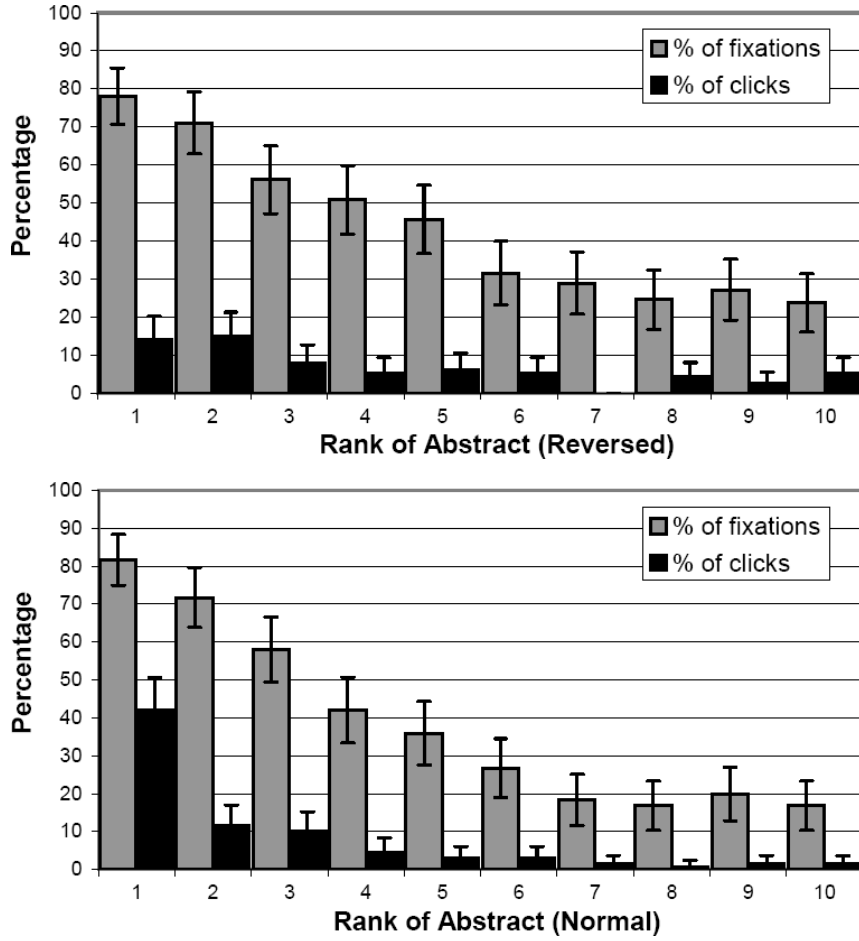


Figure 6.1: Comparison of user attention (fixation) and clicks over top 10 ranks between the normal order and the reversed order reveals position bias. Plots are extracted from Figure 4 in [62].

where $1 \leq i \leq M$, and r_{d_i} , defined as the *document relevance*, is the conditional probability of click after examination. Given E_i , C_i is conditionally independent of previous examine/click events $E_{1:i-1}, C_{1:i-1}$. This helps to disentangle click activities of various documents as being caused by position and content. Click models based on the examination hypothesis share this definition but differ in the specification of $P(E_i)$.

The second hypothesis, known as the *cascade hypothesis* [33], portrays how user examines search results one by one. It states that users always start the examination at the first document. The examination is strictly linear to the position, and a document is examined only if all documents in higher positions are examined. Formally,

$$\begin{aligned}
 P(E_1 = 1) &= 1, \\
 P(E_{i+1} = 1 | E_i = 0) &= 0.
 \end{aligned}$$

The corollary is that given E_i , E_{i+1} is conditionally independent of all examine/click events

above i , but may depend on the click C_i .

In the same manuscript, the *cascade model* is proposed by putting together previous two hypotheses and further constrain that

$$P(E_{i+1} = 1 | E_i = 1, C_i) = 1 - C_i, \quad (6.1)$$

which implies that a user keeps examining the next document until reaching the first click, after which the user simply stops the examination and abandons the query session. However, it is unclear from the model how to explain multiple clicks existing in the same query session. A quick solution is to (independently) apply the same model multiple times, which does not have a sound probabilistic interpretation.

6.2 Proposed Method

This section is devoted to the design and implementation of the dependent click model (DCM), originally presented in [51]. More sophisticated Bayesian click models were presented in [49] and [72]; they share similar design goals and application settings as DCM, and yield better performance when click data are less available, at the expense of additional computational time and storage. All these models take a single pass over the click data, scale linearly in time, and need only constant space for each query-document pair.

In the aforementioned cascade model a user always leaves the result page upon the first click and never comes back. We propose to include a position-dependent parameter λ_i to reflect the chance that the user would like to see more results after a click at position i . λ_i 's is a set of user behavior parameters shared over multiple query sessions. DCM inherits the assumption that in the case of examination without a click, the next document is always examined. This user model is shown in Figure 6.2.

Examination and click probabilities in DCM can be specified in an iterative fashion ($1 \leq i \leq M$):

$$\begin{aligned} e_{d_1,1} &= 1, \\ c_{d_i,i} &= e_{d_i,i} r_{d_i}, \\ e_{d_{i+1},i+1} &= \lambda_i c_{d_i,i} + (e_{d_i,i} - c_{d_i,i}), \end{aligned} \quad (6.2)$$

from which the following closed-form equations can be derived:

$$e_{d_i,i} = \prod_{j=1}^{i-1} (1 - r_{d_j} + \lambda_j r_{d_j}), \quad (6.3)$$

$$c_{d_i,i} = r_{d_i} \prod_{j=1}^{i-1} (1 - r_{d_j} + \lambda_j r_{d_j}). \quad (6.4)$$

This completes the formal specification of the *dependent click model* (DCM), in which examine probabilities and click probabilities at different positions i become interdependent.

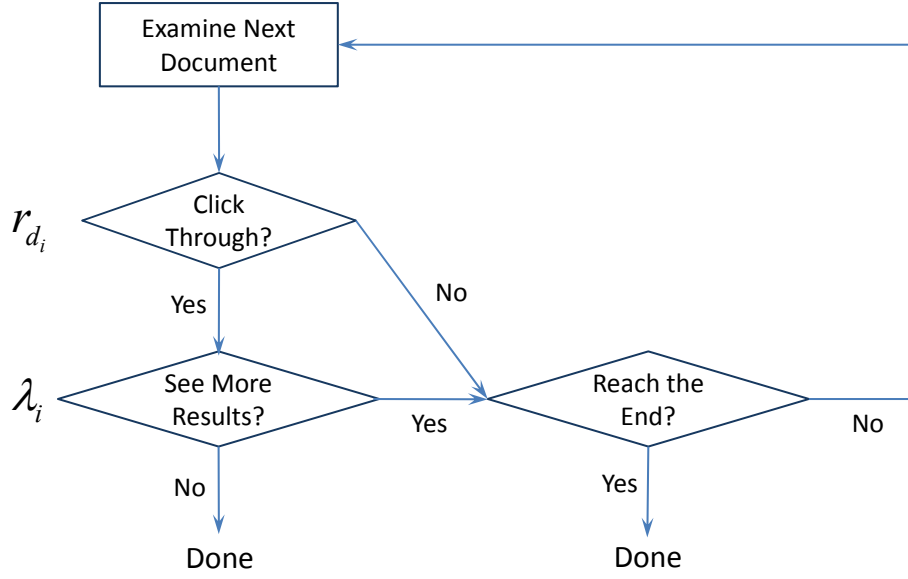


Figure 6.2: The user model of DCM, in which r_{d_i} is the document relevance of d_i , and λ_i is the user behavior parameter for position i .

Given the actual click event $\{C_1, \dots, C_M\}$ in a query session as well as the document impression $\{d_1, \dots, d_M\}$, the log-likelihood for a query session with one or more clicks is given by

$$\begin{aligned} \ell = & \sum_{i=1}^{l-1} \left(C_i (\log r_{d_i} + \log \lambda_i) + (1 - C_i) \log(1 - r_{d_i}) \right) \\ & + C_l \log r_{d_l} + (1 - C_l) \log(1 - r_{d_l}) \\ & + \log \left(1 - \lambda_l + \lambda_l \prod_{j=l+1}^n (1 - r_{d_j}) \right), \end{aligned} \quad (6.5)$$

$$\begin{aligned} \geq & \sum_{i=1}^l \left(C_i \log r_{d_i} + (1 - C_i) \log(1 - r_{d_i}) \right) \\ & + \sum_{i=1}^{l-1} C_i \log \lambda_i + \log(1 - \lambda_l). \end{aligned} \quad (6.6)$$

If there is no click in this session, then the log-likelihood is a special case with $l = M$, $C_l = \lambda_l = 0$.

We carry out DCM learning by optimizing values of r_{d_i} and λ_i to maximize the sum of the lower bound of log-likelihood in Eq. 6.6 over the entire training set. Document relevance estimate for a document d is given by:

$$r_d = \frac{\# \text{ Click on } d}{\# \text{ Impression of } d \text{ before last clicked position } l}. \quad (6.7)$$

which is the empirical conditional click probability of d given it appears higher than or at position l . And the maximum-likelihood estimate for the user behavior parameter [51]

$$\lambda_i = 1 - \frac{\# \text{ query sessions when last clicked position} = i}{\# \text{ query sessions when position } i \text{ is clicked}}, \quad (6.8)$$

for $1 \leq i \leq M - 1$, which is the empirical probability of position i being a not-last-clicked position over all query sessions in the training set.

Therefore, we can set up two counting statistics to each document d , and parse only once through the training data to get all such counts, and finally compute all document relevance estimates. Similarly, we need additional $2(M - 1)$ global counts for λ_i 's. This leads to an learning algorithm with linear time complexity with respect to the number of query sessions and linear space complexity with respect to the number of distinct query-document pairs. When new data are available, we can do fast update and re-computation based on these counts, while maintaining the linear scalability.

An important difference of DCM from simply computing the clickthrough rate, the number of clicks divided by the number of impression, is that clicks indicate both relevance and examination. So if a document is not clicked, it can be attributed to either the document abstract is examined but not relevant enough to be clicked, or it appears lower than other documents that draw the user attention away. This explain-away effect is reflected in Eq. 6.7 by a smaller denominator which only counts impression before last clicks.

Finally, we give the sampling procedure for DCM which draws examination variables \mathcal{E}_i and click variables \mathcal{C}_i one-by-one starting from the top position, as applied in the next section to carry out empirical evaluation:

$$\begin{aligned} &\mathcal{E}_1 = 1; \\ &\text{If } \mathcal{E}_i = 0, \\ &\quad \mathcal{C}_i = 0, \mathcal{E}_{i+1} = 0; \\ &\text{else} \\ &\quad \mathcal{C}_i \sim \text{Bernoulli}(r_{d_i}), \\ &\quad \mathcal{E}_{i+1} \sim \text{Bernoulli}(1 - \mathcal{C}_i + \lambda_i \mathcal{C}_i). \end{aligned} \quad (6.9)$$

6.3 Experimental Evaluation

We report our experimental studies in this section, which is based on over 8.8 million queries sessions after data preprocessing, sampled from the click log of a major commercial search engine in July 2008. We are comparing the proposed DCM with the independent click model (ICM), the baseline approach unaware of the position-bias and the chance of no examination¹, under which

¹The fulltext of [37] proposing user browsing model, which appears in the SIGIR conference in 2008, became publicly available after we have developed the DCM. ICM reflected the best performed alternative to the extent of our knowledge while we conducted the experiment.

Table 6.1: Summary of Test Set

Query Freq	# queries	# Sessions	Avg # Click
1~9	59,442	216,653 (5.4%)	1.310
10~31	30,980	543,537 (13.5%)	1.239
32~99	13,667	731,972 (17.7%)	1.169
100~316	4,465	759,661 (18.9%)	1.125
317~999	1,523	811,331 (20.1%)	1.093
1000~3162	553	965,055 (24.0%)	1.072

the probability of a click is solely determined by the identity of the document, and equals the past clickthrough rate of the same document. In the following, we start with experimental setup in Section 6.3.1, and proceed with detailed results under two evaluation metrics in Section 6.3.2 and Section 6.3.3, respectively.

6.3.1 Experimental Setup

The data set is obtained by sampling the click log of a major commercial search engine during July 2008. The click log consists of the query string, the time-stamp, document impression data (identities of top-10 documents in the first page) and click data (whether each document is clicked or not) for each query session. Only query sessions with at least one click are kept for better data quality since we find from additional meta-information that clicks on ads, query suggestions or other elements are much more likely to appear for the ignored sessions with no clicks. It also provides clearer comparison of performances on predicting the first and last clicked position. For each query, we sort its query sessions by time-stamp and split them into training set and test set of equal sizes. The number of query sessions in the training set is 4,804,633. Then these queries are categorized according to the query frequency in the test set. Top 0.16% (178) most frequently searched queries (also known as *head queries*) with frequencies greater than $10^{3.5}$ are not included in the subsequent results on test set because most search engines already do very well on these queries. After data preprocessing, the test set consists of 4,028,209 query sessions for 110,630 distinct queries in 6 query frequency categories. The average number of clicks per query session is 1.139. Statistics of the test set are summarized in Table 6.1. Note that our data set includes a great number of tail queries which are often ignored in experiments conducted in previous studies, and performances over all query sessions are not dominated by head queries or a particular query frequency range.

For each query, document relevance estimates for DCM are computed using Eq. 6.7 on the training data, and for ICM it equals the clickthrough rate. But for documents which appear very few times in the training set and which appear only in the test set, document relevance are replaced by position relevance, which are computed for each position in a similar way, for deriving log-likelihood and other metrics in the test set. This has a smoothing effect on the document relevance, and leads to better performance for the evaluation on the test data. Since the additional counts that we need to keep in the computation, $2M$ for each query, is usually much

smaller than the cost saving from low-frequency documents, the time and space complexities can also be reduced. The cut off of minimum number of impression for document relevance computation is set adaptively according to the query frequency category (as shown in Table 6.1) from 1 to 6. Finally, to avoid infinite values in the evaluation, we further imposes a lower bound of 0.01 and an upper bound of 0.99 on the learned relevance values for both models as well as user behavior parameters in DCM.

Parsing the data from the hard disk and loading them into main memory takes around 45 minutes. All the subsequent experiments are carried out in a server machine, with 2.67GHz CPU cores, 32GB memory, Windows Server 2008 64-bit OS, and MATLAB R2008a installed. The computational time for training DCM is no more than 7 minutes.

6.3.2 Evaluation based on Log-Likelihood

Log-likelihood (LL) is widely used in the machine learning community to measure model fitness. Here it indicates a soft-version of the probability that clicks from the model prediction over top 10 positions are consistent with the ground truth over the test set. More formally, given the document impression for each query session in the test data, LL is defined as the log probability of observed click events computed under the trained model. A larger LL indicates better performance, and the optimal value is 0. The improvement of LL value ℓ_1 over ℓ_2 is computed as $(\exp(\ell_1 - \ell_2) - 1) \times 100\%$. We report average LL over multiple query sessions using arithmetic mean.

Figure 6.3 presents log-likelihood curves for different query frequencies, where larger log-likelihood results indicate better fit on the test data. DCM achieves larger performance gain for more frequent queries, and consistently outperforms ICM by over 10% when the query frequency is over 100. The difference is only less significant for tail queries of frequencies less than 10.

The DCM curve goes below ICM for queries with frequencies less than $10^{1.5}$. But this does not imply that we should always apply ICM to model these queries. Instead, we suggest that lower confidence should be given in document relevance estimates derived from click models for these tail queries. We could still record counting statistics for these queries, but document relevance estimates should be reliable when new data flow in and the amount of training data is enough to obtain a good fit.

6.3.3 Evaluation based on Position-Bias Plots

Click position-bias could be easily visualized by drawing a curve for probabilities of clicks over the top-10 positions based on the test data. And we compare the derived click probabilities from both DCM and ICM with the ground truth in Figure 6.4. DCM matches these probabilities very well at the top 5 positions. The higher tail of empirical curves is probably due to user scrolling behaviors, especially for informational queries which have a higher click through rate. And we suspect that users may examine documents in a different fashion when they scroll to the bottom of the search result page, so that the 10th position receives even more last clicks than the two above. However, they contribute to a fairly small fraction of overall results: clicks after position

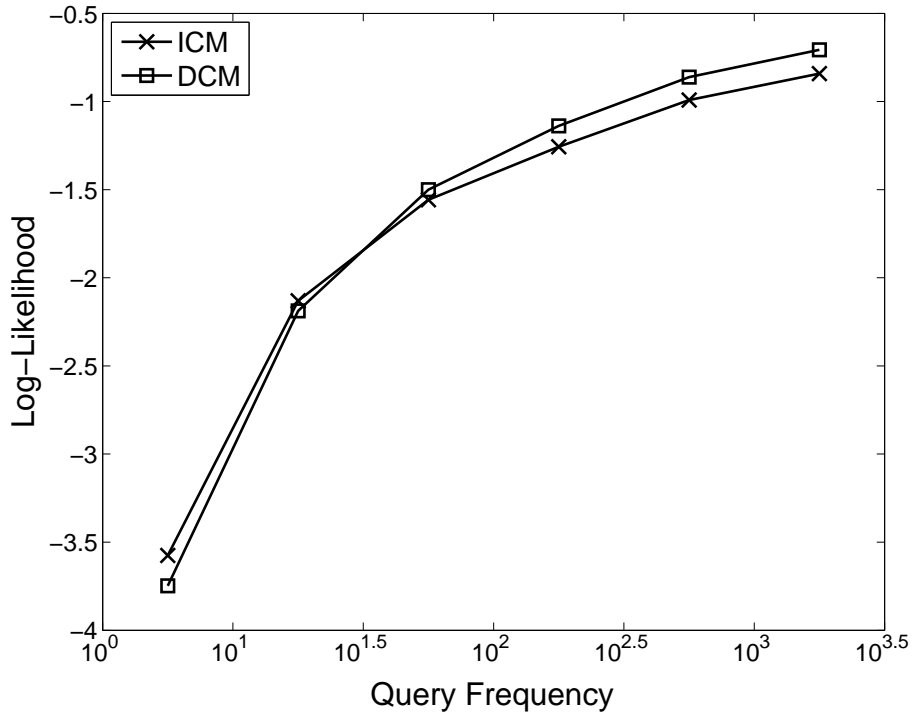


Figure 6.3: Log-likelihood per query session on the test data for different query frequencies. The overall average for DCM is -1.327, compared with -1.401 for ICM which reflects a 7% improvement.

6 represent only 6.1% of the total number. ICM tends to over-estimate clicks in lower positions, given the bias that it assumes every position is always examined.

A unique property of DCM is that examination probabilities could be computed for each query session and they are aggregated together to provide a hint on user attention over different positions, which corresponds to the dashed curve in Figure 6.4. The first position is always examined from the modeling assumption, followed by a geometrically decreasing pattern after position 2. Compared with the DCM click curve, the gap between them reflects the conditional click probabilities for each position, which suggests larger probabilities for both top and bottom positions. Note that both curves go below the empirical click for the last position, and this bias is attributed to user behaviors beyond the modeling assumption as discussed previously.

Figure 6.5 displays detailed DCM examination probabilities for different query frequencies. All of them share similar decreasing pattern but differ in absolute values. The trend is that less frequent queries tend to be examined in greater depth, and we also observe more clicks per query session in the click log for them.

6.3.4 Predicting First and Last Clicks

We now focus on clicks and test whether samples generated from ICM and DCM provide good match of first and last clicked position compared with the empirical data. Given each query

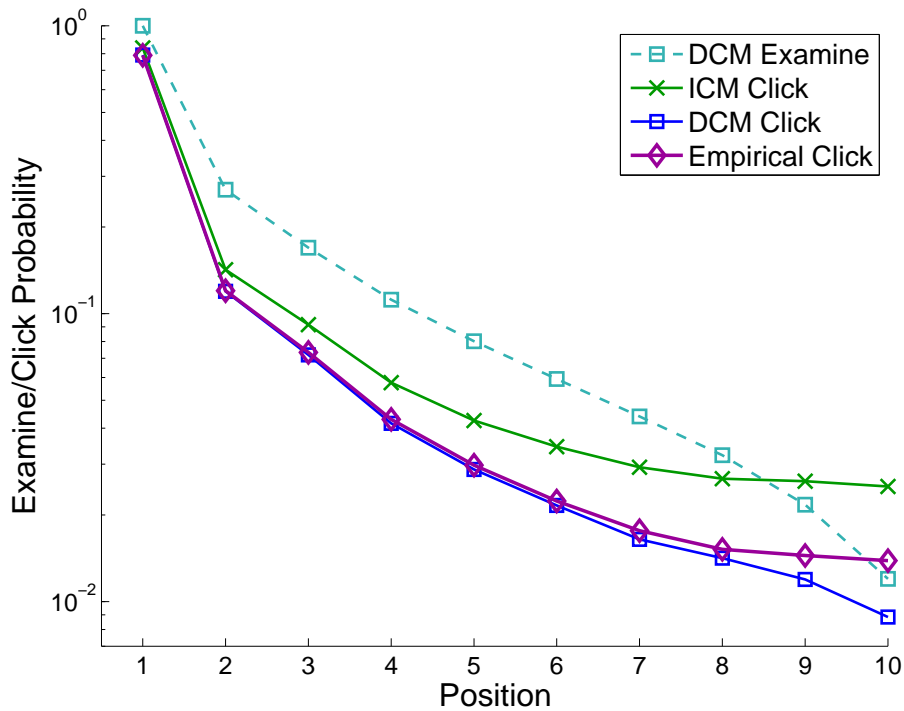


Figure 6.4: Click probabilities for different positions summarized from ICM/DCM samples as well as test data, and examine probabilities implied by DCM. The click distribution implied by DCM matches the ground truth closely.

session in the test set, we use the document relevance learned from the training set to determine the click probability. For ICM, clicks are sampled for each position independently, whereas for DCM, sampling starts from the top ranked document and ends at either the first non-examined position or the last (10th) position. For both models, we collect 100 samples with at least one click, then first and last clicked position are identified from the simulated click data and compared with the ground truth to compute RMS errors. This is the most time-consuming part in the model evaluation experiments and takes around one hour to finish. To reflect the inherent randomness in user click behavior, we also compute for each query the standard deviation of first and last clicked position and take a weighted mean over different queries to approximate the lower bound of RMS error. This corresponds to the “optimal” curves in Figure 6.6. We expect a model that gives consistently best fit of click data would have the smallest margin with respect to the optimal error, and this margin also reflects the robustness of model prediction since the RMS error metric takes account of both bias and variance in prediction. Finally, we aggregate results over all queries and compare the distribution of first and last clicks from two click models with the empirical distribution of the test data, which corresponds to the “empirical” curves in Figure 6.7.

RMS errors for ICM and DCM are close for first clicked position because their model assumptions are the same until the first click. Predicting last clicked position turns out to be a more difficult task as demonstrated by higher error curves in Figure 6.6(b) than 6.6(a). With a position-dependent modeling assumption, DCM outputs more reasonable last click estimates

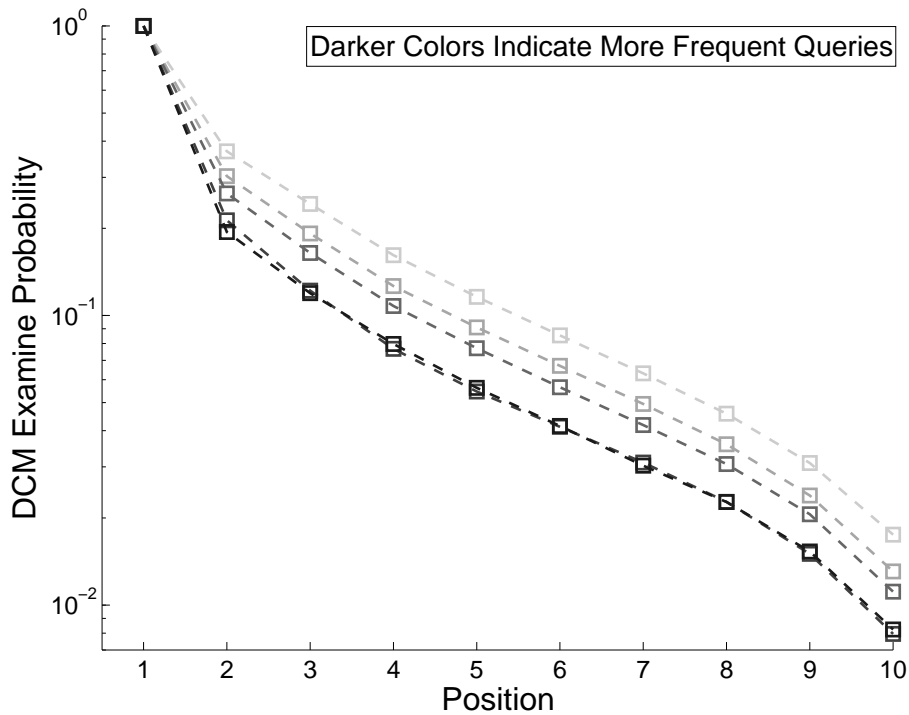
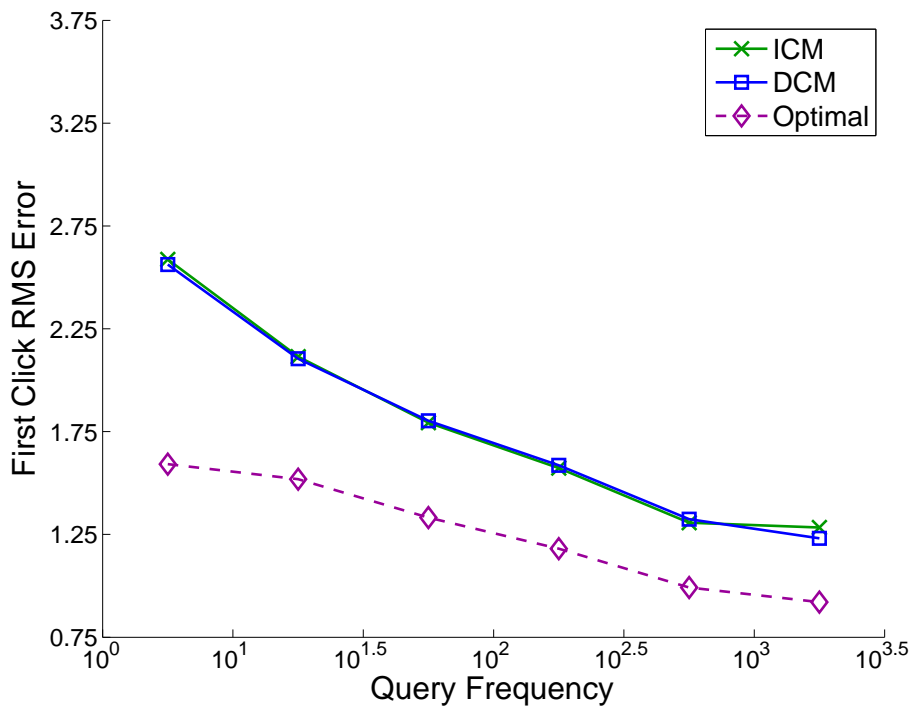


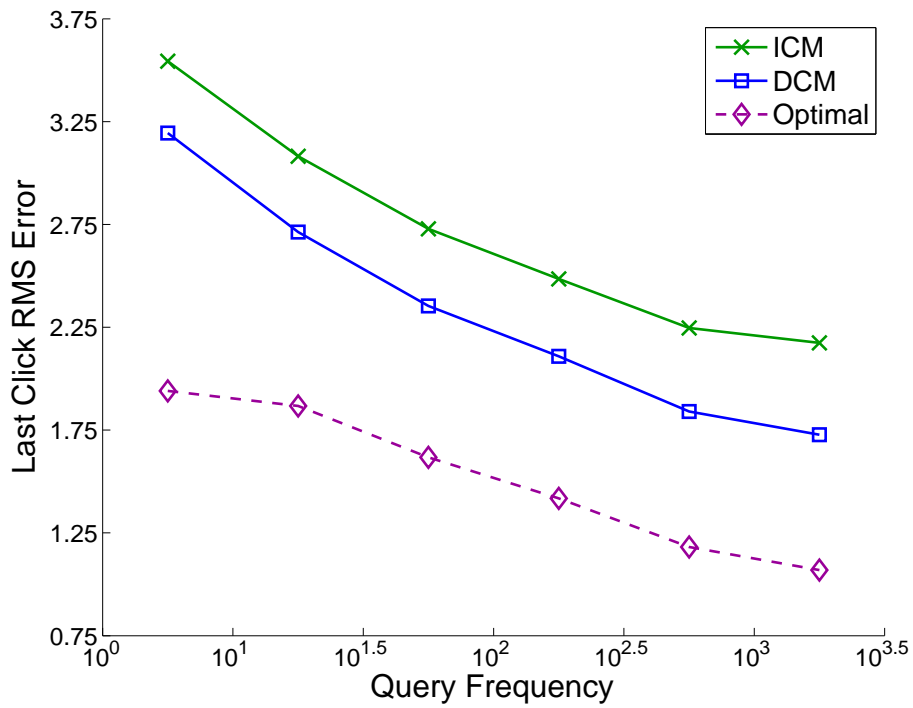
Figure 6.5: Examine probabilities implied by DCM for different query frequencies. Queries are grouped into 6 frequency ranges similarly as in Table 6.1. Darker and lower curves correspond to more frequent queries.

than ICM, reducing the RMS error gap from the optimal curve by around 30%.

Figure 6.7 illustrates generally slower than geometric decrease with the position for the empirical probabilities of both first and last clicks. DCM matches these probabilities very well at the top 5 positions. The higher tail of empirical curves is probably due to user scrolling behaviors, especially for informational queries which have a higher click through rate. And we suspect that users may examine documents in a different fashion when they scroll to the bottom of the search result page, so that the 10th position receives even more last clicks than the two above. However, they contribute to a fairly small fraction of overall results: clicks after position 6 represent only 6.1% of the total number. For ICM samples, documents that appears in lower positions may receive more clicks than the ground truth because of the position-independent assumption. This results in over-estimation of last click probabilities for these positions in Figure 6.7(b). On the other hand, the document relevance estimates in ICM is smaller than those in DCM, due to a larger denominator in computing the empirical probabilities. This under-estimation has a more significant effect on documents which usually appear in lower positions and after the last clicked position. Therefore, the first click probability distribution derived from ICM has a lower tail than the empirical curve, as shown in Figure 6.7(a).

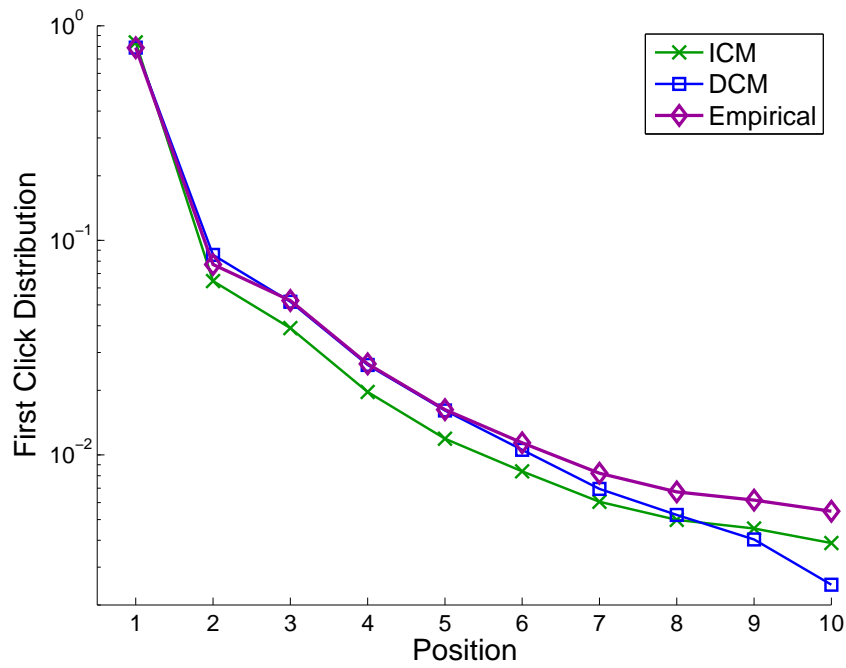


(a)

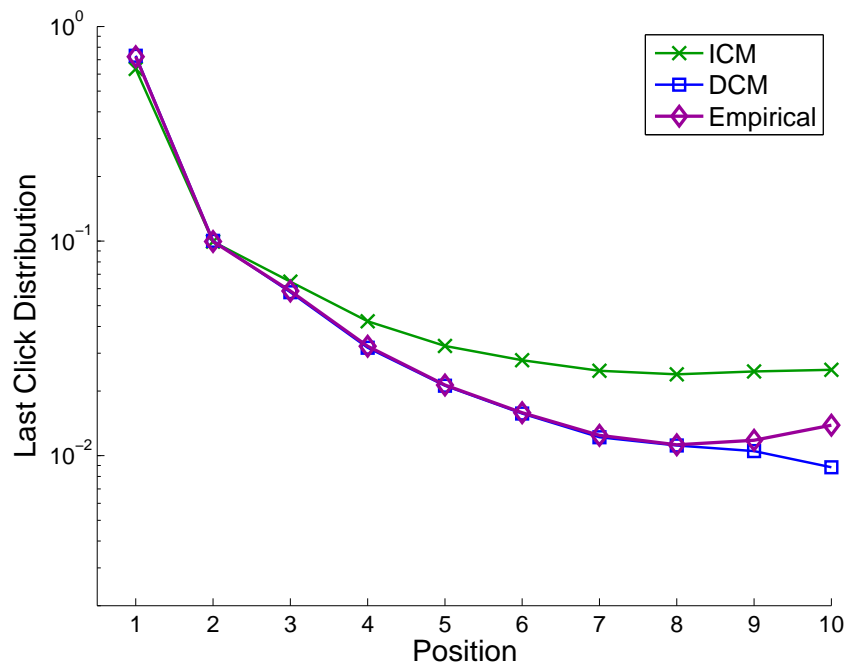


(b)

Figure 6.6: Root-mean-square (RMS) errors for predicting (a) first clicked position and (b) last clicked position. Results are averaged over 100 samples per query session.



(a)



(b)

Figure 6.7: First click distribution (a) and last click distribution (b) obtained by drawing samples from DCM and ICM given document impression. The overall first/last click distribution of DCM samples matches the empirical distribution in the test set very well, particularly for top 5 positions. Results are averaged over 100 samples per query session.

6.4 Related Work

6.4.1 Click Log Analysis and Learning to Rank

One of the earliest publications on large scale query log analysis appeared in 1999 [100], which presented interesting statistics as well as a simple correlation analysis from the Alta Vista search engine. Thereafter, search logs, especially the click-through data, have been utilized for a spectrum of search-related applications, and for learning to rank in particular.

Joachims [60] presented a pioneering study to exploit clickthrough data for optimizing the ranking function for search engines. Pairwise preference feedback, such as web document i is more relevant as web document j , are extracted from click logs and used to train a ranking support vector machine (ranking SVM) to output a retrieval function most concordant with these partial orderings. It was extended by Radlinski et al. [92], and an algorithm was proposed to detect a sequence of reformulated queries from the same user to learn an improved function. Radlinski et al. [93] followed this line of study for optimizing ranking functions but takes an alternative active-learning approach to control documents presented to users in search result pages for obtaining more helpful feedback as the next-round training data.

Xue et al. [115] proposed to use clickthrough data to improve graph-based static ranking algorithms. Bilenko et al. [15] presented a novel study in identifying “search trails” from user activity logs and used a random-walk based algorithm for improved retrieval accuracy. In [23] Carterette et al. proposed a logistic model for relevance prediction using scores obtained from human judges.

Clickthrough data could be also combined with other implicit measures or browsing data available from query logs to improve web search. The studies by Agichtein et al. proposed to extract a spectrum of features from browsing and click activities as well as textual data to train a better ranker [3] and estimate user preference [4]. An earlier work in evaluating these implicit measures appeared in [40]. Note that these additional information may not be able to be collected everyday due to the huge search volume. And it may also be subject to high level of noise, *e.g.*, web page dwelling time may be inaccurate if a user locks the screen to have a break with the browser open.

6.4.2 User Behavior Study and Click Models

A central task in utilizing search log is to understand and model user search and browsing behaviors and click decision processes. Joachims and his collaborators pioneered this direction by presenting a series of studies around some eye-tracking experiments [61, 62], which inspired a series of models that interpret user behaviors with increasing capacity, namely, the cascade model [33] already discussed, and the user browsing model proposed by Dupret et al [37].

Following the proposal of DCM, more studies have borrowed the Bayesian framework and techniques from probabilistic graphical models to develop more sophisticated alternatives with dedicated user behavior assumptions. This includes the click chain model [50], dynamic Bayesian network model [27], Bayesian browsing model [72]. Despite the improvement over click pre-

diction and extended power granted by the Bayesian modeling [73], they share a few simplifying assumptions such as homogenous treatment of different query sessions and are not without limitations in a number of important aspects, such as personalization, query reformulation, tail queries / data sparsity, and multimedia search results. Most recent developments and generalizations in this area has started to address these challenges from different perspectives.

Matthijs and Radlinski [77] presented a novel personalization approach for building user profiles based their past behavior to help re-rank future search results in better alignments with their preferences. Their implementation is publicly available as a browser add-on with detailed documentation [76]. Dupret and Liao [36] proposed the session utility model to characterize how users satisfy their information needs in a series of query submissions. Their analysis showed favorable results for informational queries and other type of queries which share a pattern that involves a relatively large number of reformulation and resubmission. Zhu et al. [121] introduced a generalized click model with the idea that relevance in the click model could be a function of multiple predicative input attributes (features), which effectively adds a regression-like component. The attributes learned are feature-specific instead of query-specific, and the model is expected to perform well with tail queries.

6.5 Conclusion and Discussion

Web search click logs record and aggregate important implicit feedbacks from user browsing and click activities, representing one of the most extensive, yet indirect, surveys on user experience. They are valuable resources for both information retrieval researchers, to better understand human interaction with retrieval results and calibrate their hypotheses or models, and web search practitioners, to measure, monitor and learn to improve search engine performance. A key challenge in click log analysis is to obtain accurate, efficient interpretation of user clicks, despite the fact that they are “informative but biased” as absolute relevance judgments, as demonstrated in a number of previous studies.

Click models usually incorporate a statistical depiction of user interaction with web search results in a query session, by specifying probabilities of examination and clicks at different positions and how they depend on each other. They provide principled solutions, scaling up to terabyte/petabyte scale click data, to inferring user-perceived relevance of web documents, and modeling outputs could be further leveraged in various search-related applications including search engine quality evaluation and sponsored search auctions. The idea and principle apply to search interfaces for multimedia search (better known as federated search) as well, with the introduction of additional types of bias over user examination and click probabilities reflecting different presentation styles (*e.g.*, images and videos).

This chapter provides a self-contained discussion of click models employing the dependent-click model as the running example. Trade-offs in model design and choices of user behavior assumptions should be dependent on specific application settings. DCM serves as an easy and efficient example that would be quite convenient for fast-prototyping and generally performs well with abundant data. A brief coverage of click chain model, featuring Bayesian statistical

techniques and more dedicated user model, is left to Chapter A in the appendix. This field has attracted a lot of interests from academic and industrial researchers within the community of Web Search and Data Mining since 2009, with a number of exciting new ideas that will further push forward the boundary of the state-of-the-art to be expected next year.

Small ideas and subtle implementation details could also play a dramatic role in addition to novel ideas and models from academic research. For example, tracking and logging how long users spend on the landing page after a click may provide informative signals and lead to a different treatment for very short clicks. Also, for pages that are “quickly viewed and reloaded” [39], these short impressions could well be eliminated at all.

Part IV
Conclusion

Chapter 7

Concluding Remarks

Having elaborated on the motivation, related literatures, algorithmic details and experimental evaluation for each study in the preceding five chapters, we bring this document to a close by reiterating major contributions under the theme of mining and querying, with a sketch of future directions that comes in sequel.

7.1 Mining Multimedia Data

Chapter 2 presented *QMAS* in the context of satellite image analysis. The discussion started off with a dedicated multi-scale feature extraction procedure from image tiles. An efficient subspace clustering algorithm was introduced to capture the similarity between tiles, achieving *over 40 times speed-up* over a prevailing implementation of approximate nearest-neighbor finding without any expense of quality. Low-labor labeling was made possible by constructing a three-layer graph based on clustering outputs, and executing a slight variation of random walk with restart algorithm. It provided high quality labeling results, even with tiny sets of pre-labeled data as inputs. It could also *spot top representatives and outliers* and offered a compact summarization of a large data set. The implementation was also employed to perform a set of practical queries over a proprietary data set by domain experts and it yielded quite positive results – correct labeling of objects where the traditional automated target recognition (ATR) approach may fail.

Chapter 3 introduced *MultiAspectForensics*, a handy tool to automatically detect and visualize *novel subgraph patterns within a local community of nodes in a heterogeneous network*, such as a set of vertices that form a dense bipartite graph whose edges share exactly the same set of attributes. It was effective even if such patterns exist among less-well connected nodes which are very likely to be ignored by many extant methods. Empirical results exhibited valuable insights derived from pattern discovered, across multiple application domains such as network traffic monitoring, knowledge networks, and bioinformatics. These successes could be attributed to the fact that we resorted to a tensor-based representation to facilitate data decomposition, reached a key observation leading to spike patterns in histogram plots, and revealed typical substructures reflecting spectral properties of heterogeneous data.

I hasten to point out that although both multimedia data mining studies briefed hereinabove

purport to adopt network representation of the multi-aspect data in a more or less similar way, their approaches are not alike. In the former scenario, the graph consists of multiple layers of node, each of which is attributed to one aspect of the data, and the structural information is made salient by the inter-layer links, *e.g.*, *a priori* curated labeling inputs were transformed into edges connecting their respective nodes in the image layer and those in the layer of annotation vocabulary. The labeling problem could be conveniently translated to cross-modal proximity query. And for the latter piece, the heterogeneity lies on the edge level, *i.e.*, edges in the graph may carry one or more attributes, which renders tensor-based representation a natural solution and spectral analysis rather straightforward to carry out. Hence, we hope that discussion in this paragraph could shed some light on the interdependence among the mining task, the choice of data representation, and subsequent algorithmic design.

7.2 Querying Multimedia Data

Chapter 4 described *CDEM*, an online interface to assist biological researchers to perform flexible querying and exploration over a large database which consists of embryo images, image annotations, as well as genes whose expression patterns are illustrated by these images. The data representation scheme is closely aligned with that in Chapter 2.

Chapter 5 provided a Bayesian approach to inference and learning with the exponential family harmonium (EFH) models and their variants for latent semantic projection of multimedia documents for subsequent data mining tasks such as classification and retrieval. The technique differed from previous chapters in that the input data are recognized as snapshots, or observations, of abstracted random variables following pre-assumed probability distributions, probably with parameters yet to be estimated. The data heterogeneity is accounted in the model setup, specifically in the selection of appropriate distributions, *e.g.*, binary word occurrence feature corresponds to a Bernoulli distribution, word counts may follow a Poisson one, whereas image features could be fit with Gaussian.

Chapter 6 served as a short tutorial of click models, with *DCM* as an introductory example. Click models have attracted a lot interests from academic researchers and industrial practitioners since just a few years ago. The studies highlighted in this manuscript are among the first few papers on this topic and represented state-of-the-art at the time of publication. They provide principled solutions to obtain accurate interpretation of user clicks as they interact with Web search results, *to measure, monitor and learn to improve search engine performance*. Our proposed models represented state-of-the-art along this line of research, scaled up to terabyte/petabyte size of click data, and have been further leveraged in various search-related applications including search engine quality evaluation [49] and post-rank reordering [73]. The idea and principle are ready to be applied to search interfaces for multimedia databases as well.

An interesting observation is that compared with studies covered in the previous three chapters takes a quite user-centric perspective, while those on the topic of mining addresses the need of data owner to take advantage of their assets. Moreover, it is worth noting that exact inference for most probabilistic graphical models in practice, including those proposed in Chapters 5

and 6, is intractable, and when there are multiple approximation alternatives, specific application requirements usually dictate the trade-off between quality and scalability. For instance, Web search click models are desired to be both single-pass and incremental, to avoid the potentially high cost to retain and revisit old data, and adapt to new trends without much effort.

7.3 Discussion and Future Directions

Studies presented in this thesis represents a miniature of diverse tasks, goals, design constraints, algorithms, heuristics, and experimental methods belated to multimedia data analysis in practice. We sought to achieve an appropriate trade-off between the two aspects of performance – quality and scalability. It is not uncommon in the design of a real mining and querying system, different components will be constrained in dissimilar ways. For example, expensive tensor decomposition algorithms could be applied to build an offline index for a recommendation system, whereas online ranking adjustments should be carried out in a much more efficient fashion. To rank thousands or millions of candidates for a given query, cheap and quick heuristics could be implemented at the indexing layer, whereas the final ranking layer, which receives a small subset of more relevant candidates, could afford more dedicated models.

The constantly evolving scale, genre and ubiquity of multimedia data brings up challenges and opportunities into the fast developing field of research and practice of multimedia mining and querying, with a quite incomplete list of questions highlighted as follows:

- Can we better leverage the distributed computing frameworks to grant greater scalability to existing solutions? The PEGASUS system [88] serves as an excellent example along this direction.
- Can we make available generic implementations of state-of-the-art algorithms to stimulate cross-disciplinary impacts? This may seem tedious at first but would probably lead to long-term benefits.
- How to evaluate the effectiveness of a mining algorithm for novel pattern discovery beyond validating anecdotal evidence? In particular, in what case is crowd-sourcing a reliable alternative when domain expertise is inaccessible or prohibitive?

Recall the opening example in this thesis illustrating the emerging trend of mobile and social in the beginning of the current decade, the quest of better vehicles to boosting the bandwidth and quality of information flow via computation, in particular, improved recommendation and ranking of multimedia units across different platforms and contexts, would always be exciting problems to be working on, which I'd like to pursue as part of my future career.

Part V
Appendix

Appendix A

Click Chain Model

A.1 Introduction

Web click are informative but biased. In order to neutralize various biases, numerous click models have been developed as a principled approach to inferring user-perceived relevance of web documents. Furthermore, as search logs can easily run over terabytes into petabytes and usually comes in a data stream on a daily basis, we would ideally expect click models to be amenable to click data streams, which essentially means scalability and the capability to be incrementally updated.

In this part of the appendix, we present the *Click Chain Model* (CCM), which features a solid Bayesian foundation and great scalability. In particular, document relevance are represented as random variables in the model, and a closed-form solution to the relevance posterior is derived from the proposed approximate inference scheme. Based on a data set containing 8.8 million query sessions, we show that CCM consistently outperforms two state-of-the-art competitors in a number of metrics, with over 9.7% better log-likelihood, over 6.2% better click perplexity, and much more robust (up to 30%) prediction of the last clicked position.

A.2 Background

A web search user initializes a *query session* by submitting a *query* to the search engine. We regard re-submissions and reformulations of the same query as distinct query sessions. We use *document impression* to refer to the *web documents* (or URLs) presented in the first result page, and discard other page elements such as sponsored ads and related search. The document impression can be represented as $\mathbf{d} = \{d_1, \dots, d_M\}$ (usually $M = 10$), where d_i is an index into a set of documents for the query, *i.e.*, d_1 denotes the document shown on the top position. A document is in a higher *position* (or rank) if it appears before those in lower positions.

Examination and click events for impressed documents are treated in a probabilistic way. For a given query session, we use binary random variables E_i and C_i to represent the examination event and the click event at position i respectively. Therefore, $P(E_i = 1)$ and $P(C_i = 1)$ are the

examination probability and the click probability at the same position.

The examination hypothesis [94], cascade hypothesis and the cascade model [33] have already been introduced in Section 6.1.2 where the dependent click model [51] is presented. The remainder of this section is devoted to another click model known as the *user browsing model* [37].

The user browsing model, or UBM, is also based on the examination hypothesis, but does not follow the cascade hypothesis. Instead, it assumes that the examination probability E_i depends on both i and the previous clicked position $l_i = \operatorname{argmax}_{l < i} \{C_l = 1\}$:

$$P(E_i = 1 | C_1, \dots, C_{i-1}) = \gamma_{i, l_i}. \quad (\text{A.1})$$

If there is no click before i , then $l_i = 0$, therefore $0 \leq l_i < i \leq M$. The total number of user behavior parameters is $M(M + 1)/2$, which are again shared by all query sessions.

Under UBM, the log-likelihood for each query session is

$$\ell(\gamma) = \sum_{i=1}^M \left(C_i \log r_{d_i} + C_i \log \gamma_{i, l_i} + (1 - C_i) \log(1 - r_{d_i} \gamma_{i, l_i}) \right) \quad (\text{A.2})$$

The coupling of relevance r and parameter γ introduced in the second term makes exact computation intractable. The algorithm could alternatively be carried out in an iterative, coordinate-ascent fashion. However, we found that the fix-point equation update proposed in [37] does not have convergence guarantee and is sub-optimal in certain cases. Instead, we designed the following update scheme which takes a few dozen iterations before achieving convergence according to our evaluation over a real-world click log data set.

Given a query for each document d , we keep its count of click K_d and non-click $L_{d,j}$ where $1 \leq j \leq M(M + 1)/2$, and they map one-to-one with all possible γ indices, and $1 \leq d \leq D$ maps one-to-one to all query-document pair. Similarly, for each γ_j , we keep its count of click K_j . Then given the initial value of $\gamma = \gamma^0$, optimization of relevance can be carried out iteratively,

$$r_d^{t+1} = \operatorname{arg\,max}_{0 < r < 1} \left\{ K_d \log r + \sum_{j=1}^{M(M+1)/2} L_{d,j} \log(1 - r \gamma_j^t) \right\} \quad (\text{A.3})$$

$$\gamma_j^{t+1} = \operatorname{arg\,max}_{0 < \gamma < 1} \left\{ K_j \log \gamma + \sum_{d=1}^D L_{d,j} \log(1 - r_d^{t+1} \gamma) \right\} \quad (\text{A.4})$$

for all possible d and j respectively. The “arg-max” operation can be carried out by evaluating the objective function for a sequential scan of r or γ values between 0 and 1. And we suggest a granularity of 0.01 for both scans.

A.3 Click Chain Model

The CCM model is based on the generative process of user interaction illustrated in the form of a flowchart in Figure A.1. The user initiates the examination of the search result in each

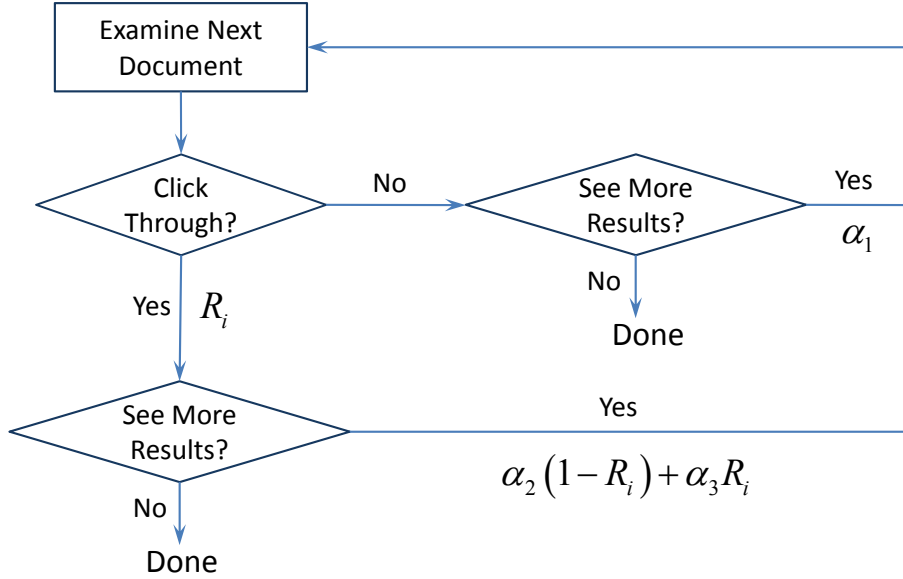


Figure A.1: Our proposed user model of CCM, in which R_i is the relevance variable of d_i at position i , and α 's form the set of user behavior parameters.

query session from the top ranked document. At each position i , the user can choose to click or skip the document d_i according to the perceived relevance R_i . Since R_i is a random variable under CCM, we can draw a sample from the distribution of R_i to determine the chance of click. Whatever the outcome for this click decision, the user can choose to continue the examination or abandon the current query session. The probability of continuing to examine d_{i+1} depends on the click decision at position i . Specifically, if the user skips d_i , this probability is an (unknown) constant α_1 ; on the other hand, if the user clicks d_i , the probability to examine d_{i+1} depends on the user-perceived relevance R_i and range between α_3 and α_2 .

CCM shares the following assumptions with the cascade model and DCM: (1) users are homogeneous: their information needs are similar given the same query; (2) decoupled examination and click events: the click probability is solely determined by the examination probability and the document relevance at a given position; (3) cascade examination: examination is in strict sequential order without breaks.

CCM distinguishes itself from other click models by representing document relevance as random variables and performing (approximate) Bayesian inference to infer their posterior distributions. Model fitting of CCM includes both inferring the posterior distribution of R_i and estimating user-behavior parameters $\alpha = \{\alpha_1, \alpha_2, \alpha_3\}$. The posterior distribution could be further leveraged for applications such as automated ranking alterations because it is possible to derive confidence measures and other useful features using standard statistical techniques [73].

The graphical model of CCM for one query session is shown in Figure A.2. There are three layers of random variables: for each position i , E_i and C_i denote examination and click events as usual, whereas R_i is the user-perceived relevance of d_i . The click layer is fully observed given

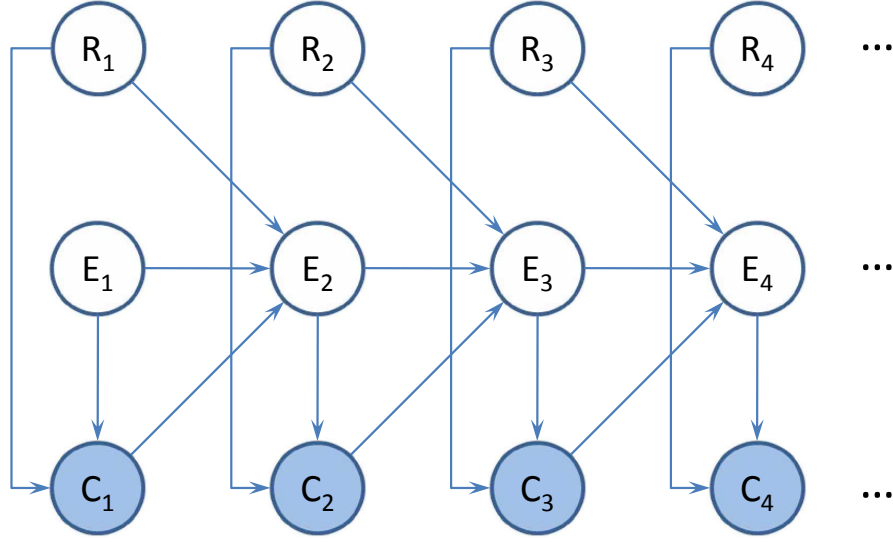


Figure A.2: The graphical model representation of CCM. Shaded nodes are observed click variables.

the click log. CCM is named after the chain structure of variables representing the sequential examination and clicks through every position in the search result.

The following conditional probabilities are defined for Figure A.2 according to the modeling assumption:

$$\begin{aligned}
 P(C_i = 1 | E_i = 0) &= 0 \\
 P(C_1 = 1 | E_1 = 1, R_1) &= R_1 \\
 P(E_{i+1} = 1 | E_i = 0) &= 0 \\
 P(E_{i+1} = 1 | E_i = 1, C_i = 0) &= \alpha_1 \\
 P(E_{i+1} = 1 | E_i = 1, C_i = 1, R_i) &= \alpha_2(1 - R_i) + \alpha_3 R_i
 \end{aligned} \tag{A.5}$$

To complete the model specification we let $P(E_1 = 1) = 1$ and impose independent uniform priors over $[0, 1]$ for every document relevance variable R_i , *i.e.*, $p(R_i) = 1$ for $0 \leq R_i \leq 1$. Note that in this model specification, we are not putting a limit on the length of the chain structure. Instead we allow the chain to go infinitely. We will discuss, in the next section, that this choice simplifies the inference algorithm, provides an order of magnitude savings in space, and sacrifices little performance since the chance of examination diminishes exponentially after the last click. An alternative is to truncate the click chain to a finite length of M . The single-pass inference and learning algorithms detailed hereinafter for the (standard) CCM could be adapted to this truncated version, to offer more accurate characterization of the posterior when the last clicked position is close to M .

A.4 Algorithms

This section describes inference and learning algorithms for the CCM. We start with the algorithm for computing the posterior distribution over the relevance of each document in Sections A.4.1 and A.4.2. Then we describe how to estimate the α parameters in Section A.4.3. We will also provide a brief discussion in Section A.4.2 to show that by keeping appropriate counts as sufficient statistics, it is straightforward to extend these algorithms to incrementally update the relevance posteriors and α parameters with newly available data.

Given the click data $\mathcal{C} = \{\mathbf{C}^1, \dots, \mathbf{C}^U\}$ from U query sessions for the query of interest, we want to compute the posterior $p(R_i|\mathcal{C})$ for the relevance of a document i . For a single query session, this posterior can be efficiently computed due to the chain structure of the graphical model (Figure A.2), and the distribution function is polynomial with respect to R_i . However, given multiple query sessions, exact inference becomes intractable due to sharing of relevance variables between query sessions. For example, if documents i and j may both appear in two sessions at different positions, their posteriors are generally dependent on each other. To carry out approximate inference, we could resort to off-the-shelf iterative algorithms such as expectation propagation [79]. However, to scale the model to terabyte data, we propose a faster method that requires only a single pass.

The approximation is that, when computing the posterior for R_i , we assume that the clicks in query sessions are conditionally independent given R_i :

$$p(R_i|\mathcal{C}) \approx (\text{constant}) \times p(R_i) \prod_{u=1}^U P(\mathbf{C}^u|R_i). \quad (\text{A.6})$$

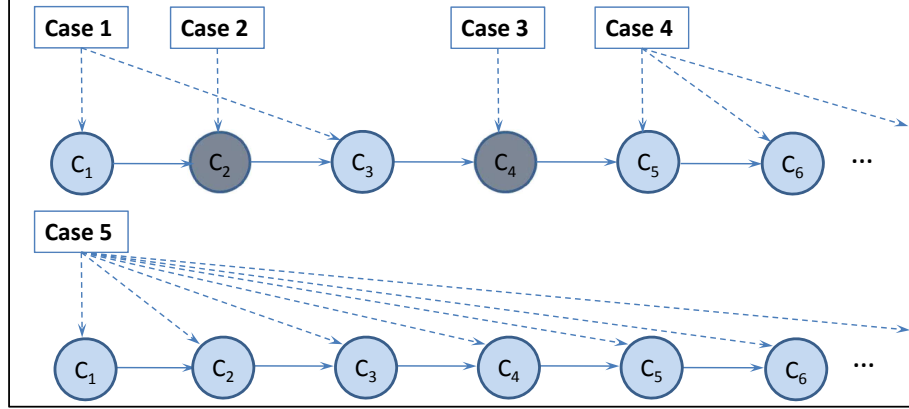
Since the prior $p(R_i)$ is given, we only need to compute $P(\mathbf{C}^u|R_i)$ for each query session u , up to a constant w.r.t. R_i , in order to obtain an un-normalized version of the posterior. Section A.4.1 will elaborate on the computation of this conditional probability $P(\mathbf{C}^u|R_i)$, and the superscript u is omitted in the following as we focus on a single query session.

A.4.1 Deriving the Conditional Probabilities

Before diving into the details of deriving $P(\mathbf{C}|R_i)$ for a particular session, we first highlight the following three properties of CCM, which will greatly simplify the variable elimination procedure as we take sum or integration over all hidden variables other than R_i :

1. If $C_j = 1$ for some j , then $\forall i \leq j, E_i = 1$.
2. If $E_j = 0$ for some j , then $\forall i \geq j, E_i = 0, C_i = 0$.
3. If $E_i = 1, C_i = 0$, then $P(E_{i+1}|E_i, C_i, R_i) = \alpha_1^{E_{i+1}}(1 - \alpha_1)^{1-E_{i+1}}$ does not depend on R_i .

Property (1) states that every document before the last clicked position is examined, so for these documents, we only need take care of different values of random variables within its Markov blanket in Figure A.2, which consists of C_i, E_i and E_{i+1} . Property (2) is a corollary from the cascade hypothesis, and it reduces the cost of eliminating E variables from exponential time to linear time by using branch-and-conquer. Property (3) is derived from the model specification



Case	Conditions	Results
1	$i < l, C_i = 0$	$1 - R_i$
2	$i < l, C_i = 1$	$R_i(1 - (1 - \alpha_3/\alpha_2)R_i)$
3	$i = l$	$R_i \left(1 + \frac{\alpha_2 - \alpha_3}{2 - \alpha_1 - \alpha_2} R_i\right)$
4	$i > l$	$1 - \frac{2}{1 + \frac{6 - 3\alpha_1 - \alpha_2 - 2\alpha_3}{(1 - \alpha_1)(\alpha_2 + 2\alpha_3)} (2/\alpha_1)^{(i-l)-1}} R_i$
5	No Click	$1 - \frac{2}{1 + (2/\alpha_1)^{i-1}} R_i$

Figure A.3: Different cases for computing $P(C | R_i)$ up to a constant where l indicates the last clicked position. Darker nodes in the figure above represent clicks at these positions, whereas lighter nodes represent skips.

(Eq. A.5), and it enables re-arrangement of the sum operation over E and R variables to minimize the computational effort.

From property 1, we know that the last clicked position $l = \operatorname{argmax}_{1 \leq i \leq M} \{C_i = 1\}$ plays an important role. Figure A.3 lists the complete results separated into two categories, depending on whether there is any click in the current query session.

Derivation for each case is presented in the following:

Case 1: $i < l, C_i = 0$

By property 1 and property 3, $E_i = 1, E_{i+1} = 1$ and $P(E_{i+1} = 1 | E_i = 1, C_i = 0, R_i) = \alpha_1$ does not depend on R_i . Since any constant w.r.t. R_i can be ignored, we have

$$P(C | R_i) \propto P(C_i = 0 | E_i = 1, R_i) = 1 - R_i \quad (\text{A.7})$$

Case 2: $i < l, C_i = 1$

By property 1, $E_i = 1, E_{i+1} = 1$, the Markov blanket of R_i consists of C_i, E_i and E_{i+1} .

$$\begin{aligned} P(C | R_i) &\propto P(C_i = 1 | E_i = 1, R_i) P(E_{i+1} = 1 | E_i = 1, C_i = 1, R_i) \\ &\propto R_i(1 - (1 - \alpha_3/\alpha_2)R_i) \end{aligned} \quad (\text{A.8})$$

Case 3: $i = l$

By property 1, the Markov blanket of R_i does not contain any variable before position i , and we also know that $C_i = 1, E_i = 1$ and $\forall j > i, C_j = 0$. We need to take sum/integration over all the E_j, R_j variables where $j > i$ and it can be performed as follows:

$$\begin{aligned}
P(\mathbf{C} | R_i) &\propto P(C_i = 1 | E_i = 1, R_i) \cdot \sum_{\{E_j | j > i\}} \int_{\{R_j | j > i\}} \prod_{j > i} \\
&\quad \left(P(E_j | E_{j-1}, C_{j-1}, R_{j-1}) \cdot p(R_j) \cdot P(C_j | E_j, R_j) \right) \\
&= R_i \cdot \left\{ P(E_{i+1} = 0 | E_i = 1, C_i = 1, R_i) \cdot 1 \right. \\
&\quad + P(E_{i+1} = 1 | E_i = 1, C_i = 1, R_i) \cdot \\
&\quad \quad \left(\int_0^1 p(R_{i+1}) P(C_{i+1} = 0 | E_{i+1} = 1, R_{i+1}) dR_{i+1} \right) \cdot \\
&\quad \quad \left\{ P(E_{i+2} = 0 | E_{i+1} = 1, C_{i+1} = 0) \cdot 1 \right. \\
&\quad \quad + P(E_{i+2} = 1 | E_{i+1} = 1, C_{i+1} = 0) \cdot \\
&\quad \quad \quad \left(\int_0^1 p(R_{i+2}) P(C_{i+2} = 0 | E_{i+2} = 1, R_{i+2}) dR_{i+2} \right) \cdot \\
&\quad \quad \quad \left. \left. \left. \dots \right\} \right\} \right\} \\
&= R_i \left((1 - \alpha_2(1 - R_i) - \alpha_3 R_i) + (\alpha_2(1 - R_i) + \alpha_3 R_i) \cdot \right. \\
&\quad \left. \frac{1}{2} \cdot \left((1 - \alpha_1) + \alpha_1 \cdot \frac{1}{2} \cdot (\dots) \right) \right) \\
&\propto R_i \left(1 + \frac{\alpha_2 - \alpha_3}{2 - \alpha_1 - \alpha_2} R_i \right) \tag{A.9}
\end{aligned}$$

Case 4: $i > l$

Now the result will be a function of $k = i - l$: the offset of the document from the last clicked position. The branch-and-conquer approach we take to sum over variables in the Markov blanket of R_i is similar to Case 3. The major difference in the equation below is that we are integrating the outmost R_l while leaving the R_i inside without the integration,

and we will replace the dummy variable R_l by R for easier reading:

$$\begin{aligned}
& P(\mathbf{C} | R_i) \\
& \propto \begin{cases} \int_0^1 R \left((1 - \alpha_2(1 - R) - \alpha_3)R + (\alpha_2(1 - R) + \alpha_3)R \cdot (1 - R_i) \cdot \frac{1 - \alpha_1}{1 - \alpha_1/2} \right) dR & (k = 1) \\ \int_0^1 R \left((1 - \alpha_2(1 - R) - \alpha_3)R + (\alpha_2(1 - R) + \alpha_3)R \cdot \left(\frac{1 - \alpha_1}{2} \sum_{j=0}^{k-2} (\alpha_1/2)^j + (\alpha_1/2)^{k-1} (1 - R_i) \frac{1 - \alpha_1}{1 - \alpha_1/2} \right) \right) dR & (k > 1) \end{cases} \\
& = \int_0^1 R \left((1 - \alpha_2(1 - R) - \alpha_3)R + (\alpha_2(1 - R) + \alpha_3)R \cdot \frac{(1 - \alpha_1)(1 + (\alpha_1/2)^{k-1} - 2(\alpha_1/2)^{k-1}R_i)}{2 - \alpha_1} \right) dR \\
& \propto 1 - \frac{2}{1 + \frac{6 - 3\alpha_1 - \alpha_2 - 2\alpha_3}{(1 - \alpha_1)(\alpha_2 + 2\alpha_3)} (2/\alpha_1)^{k-1}} R_i \tag{A.10}
\end{aligned}$$

Case 5: (No click)

When $i = 1$, we know $E_1 = 1$, therefore $P(E_2 | E_1 = 1, C_1 = 0)$ does not depend on R_1 . We have

$$P(\mathbf{C} | R_i) \propto P(C_1 = 0 | E_1 = 1, R_1) = 1 - R_1 \tag{A.11}$$

When $i > 1$ the derivation is similar to Case 4 and is much simpler since there is no α_2, α_3 term:

$$\begin{aligned}
P(\mathbf{C} | R_i) & \propto \int_0^1 (1 - R) dR \left((1 - \alpha_1) \cdot \sum_{j=0}^{i-2} (\alpha_1/2)^j + (\alpha_1/2)^{i-2} \alpha_1 (1 - R_i) \frac{1 - \alpha_1}{1 - \alpha_1/2} \right) \\
& \propto (1 - \alpha_1) \frac{1 - (\alpha_1/2)^{i-1}}{1 - \alpha_1/2} + 2(\alpha_1/2)^{i-1} (1 - R_i) \frac{1 - \alpha_1}{1 - \alpha_1/2} \\
& \propto 1 - \frac{2}{1 + (2/\alpha_1)^{i-1}} R_i \tag{A.12}
\end{aligned}$$

Eq. A.11 can be treated as a special case of Eq. A.12 when $i = 1$.

Both case 4 and case 5 need to take sum over latent variables after the current position i , and results depend on the distance from the last clicked position. Therefore the total number of distinct terms obtained in these 5 cases when $1 \leq i \leq M$ are $1 + 1 + 1 + (M - 1) + M = 2M + 2$. If we impose a finite chain length M on CCM and let $P(E_{M+1}) = 0$ in Figure A.2, then the number of distinct results would be $M^2 + 2$, which is an order of magnitude higher than the current design, and further increases the cost of subsequent steps of inference and parameter estimation.

A.4.2 Computing the Posterior

All the conditional probabilities in Figure A.3 for a single query session can be written in the form of $R_i^{C_i} (1 - \beta_j R_i)$ where β_j is a case-specific coefficient whose value depend on user behavior

Algorithm 3: Computing the Un-normalized Posterior

Input: Click Data \mathcal{C} ($M \times U$ matrix);

$C_{iu} = 1$ if user u clicks the document at position i

Impression Data \mathcal{D} ($M \times U$ matrix);

$D_{iu} = d$ if the document d is impressed at position i
to user u , $1 \leq d \leq N$

Parameters α

Output: Coefficients β ($(2M + 2)$ -dim vector)

Exponents \mathcal{T} ($N \times (2M + 2)$ matrix)

Compute β using the results in Figure A.3.

Initialize \mathcal{T} by setting all its elements to 0.

for $1 \leq u \leq U$

 for $1 \leq i \leq M$

 Identify the linear factors using the click and impression data,

 let β_j be the corresponding coefficient for the current position,

$T_{D_{iu},j} = T_{D_{iu},j} + 1$.

Time Complexity: $O(MU)$.

Extra Space: $O(MN)$.

(Usually $U > N \gg M = 10$.)

parameters α . Let M be the number of web documents in the first result page and it usually equals 10. There are at most $(2M + 2)$ different coefficients from the 5 cases as discussed above. From Eq. A.6, we know that the un-normalized posterior $p(R_i|\mathcal{C})$ is a polynomial function of R_i which consists of at most $(2M + 3)$ distinct linear factors (including R_i itself), so given user behavior parameters α , we only need to record $(2M + 2)$ exponents as sufficient statistics to fully characterize the posterior distribution for every query-document pair. The detailed algorithm for parsing through the click log and updating exponents is listed in Algorithm 3. Given output exponents \mathcal{T} as well as coefficients β from Algorithm 3, the un-normalized relevance posterior of a document indexed by d is

$$\tilde{p}_d(r) \propto r^{T_{d,2}+T_{d,3}} \prod_{j=1}^{2M+2} (1 - \beta_j r)^{T_{d,j}} \quad (\text{A.13})$$

Furthermore, when new click data becomes available, we can run Algorithm 3 to update exponents stored in \mathcal{T} by incrementing the counts for each query-document pair. The computational time is thus $O(MU')$, where U' is the number of query sessions in the new data. Extra space is only needed only when there are previously unseen web documents of interest.

Eq. A.13 gives the analytical formula of the un-normalized posterior as a polynomial function whose order depend on the number of impressions and clicks of the corresponding document. To evaluate the normalization constant, we could perform integration of $\tilde{p}_d(r)$ over $r \in [0, 1]$ in a straightforward way by sequentially multiplying all the linear factors to obtain every coefficient

Algorithm 4 : Numerical Integration for Posteriors

Input: Exponents \mathcal{T} ($N \times (2M + 2)$ matrix)
Coefficients β ($(2M + 2)$ -dim vector)
Number of bins B

Output: Posterior moments \mathcal{F} ($N \times K$ matrix)
 F_{dk} is the k th posterior moment for document d

Create a $N \times (K + 1)$ matrix $\tilde{\mathcal{F}}$ for intermediate results.
Create a B -dimensional vector r to keep the center of each bin:

$$r_b = (b - 0.5)/B \text{ for } 1 \leq b \leq B$$

for $1 \leq k \leq K + 1$

for $1 \leq d \leq N$

$$\tilde{F}_{dk} = \log \left(\sum_{b=1}^B \exp \left(-\log(B) + (T_{d2} + T_{d3} + k - 1) \cdot \log(r_b) \right. \right. \\ \left. \left. + \sum_{j=1}^{2M+2} T_{dj} \log(1 + \beta_j r_b) \right) \right).$$

for $1 \leq k \leq K$

for $1 \leq d \leq N$

$$F_{dk} = \exp \left(\tilde{F}_{d,k+1} - \tilde{F}_{d,1} \right)$$

Time Complexity: $O(KMN B)$.

Extra Space: $O(KN + MB)$.

of the polynomial function. However, the accuracy suffers significantly from rounding errors as the size of the input and the order of the polynomial functions goes up, with a more expensive computational cost as well. To finesse this numerical problem, we propose Algorithm 4 for computing posterior moments using the midpoint rule with B equal-size bins to approximate the integral and posterior moments. $B = 100$ is usually sufficient in most cases, though the number of bins could be adaptively set if necessary. Let

$$f_d(k) = \int_0^1 r^k \tilde{p}_d(r) dr \approx \frac{1}{B} \sum_{b=1}^B r_b^{T_{d,2} + T_{d,3} + k} \prod_{j=1}^{2M+2} (1 + \beta_j r_b)^{T_{d,j}}, \quad (\text{A.14})$$

then the estimation for the k th moment for document d is $f_d(k)/f_d(0)$. To avoid numerical problems in the implementation, we usually take sum of log values for the linear factors instead of multiplying them directly. And the above procedure could be extended to compute other posterior estimates depending on the application scenario.

A.4.3 Parameter Estimation

We perform approximate maximum-likelihood estimation to learn model parameters α . The approximation is based on the same assumption we proposed at the beginning of this section by

imposing independent priors for document relevance variables across query sessions, such that

$$P(\mathcal{C}) = \int p(\mathbf{R}) \prod_{u=1}^U p(\mathbf{C}^u | \mathbf{R}) d\mathbf{R} \approx \prod_{u=1}^U \int p(\mathbf{R}) p(\mathbf{C}^u | \mathbf{R}) d\mathbf{R}, \quad (\text{A.15})$$

where $\mathbf{R} = \{R_i | 1 \leq i \leq N\}$ is the set of document relevance variables for the query of interest. And to compute the log-likelihood function, we simply need to compute the last integral in the previous equation for each query session and sum over log values across the data set. The derivation of the value of the integral is analogous to that in Section A.4.1, and we could write the resulting approximate log-likelihood function as follows:

$$\begin{aligned} \ell(\boldsymbol{\alpha}) = & N_1 \log \alpha_1 + N_2 \log(\alpha_2 + 2\alpha_3) + N_3 \log(6 - 3\alpha_1 - (\alpha_2 + 2\alpha_3)) \\ & + N_5 \log(1 - \alpha_1) - (N_3 + N_5) \log(2 - \alpha_1) + (\text{constant}) \end{aligned} \quad (\text{A.16})$$

where N_i is the total number of times documents fall into case i in Figure A.3 in the click data.

By maximizing this approximate log-likelihood w.r.t. $\boldsymbol{\alpha}$, we have

$$\alpha_1 = \frac{3N_1 + N_2 + N_5 - \sqrt{(3N_1 + N_2 + N_5)^2 - 8N_1(N_1 + N_2)}}{2(N_1 + N_2)} \quad (\text{A.17})$$

and

$$\alpha_2 + 2\alpha_3 = \frac{3N_2(2 - \alpha_1)}{N_2 + N_3} \quad (\text{A.18})$$

Eq. A.18 leave a degree of freedom in the choice of α_2 and α_3 values, which is introduced by the *iid* uniform priors over all documents and the approximation scheme we took. We can assign a value to α_2/α_3 according to the context of the model application (more on this in experiments). Note that parameter values do not depend on N_4 when the chain length is infinite.

A.5 Performance Evaluation

In this section, we report on the performance evaluation and comparison based on a data set with 8.8 million query sessions that are uniformly sampled from a commercial search engine. We measure the performance of three click models with a number of evaluation metrics. Our results show that CCM consistently outperforms its competitors UBM and DCM with: (1) over 9.7% better log-likelihood (Section A.5.2); (2) over 6.2% improvement in click perplexity (Section A.5.3); (3) more robust (up to 30%) click statistics prediction (Section A.5.4). As these widely used metrics measure model performance from different aspects, the uniformly better results demonstrate that CCM robustly captures user clicks in a number of contexts. Finally, in Section A.5.5, we present the empirical examine and click distribution curves, which help illustrate the differences in modeling assumptions.

A.5.1 Experimental Setup

The experiment data set is identical to the one described in Section 6.3.1. For each query, we compute document relevance and position relevance based on each of the three models, respectively. Position relevance is computed by treating each position as a pseudo-document. The position relevance can substitute the document relevance estimates for documents that appear zero or very few times in the training set but do appear in the test set. This essentially smoothes the predictive model and improves the performance on the test set. The cutoff is set adaptively to $\lfloor 2 \log_{10}(\text{Query Frequency}) \rfloor$. For CCM, the number of bins B is set to 100 which already provides an adequate level of accuracy.

To account for heterogeneous browsing patterns for different queries, we fit different models for navigational queries and informational queries and learn two sets of parameters for each model according to the median of click distribution over positions [49, 71]. In particular, CCM sets the ratio α_2/α_3 equal to 2.5 for navigational queries and 1.5 for informational queries, because a larger ratio implies a smaller examination probability after a click. The value of α_1 equals 1 as a result of discarding query sessions with no clicks ($N_5 = 0$ in Eq. A.17).

To evaluate model performance on test data, we compute the log-likelihood and other statistics needed for each query session using the document relevance and user behavior parameter estimates learned/inferred from training data. In particular, by assuming independent document relevance priors in CCM, all the necessary statistics can be derived in closed form, as summarized in [50] (Appendix B). Finally, to avoid infinite values in log-likelihood, a lower bound of 0.01 and an upper bound of 0.99 are applied to document relevance estimates for DCM and UBM.

All experiments described in the remainder of this section were carried out on a 64-bit server with 32GB RAM and eight 2.8GHz cores with MATLAB 2008b installed. The total time of model training for UBM, DCM and CCM is 333 minutes, 5.4 minutes and 9.8 minutes, respectively. For CCM, obtaining sufficient statistics (Algorithm 3), parameter estimation, and posterior computation (Algorithm 4) account for 54%, 2.0% and 44% of the total time, respectively. For UBM, the learning algorithm converges in 34 iterations.

A.5.2 Results on Log-Likelihood

Log-likelihood (LL) is widely used to measure model fitness. Given the document impression for each query session in the test data, LL is defined as the log probability of observed click events computed under the trained model. A larger LL indicates better performance, and the optimal value is 0. The improvement of LL value ℓ_1 over ℓ_2 is computed as $(\exp(\ell_1 - \ell_2) - 1) \times 100\%$. We report average LL over multiple query sessions using arithmetic mean values.

Figure A.4 presents LL curves for the three models over different frequency categories. The average LL over all query sessions is -1.171 for CCM, which means that with 31% chance, CCM predicts user clicks exactly over top 10 positions (out of the 2^{10} possibilities) for a query session in the test data set. The average LL is -1.264 for UBM and -1.302 for DCM, with corresponding improvement ratios to be 9.7% and 14% respectively. Thanks to the Bayesian modeling of document relevance, CCM outperforms the other two models more significantly on less frequent

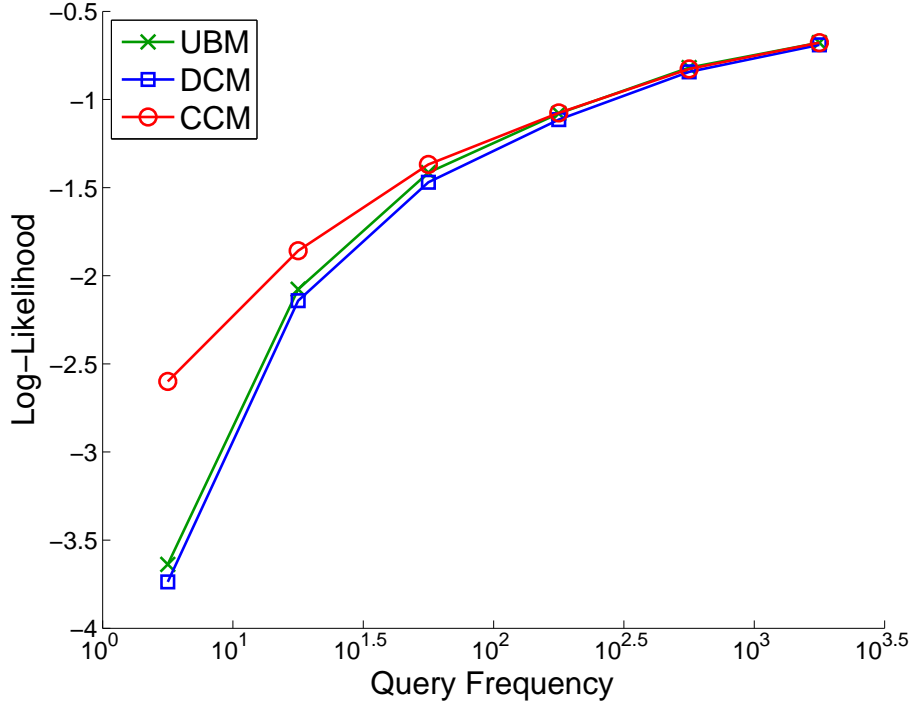


Figure A.4: Log-likelihood per query session on test data for different query frequencies. The overall average for CCM is -1.171, 9.7% better than UBM (-1.264) and 14% better than DCM (-1.302).

queries, as indicated in Figure A.4. Fitting different models for navigational and informational queries leads to 2.5% better LL for DCM compared with a previous implementation in [51] on the same data set (average LL = -1.327), which is consistent with our results [49] obtained from another data source.

A.5.3 Results on Click Perplexity

Click perplexity was used as the evaluation metric previously in [37]. It is derived from the entropy measure for binary click events at each position in a query session independently, whereas log-likelihood computation is based on the whole click vector. For a set of query sessions indexed by $n = 1, \dots, N$, if we use q_i^n to denote the probability of click derived from the click model on position i and C_i^n to denote the corresponding binary click event, then the click perplexity

$$p_i = 2^{-\frac{1}{N} \sum_{n=1}^N (C_i^n \log_2 q_i^n + (1 - C_i^n) \log_2 (1 - q_i^n))}. \quad (\text{A.19})$$

A smaller perplexity value indicates higher prediction quality, and the optimal value is 1. The improvement of perplexity values p_1 over p_2 is given by $(p_2 - p_1)/(p_2 - 1) \times 100\%$.

The average click perplexity over all query sessions and positions is 1.1479 for CCM, which gives a 6.2% improvement per position over UBM (1.1577) and a 7.0% improvement over DCM (1.1590). Again, CCM outperforms the other two models more significantly on less frequent

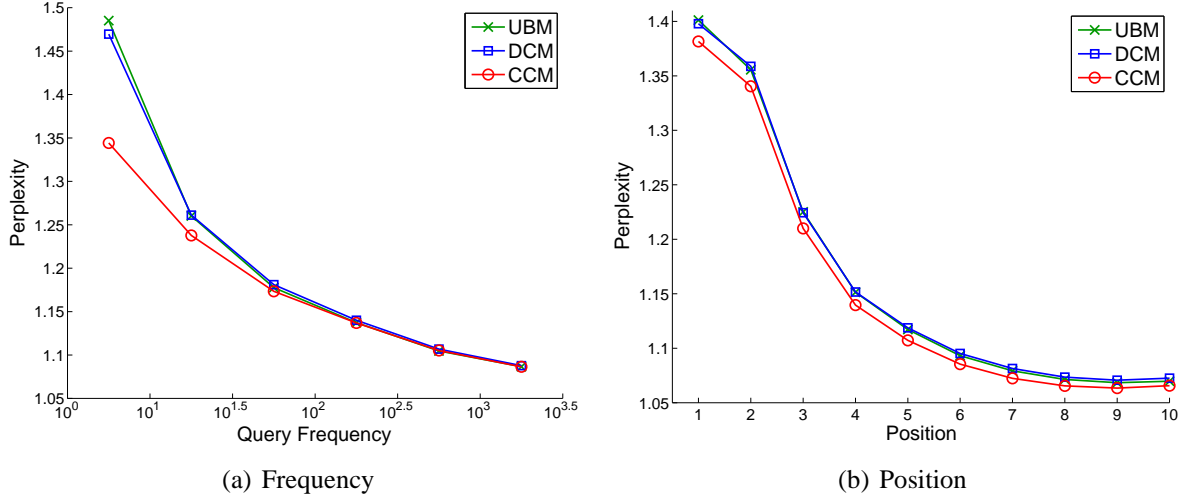


Figure A.5: Perplexity results on test data for (a) for different query frequencies or (b) different positions. Average click perplexity over all positions for CCM is 1.1479, 6.2% improvement over UBM (1.1577) and 7.0% over DCM (1.1590).

queries than on more frequent queries. As shown in Figure A.5(a), the improvement ratio is over 26% when compared with DCM and UBM on the least frequent query category. Figure A.5(b) illustrates that click prediction quality of CCM is the best of the three models for every position, whereas DCM and UBM are almost comparable with each other. Perplexity values for CCM on position 1 and position 10 equal the perplexity of predicting the outcome of tossing a biased coin of with known $p(\text{Head}) = 0.099$ and $p(\text{Head}) = 0.0117$ respectively.

A.5.4 Last-Click Prediction

Root-mean-square (RMS) error on click statistics prediction was used as an evaluation metric previously in [51]. It is calculated by comparing the observation statistics such as the first clicked position or the last clicked position in a query session with the model predicted position. Predicting the last clicked position is particularly challenging for click models while predicting the first clicked position is a relatively easy task. We evaluate the performance of last-click prediction on the test data under two settings and present results in Figure A.6.

First, given the impression data, we compute the distribution of the last clicked position in closed form, and compare the expected value with the observed statistics to compute the RMS error. The optimal RMS value under this setting is approximately the standard deviation of last clicked positions over all query sessions for a given query, and it is included in Figure A.6 as the “optimal-exp” curve. We expect that a model that gives consistently good fit of click data would have a small margin with respect to the optimal error. The improvement of RMS error values e_1 over e_2 w.r.t. the optimal value e^* is given by $(e_2 - e_1)/(e_2 - e^*) * 100\%$. We report average error by taking the RMS mean over all query sessions. The optimal RMS error under this setting for last clicked position is 1.443, whereas the error of CCM is 1.548, which is 9.8% improvement

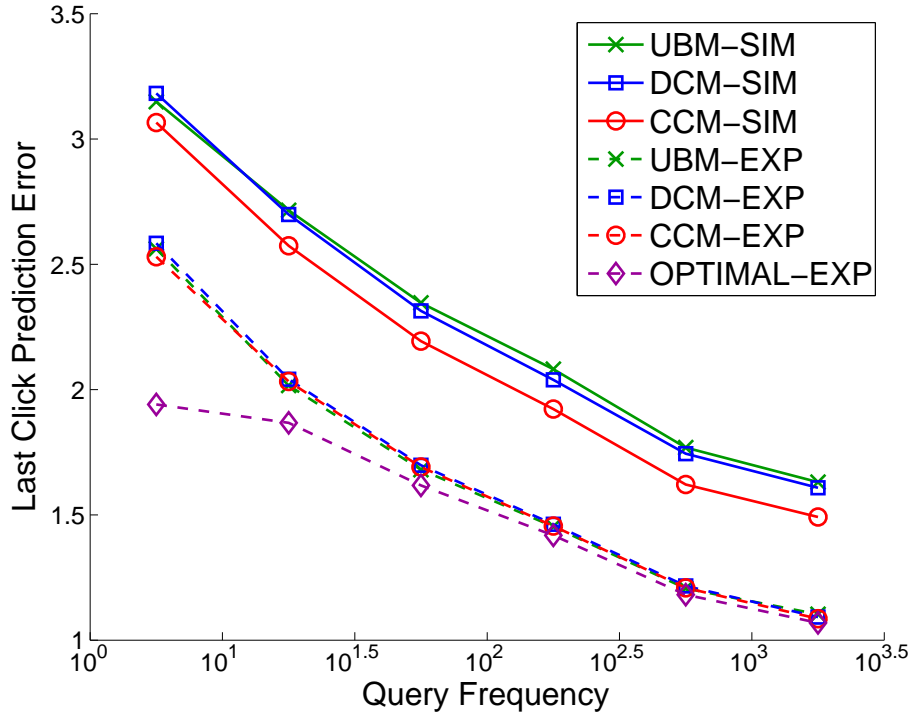


Figure A.6: Root-mean-square (RMS) errors for predicting the last clicked position. Prediction is based on the SIMulation for solid curves and EXPection for dashed curves.

for DCM (1.560) but only slightly better than UBM (0.2%).

Under the second setting, we simulate query session clicks from the model, collect those samples with clicks, compare the last clicked position against the ground truth in the test data and compute the RMS error, in the same way as [51]. The number of samples in the simulation is set to 10. The optimal RMS error is the same as in the previously discussed setting, but it is much more difficult to achieve this lower bound under the current setting because errors from simulations reflect both biases and variances of the prediction. We report the RMS error margin for the same model between the two settings as follows; a *more robust* click model should have a *smaller margin*. The error margin on last-click prediction, the margin is 0.460 for CCM, compared with 0.566 for DCM (23% larger) and 0.599 for UBM (30% larger).

In summary, CCM is the best of the three models in this experiment, to predict the first and the last clicked position effectively and robustly.

A.5.5 Position-bias of Examination and Click

Model click distribution over positions are the averaged click probabilities derived from click models based on the document impression in the test data. It reflects the position-bias implied by the click model and can be compared with the objective ground truth—the *empirical click distribution* over the test set. Any deviation of model click distribution from the ground truth would suggest the existing modeling bias in clicks. Note that on the other side, a close match

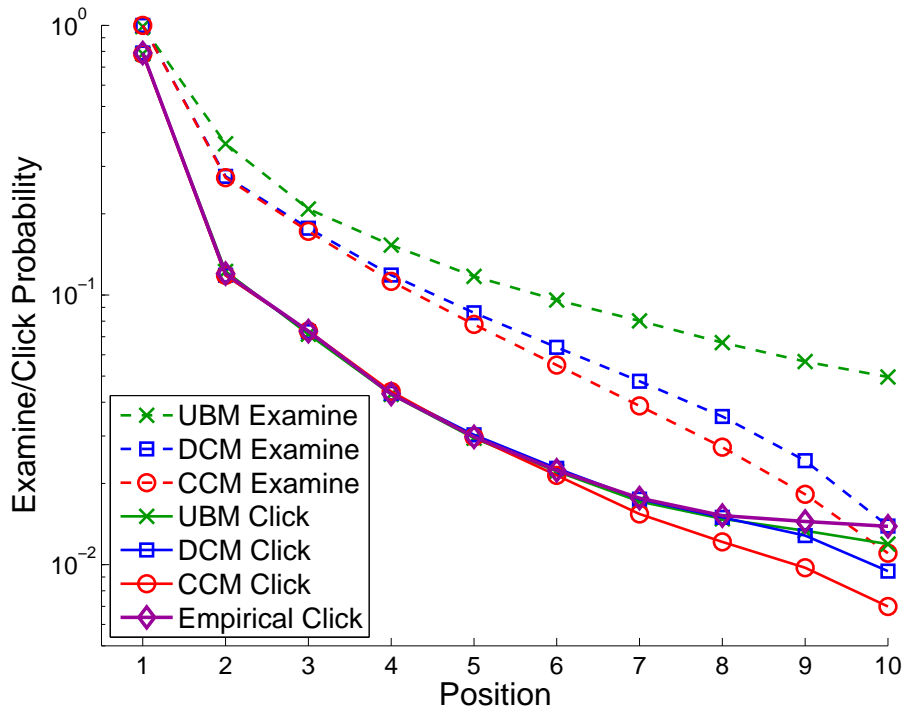


Figure A.7: Examination and click probability distributions over the top 10 positions.

does not necessarily imply excellent click prediction, for example, prediction of clicks $\{00, 11\}$ as $\{01, 10\}$ would still have a perfect empirical distribution. *Model examination distribution* over positions can be computed in a similar way to the click distribution. But there is no ground truth to be contrasted with. Under the examination hypothesis, the gap between examination and click curves of the same model reflects the average document relevance.

Figure A.7 illustrates the model examination and click distribution derived from CCM, DCM and UBM as well as the empirical click distribution on the test data. All three models underestimate the click probabilities in the last few positions. CCM has a larger bias due to the approximation of infinite-length in inference and estimation. This approximation is immaterial as CCM gives the best results in the above experiments. For certain applications that require very accurate modeling of the last few positions, the finite version of CCM could be employed instead.

Examination curves of CCM and DCM decrease with similar exponential rates. Both models are based on the cascade assumption, under which examination is a linear traverse through the rank. The examination probability derived by UBM is much larger, and this suggests that the absolute value of document relevance derived from UBM is not directly comparable to that from DCM and CCM. Relevance judgments from click models is still a relative measure.

A.6 Conclusion

We have discussed the click chain model (CCM), which features Bayesian modeling and inference of user-perceived relevance. Using an approximate closed-form representation of the relevance posterior, CCM is both scalable and incremental, perfectly meeting the challenges posed by real world practice. We carry out a set of extensive experiments with various evaluation metrics, and the results clearly evidence the advancement of the state-of-the-art.

Appendix B

RTW Knowledge Base Query Interface

The “Read the Web” research project at Carnegie Mellon University [95] has made available an ever-updating web knowledge base which is made up millions of structured beliefs like (Facebook, HeadquarteredIn, Palo_Alto). This part of the appendix describes the implementation of a more user-friendly online interface to ease the browsing process and perform simple proximity query. The data set herein consists of 0.4M promoted beliefs, a cleaner version of the raw data set in the same format as the aforementioned triplet, by courtesy of Bryan Kisiel at Carnegie Mellon University.

Figure B.1 offers a snapshot of the online interface, available at <http://www.db.cs.cmu.edu/wk/>. It supports two categories of user actions:

- *Fact Browsing*: list all facts for a given query word and its role, where each fact could be represented as a triplet of (entity, relationship, value). Figure B.2(a) highlights a browsing example.
- *Similarity Query*: list relevant information for a given a query word. A graph is constructed in a way that each fact in the knowledge base induces an edge pointing from “entity” to “value”. Proximity query results are based on a fast simulation-based approximation of personalized PageRank [8]. Figure B.2(b) highlights a querying example.

The implementation of the interface is graphically illustrated in Figure B.3. It could be roughly divided into three parts:

- **Data Store**: The raw data set is stored as a single table in the MySQL database. A python script is written to generate derived data tables to provide more friendly interfaces to other parts of the system.
- **Query Processing Backend**: Upon initialization, it connects to the data server and build up in-memory data structures. It employs Thrift [105] to set up the interface between the C++ backend and the PHP web server. Caching of recent results was also implemented to reduce the response time and provide more consistent results despite the randomness of the approximation algorithm.
- **Web Client and Server**: The client-side collects user inputs and sends to the server-side using simple Ajax to smooth the user interaction process and results refreshing. The

server-side PHP communicates directly to the MySQL server to provide typeahead-like query suggestion. Suggestions for single-character inputs are hard-coded to eliminate the possible caveats in case of prolonged ajax response time.

RTW Knowledge Base Query Interface

This is a simple interface to query the knowledge base from [Read The Web \(RTW\) Project](#), which consists millions of (entity, relation, value) triplets, also known as facts, such as ("pittsburgh", "citylocatedinstate", "pennsylvania"). It supports the following two functions:

1. Fact Browsing: list all facts for a given query word and its role;
2. Similarity Query: based on a fast approximation of personalized pagerank.

Query word: Show query suggestion

Choose a task for the query word:

Query Results:

Figure B.1: A snapshot of the online query interface.

Query word: Show query suggestion
 Candidate Words: pittsburgh, pittsburgh_steelers, pittsburgh_pirates, pittsburgh_penguins, pittsburgh_post_gazette
 Choose a task for the query word: Fact Browsing
 Choose a role: Entity Relation Value

Query Results:

Time elapsed: 0.022 seconds.

Number of results: 278.

Entity	Relation	Value
pittsburgh	acquired	carrick
pittsburgh	actorstarredinmovie	city
pittsburgh	actorstarredinmovie	july
pittsburgh	agentcollaborateswithagent	roethlisberger

(a) Fact Browsing

Query word: Show query suggestion
 Candidate Words: perth, perthshire, perth_amboy, perth_mint, perthes___disease
 Choose a task for the query word: Similarity Query
 Max number of results: Show more parameters
 Number of Sampling Paths: Probability of Restart:

Query Results:

Time elapsed: 0.398 seconds.

Rank	Name	Score	Connection
1	western_australia	0.0285	perth<-(statehascapital)<-western_australia
2	city	0.0264	perth->(hasofficecity)->city
3	australia	0.0212	perth<-(statehascapital)<-australia
4	scotland	0.019	perth<-(statehascapital)<-scotland
5	skywest	0.0183	perth->(citylocatedincountry)->skywest
6	swan	0.0167	perth->(citylocatedinstate)->swan
7	tay	0.0152	perth->(cityisonriver)->tay

(b) Similarity Query

Figure B.2: Illustration of user actions supported by the query interface.

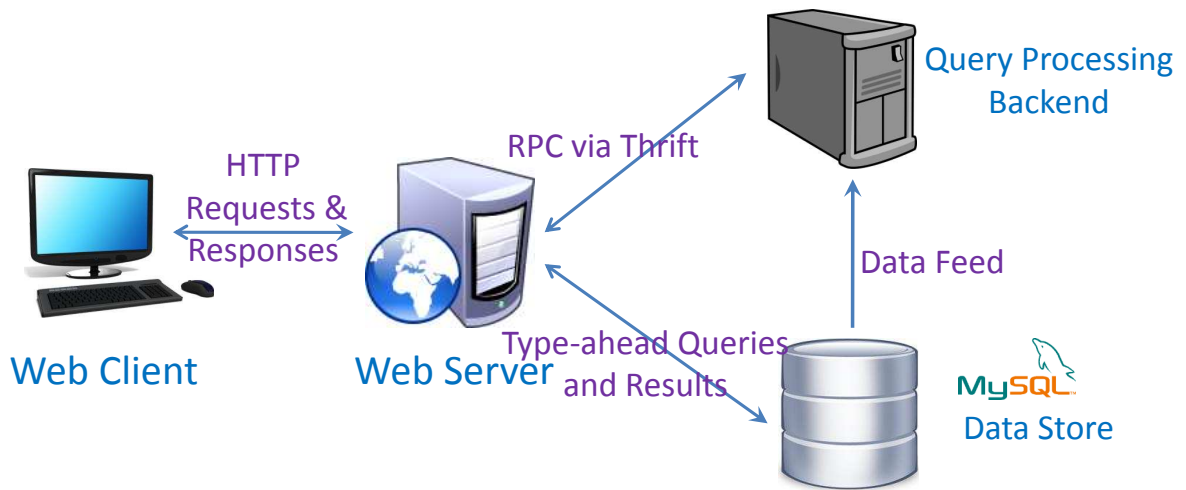


Figure B.3: A graphical illustration of the interface.

Bibliography

- [1] Evrim Acar, Daniel M. Dunlavy, Tamara G. Kolda, and Morten Mørup. Scalable tensor factorizations with missing data. In *Proceedings of the 2010 SIAM International Conference on Data Mining (SDM '10)*, pages 701–712, 2010.
- [2] Charu C. Aggarwal and Philip S. Yu. Outlier detection for high dimensional data. *SIGMOD Record*, 30:37–46, May 2001. ISSN 0163-5808.
- [3] Eugene Agichtein, Eric Brill, and Susan Dumais. Improving web search ranking by incorporating user behavior information. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '06*, pages 19–26, 2006.
- [4] Eugene Agichtein, Eric Brill, Susan Dumais, and Robert Ragno. Learning user interaction models for predicting web search result preferences. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 3–10, 2006.
- [5] Amr Ahmed and Eric P. Xing. On tight approximate inference of the logistic-normal topic admixture model. In *Proceedings of the 11th Tenth International Workshop on Artificial Intelligence and Statistics, AISTATS '07*, 2007.
- [6] Leman Akoglu, Mary McGlohon, and Christos Faloutsos. OddBall: Spotting anomalies in weighted graphs. In *Proceedings of the 14th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD '10)*, pages 410–421, 2010.
- [7] Claus A. Andersson and Rasmus Bro. The N-way toolbox for MATLAB. *Chemometrics and Intelligent Laboratory Systems*, 52(1):1–4, 2000.
- [8] Konstantin Avrachenkov, Nelly Litvak, Danil Nemirovsky, Elena Smirnova, and Marina Sokol. Quick detection of top-k personalized pagerank lists. In Alan Frieze, Paul Horn, and Pawel Pralat, editors, *Algorithms and Models for the Web Graph*, volume 6732 of *Lecture Notes in Computer Science*, pages 50–61. Springer Berlin / Heidelberg, 2011.
- [9] Brett W. Bader and Tamara G. Kolda. MATLAB Tensor Toolbox Version 2.4, March 2010. URL <http://csmr.ca.sandia.gov/~tgkolda/TensorToolbox/>.
- [10] Kobus Barnard, Pinar Duygulu, David Forsyth, Nando de Freitas, David M. Blei, and Michael I. Jordan. Matching words and pictures. *Journal of Machine Learning Research*, 3:1107–1135, March 2003.

- [11] V. Barnett and T. Lewis. *Outliers in Statistical Data*. John Wiley and Sons, 1994.
- [12] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. SURF: Speeded-up robust features. *Computer Vision and Image Understanding*, 110(3):346–359, 2008. ISSN 1077-3142.
- [13] BDGP. Berkeley Drosophila Genome Project. Patterns of gene expression in Drosophila embryogenesis. URL <http://insitu.fruitfly.org/cgi-bin/ex/insitu.pl>.
- [14] Arnab Bhattacharya, Vebjorn Ljosa, Jia-Yu Pan, Mark R. Verardo, Hyungjeong Yang, Christos Faloutsos, and Ambuj K. Singh. Vivo: Visual vocabulary construction for mining biomedical images. In *ICDM '05: Proceedings of the Fifth IEEE International Conference on Data Mining*, pages 50–57, 2005.
- [15] Mikhail Bilenko and Ryen W. White. Mining the search trails of surfing crowds: identifying relevant websites from user activity. In *WWW '08: Proceeding of the 17th international conference on World Wide Web*, pages 51–60, 2008.
- [16] David M. Blei and Michael I. Jordan. Modeling annotated data. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 127–134, 2003.
- [17] David M. Blei and John Lafferty. Correlated topic models. In *Advances in Neural Information Processing Systems 18*, pages 147–154, 2006.
- [18] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.
- [19] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. LOF: identifying density-based local outliers. *SIGMOD Record*, 29:93–104, May 2000. ISSN 0163-5808.
- [20] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka Jr., and Tom M. Mitchell. Toward an architecture for never-ending language learning. In *Proceedings of the Twenty-Fourth Conference on Artificial Intelligence (AAAI 2010)*, pages 1306–1313, 2010.
- [21] Andrew Carlson, Justin Betteridge, Richard C. Wang, Estevam R. Hruschka, Jr., and Tom M. Mitchell. Coupled semi-supervised learning for information extraction. In *WSDM '10: Proceedings of the third ACM international conference on Web search and data mining*, pages 101–110, 2010.
- [22] Miguel Á. Carreira-Perpiñ and Geoffrey Hinton. On contrastive divergence learning. In *Proceedings of the 10th Tenth International Workshop on Artificial Intelligence and Statistics, AISTATS '05*, pages 33–40, 2005.
- [23] Ben Carterette, Paul N. Bennett, David Maxwell Chickering, and Susan T. Dumais. Here or there: Preference judgments for relevance. In *ECIR '08: Proceedings of the 30th European Conference on Information Retrieval*, pages 16–27, 2008.

- [24] Deepayan Chakrabarti. AutoPart: parameter-free graph partitioning and outlier detection. In *Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD '04)*, pages 112–124, 2004.
- [25] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Computing Surveys*, 41:15:1–15:58, July 2009. ISSN 0360-0300.
- [26] Vineet Chaoji, Mohammad Al Hasan, Saeed Salem, and Mohammed J. Zaki. SPARCL: Efficient and effective shape-based clustering. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining (ICDM '08)*, pages 93–102, 2008.
- [27] Olivier Chapelle and Ya Zhang. A dynamic bayesian network click model for web search ranking. In *Proceedings of the 18th international conference on World wide web, WWW '09*, pages 1–10, 2009.
- [28] Andrew M. Childs, Ryan B. Patterson, and David J. C. MacKay. Exact sampling from non-attractive distributions using summary states. *Physical Review E*, 63(3):036113, 2001.
- [29] Diane J. Cook and Lawrence B. Holder. Substructure discovery using minimum description length and background knowledge. *Journal of Artificial Intelligence Research*, 1: 231–255, February 1994.
- [30] Robson L. F. Cordeiro, Fan Guo, Donna S. Haverkamp, James H. Horne, Ellen K. Hughes, Gunhee Kim, Agma J. M. Traina, Caetano Traina Jr., and Christos Faloutsos. Qmas: Querying, mining and summarization of multi-modal databases. In *Proceedings of the 10th IEEE International Conference on Data Mining, ICDM '10*, pages 785–790, 2010.
- [31] Robson L. F. Cordeiro, Fan Guo, Donna S. Haverkamp, James H. Horne, Ellen K. Hughes, Gunhee Kim, Agma J. M. Traina, Caetano Traina Jr., and Christos Faloutsos. Qmas: Querying, mining and summarization of multi-modal databases. Technical Report CMU-CS-10-144, Carnegie Mellon University, 2010.
- [32] Robson L. F. Cordeiro, Agma J. M. Traina, Christos Faloutsos, and Caetano Traina Jr. Finding clusters in subspaces of very large, multi-dimensional datasets. In *Proceedings of the 26th International Conference on Data Engineering, ICDE '10*, pages 625–636, 2010.
- [33] Nick Craswell, Onno Zoeter, Michael Taylor, and Bill Ramsey. An experimental comparison of click position-bias models. In *WSDM '08: Proceedings of the First ACM International Conference on Web Search and Data Mining*, pages 87–94, 2008.
- [34] Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of American Society of Information Science*, 41(6):391–407, 1990.
- [35] Luc Dehaspe and Hannu Toivonen. Discovery of frequent datalog patterns. *Data Mining and Knowledge Discovery*, 3:7–36, March 1999. ISSN 1384-5810.
- [36] Georges Dupret and Ciya Liao. A model to estimate intrinsic document relevance from the clickthrough logs of a web search engine. In *WSDM '10: Proceedings of the third ACM international conference on Web search and data mining*, pages 181–190, 2010.

- [37] Georges E. Dupret and Benjamin Piwowarski. A user browsing model to predict search engine click data from past observations. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 331–338, 2008.
- [38] William Eberle and Lawrence Holder. Discovering structural anomalies in graph-based data. In *Proceedings of the Seventh International Conference on Data Mining Workshops (ICDMW '07)*, pages 393–398, 2007.
- [39] Facebook. Ads: Click and impression quality, 2010. URL <http://www.facebook.com/help/?page=926>.
- [40] Steve Fox, Kuldeep Karnawat, Mark Mydland, Susan Dumais, and Thomas White. Evaluating implicit measures to improve web search. *ACM Trans. Inf. Syst.*, 23:147–168, April 2005. ISSN 1046-8188.
- [41] Thomas Franz, Antje Schultz, Sergej Sizov, and Steffen Staab. TripleRank: Ranking semantic web data by tensor decomposition. In *Proceedings of the 8th International Semantic Web Conference (ISWC '09)*, pages 213–228, 2009.
- [42] Erwin Frise, Ann S. Hammonds, and Susan E. Celniker. Systematic image-driven analysis of the spatial *Drosophila* embryonic expression landscape. *Mol Syst Biol*, 6:345, 2010.
- [43] Peter V. Gehler, Alex D. Holub, and Max Welling. The rate adapting poisson model for information retrieval and object recognition. In *ICML '06: Proceedings of the 23rd international conference on Machine learning*, pages 337–344, 2006.
- [44] Laurie Gibson and Dean Lucas. Spatial data processing using generalized balanced ternary. In *Proceedings of the IEEE Conference on Pattern Recognition and Image Processing*, pages 566–571, 1982.
- [45] Laurie Gibson, James Horne, and Donna Haverkamp. CASSIE: contextual analysis for spectral and spatial information extraction. In *Proceedings of the SPIE Conference on Signal Processing, Sensor Fusion, and Target Recognition*, 2009.
- [46] Leo Grady. Random walks for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:1768–1783, November 2006.
- [47] Fan Guo and Eric P. Xing. Bayesian exponential family harmoniums. Technical Report CMU-ML-06-103, Machine Learning Department, Carnegie Mellon University, May 2006.
- [48] Fan Guo, Lei Li, Christos Faloutsos, and Eric P. Xing. C-DEM: a multi-modal query system for *Drosophila* embryo databases. *Proc. VLDB Endow.*, 1(2):1508–1511, 2008.
- [49] Fan Guo, Lei Li, and Christos Faloutsos. Tailoring click models to user goals. In *WSCD '09: Proceedings of the 2009 workshop on Web Search Click Data*, pages 88–92, 2009.
- [50] Fan Guo, Chao Liu, Anitha Kannan, Tom Minka, Michael Taylor, Yi-Min Wang, and Christos Faloutsos. Click chain model in web search. In *WWW '09: Proceedings of the 18th international conference on World wide web*, pages 11–20, 2009.

- [51] Fan Guo, Chao Liu, and Yi-Min Wang. Efficient multiple-click models in web search. In *WSDM '09: Proceedings of the Second ACM International Conference on Web Search and Data Mining*, pages 124–131, 2009.
- [52] Zhen Guo, Zhongfei Zhang, Eric P. Xing, and Christos Faloutsos. Enhanced max margin learning on multimodal data mining in a multimedia database. In *KDD '07: Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 340–349, 2007.
- [53] Johan Håstad. Tensor rank is NP-complete. *Journal of Algorithms*, 11:644–654, December 1990. ISSN 0196-6774.
- [54] Douglas Hawkins. *Identification of outliers (Monographs on Statistics & Applied Probability)*. Chapman and Hall, 1980.
- [55] Geoffrey E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800, 2002.
- [56] Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '99, pages 50–57, 1999.
- [57] Glen Jeh and Jennifer Widom. Scaling personalized web search. In *WWW '03: Proceedings of the 12th international conference on World Wide Web*, pages 271–279, 2003.
- [58] Wen Jin, Anthony K. H. Tung, and Jiawei Han. Mining top-n local outliers in large databases. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '01)*, pages 293–298, 2001.
- [59] Thorsten Joachims. Transductive inference for text classification using support vector machines. In *ICML '99: Proceedings of the 16th international conference on Machine Learning*, pages 200–209, 1999.
- [60] Thorsten Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '02, pages 133–142, 2002.
- [61] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, and Geri Gay. Accurately interpreting clickthrough data as implicit feedback. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 154–161, 2005.
- [62] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, Filip Radlinski, and Geri Gay. Evaluating the accuracy of implicit feedback from clicks and query reformulations in web search. *ACM Trans. Inf. Syst.*, 25(2):7, 2007.
- [63] Edwin M. Knorr, Raymond T. Ng, and Vladimir Tucakov. Distance-based outliers: algorithms and applications. *The VLDB Journal*, 8(3-4):237–253, 2000. ISSN 1066-8888.
- [64] Tamara G. Kolda and Brett W. Bader. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500, September 2009.

- [65] Charlotte Konikoff, Michael McCutchan, Bernard Van Emden, Christopher Busick, Kailah Davis, Shuiwang Ji, Lin-Wei Wu, Hector Ramos, Thomas Brody, Sethuraman Panchanathan, Jieping Ye, Timothy Karr, Stuart J. Newfeld, and Sudhir Kumar. FlyExpress: A platform for discovering co-expressed genes via comparative image analysis of spatial patterns in *Drosophila* embryogenesis, 2009. (In preparation).
- [66] Flip Korn, Nikolaos Sidiropoulos, Christos Faloutsos, Eliot Siegel, and Zenon Protopapas. Fast nearest neighbor search in medical image databases. In *Proceedings of the 22nd International Conference on Very Large Data Bases, VLDB '96*, pages 215–226, 1996.
- [67] Michihiro Kuramochi and George Karypis. An efficient algorithm for discovering frequent subgraphs. *IEEE Transactions on Knowledge and Data Engineering*, 16:1038–1051, September 2004. ISSN 1041-4347.
- [68] Peter A. Lawrence. *The Making of a Fly: The Genetics of Animal Design*. Wiley-Blackwell, 1992.
- [69] Svetlana Lazebnik and Maxim Raginsky. An empirical Bayes approach to contextual region classification. In *Proceedings of the 22nd International Conference on Computer Vision and Pattern Recognition, CVPR '09*, pages 2380–2387, 2009.
- [70] LBNL. Lawrence Berkeley National Laboratory and ICSI. LBNL/ICSI enterprise tracing project. URL <http://www.icir.org/enterprise-tracing/>.
- [71] Uichin Lee, Zhenyu Liu, and Junghoo Cho. Automatic identification of user goals in web search. In *WWW '05: Proceedings of the 14th international conference on World Wide Web*, pages 391–400, 2005.
- [72] Chao Liu, Fan Guo, and Christos Faloutsos. BBM: Bayesian browsing model from petabyte-scale data. In *KDD '09: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 537–546, 2009.
- [73] Chao Liu, Mei Li, and Yi-Min Wang. Post-rank reordering: resolving preference misalignments between search engines and end users. In *CIKM '09: Proceeding of the 18th ACM conference on Information and knowledge management*, pages 641–650, 2009.
- [74] David G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the 7th IEEE International Conference on Computer Vision, ICCV '99*, pages 1150–1157, 1999.
- [75] Koji Maruhashi, Fan Guo, and Christos Faloutsos. MultiAspectForensics: Pattern mining on large-scale heterogeneous networks with tensor analysis. In *Proceedings of the Third International Conference on Advances in Social Network Analysis and Mining, ASONAM '11*, 2011.
- [76] Nicolaas Matthijs. AlterEgo: Personalized web search, 2010. URL <https://github.com/nicolaasmthijs/AlterEgo/>.
- [77] Nicolaas Matthijs and Filip Radlinski. Personalizing web search using long term browsing history. In *Proceedings of the fourth ACM international conference on Web search and*

- data mining*, WSDM '11, pages 25–34, 2011.
- [78] Andrew McCallum, Chris Pal, Gregory Druck, and Xuerui Wang. Multi-conditional learning: Generative/discriminative training for clustering and classification. In *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI'06)*, pages 433–439, 2006.
 - [79] Thomas P. Minka. Expectation propagation for approximate bayesian inference. In *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence*, UAI '01, pages 362–369, 2001.
 - [80] David M. Mount and Sunil Arya. Ann: A library for approximate nearest neighbor searching, 2010. URL <http://www.cs.umd.edu/~mount/ANN/>.
 - [81] Iain Murray and Zoubin Ghahramani. Bayesian learning in undirected graphical models: approximate mcmc algorithms. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, UAI '04, pages 392–399, 2004.
 - [82] Caleb C. Noble and Diane J. Cook. Graph-based anomaly detection. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '03)*, pages 631–636, 2003.
 - [83] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999.
 - [84] Jia-Yu Pan, Hyung-Jeong Yang, Christos Faloutsos, and Pinar Duygulu. Gcap: Graph-based automatic image captioning. In *CVPRW '04: Proceedings of the 2004 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'04) Volume 9*, page 146, 2004.
 - [85] Jia-Yu Pan, André G. R. Balan, Eric P. Xing, Agma Juci Machado Traina, and Christos Faloutsos. Automatic mining of fruit fly embryo images. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 693–698, 2006.
 - [86] Ruoming Pang, Mark Allman, Mike Bennett, Jason Lee, Vern Paxson, and Brian Tierney. A first look at modern enterprise traffic. In *Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement (IMC '05)*, pages 2–2, 2005.
 - [87] Ruoming Pang, Mark Allman, Vern Paxson, and Jason Lee. The devil and packet trace anonymization. *SIGCOMM Computer Communication Review*, 36:29–38, January 2006. ISSN 0146-4833.
 - [88] Pegasus. PEGASUS: Peta-Scale Graph Mining System. URL <http://http://www.cs.cmu.edu/~pegasus/>.
 - [89] Hanchuan Peng, Fuhui Long, and Chris Ding. Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1226–1238, 2005.
 - [90] Hanchuan Peng, Fuhui Long, Jie Zhou, Garmay Leung, Michael Eisen, and Eugene My-

- ers. Automatic image analysis for gene expression patterns of fly embryos. *BMC Cell Biology*, 8(Suppl 1):S7, 2007.
- [91] Yuan Qi, Martin Szummer, and Thomas P. Minka. Bayesian conditional random fields. In *Proceedings of the 10th Tenth International Workshop on Artificial Intelligence and Statistics*, AISTATS '05, pages 269–276, 2005.
- [92] Filip Radlinski and Thorsten Joachims. Query chains: learning to rank from implicit feedback. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, KDD '05, pages 239–248, 2005.
- [93] Filip Radlinski and Thorsten Joachims. Active exploration for learning rankings from clickthrough data. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '07, pages 570–579, 2007.
- [94] Matthew Richardson, Ewa Dominowska, and Robert Ragno. Predicting clicks: estimating the click-through rate for new ads. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 521–530, 2007.
- [95] RTW. Read the Web: Research Project at Carnegie Mellon University. URL <http://rtw.ml.cmu.edu/>.
- [96] J. C. Sexton and D. H. Weingarten. Hamiltonian evolution for the hybrid monte carlo algorithm. *Nuclear Physics B*, 380(3):665–677, 1992.
- [97] Amnon Shashua and Tamir Hazan. Non-negative tensor factorization with applications to statistics and computer vision. In *Proceedings of the 22nd international conference on Machine learning (ICML '05)*, pages 792–799, 2005.
- [98] Jamie Shotton, John Winn, Carsten Rother, and Antonio Criminisi. Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context. *Int. J. Comput. Vision*, 81:2–23, January 2009. ISSN 0920-5691.
- [99] Jeremy Siek, Lie-Quan Lee, and Andrew Lumsdaine. Boost Graph Library: a powerful C++ graph library. URL <http://www.boost.org/libs/graph/>.
- [100] Craig Silverstein, Hannes Marais, Monika Henzinger, and Michael Moricz. Analysis of a very large web search engine query log. *SIGIR Forum*, 33(1):6–12, 1999. ISSN 0163-5840.
- [101] Alan F. Smeaton and Paul Over. Trecvid: benchmarking the effectiveness of information retrieval tasks on digital video. In *Proceedings of the 2nd international conference on Image and video retrieval*, CIVR'03, pages 19–27, 2003.
- [102] Maria Patrizia Somma, Francesca Ceprani, Elisabetta Bucciarelli, Valeria Naim, Valeria De Arcangelis, Roberto Piergentili, Antonella Palena, Laura Ciapponi, Maria Grazia Giansanti, Claudia Pellacani, Romano Petrucci, Giovanni Cenci, Fiammetta Vern, Barbara Fasulo, Michael L. Goldberg, Ferdinando Di Cunto, and Maurizio Gatti. Identification of Drosophila mitotic genes by combining co-expression analysis and RNA interference. *PLoS Genet*, 4(7):e1000126, 2008.

- [103] Jimeng Sun, Huiming Qu, Deepayan Chakrabarti, and Christos Faloutsos. Neighborhood formation and anomaly detection in bipartite graphs. In *Proceedings of the Fifth IEEE International Conference on Data Mining (ICDM '05)*, pages 418–425, 2005.
- [104] Jimeng Sun, Dacheng Tao, Spiros Papadimitriou, Philip S. Yu, and Christos Faloutsos. Incremental tensor analysis: Theory and applications. *ACM Transactions on Knowledge Discovery from Data*, 2:11:1–11:37, October 2008.
- [105] Thrift. Apache Thrift. URL <http://thrift.apache.org/>.
- [106] Pavel Tomancak, Amy Beaton, Richard Weiszmann, Elaine Kwan, ShengQiang Shu, Suzanna Lewis, Stephen Richards, Michael Ashburner, Volker Hartenstein, Susan Celniker, and Gerald Rubin. Systematic determination of patterns of gene expression during *Drosophila* embryogenesis. *Genome Biology*, 3(12):research0088.1–0088.14, 2002.
- [107] Pavel Tomancak, Benjamin Berman, Amy Beaton, Richard Weiszmann, Elaine Kwan, Volker Hartenstein, Susan Celniker, and Gerald Rubin. Global analysis of patterns of gene expression during *Drosophila* embryogenesis. *Genome Biology*, 8(7):R145, 2007.
- [108] Hanghang Tong, Christos Faloutsos, and Jia-Yu Pan. Fast random walk with restart and its applications. In *ICDM '06: Proceedings of the Sixth International Conference on Data Mining*, pages 613–622, 2006.
- [109] Hanghang Tong, Spiros Papadimitriou, Jimeng Sun, Philip S. Yu, and Christos Faloutsos. Colibri: fast mining of large static and dynamic graphs. In *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '08)*, pages 686–694, 2008.
- [110] Antonio B. Torralba, Robert Fergus, and William T. Freeman. 80 million tiny images: A large data set for non-parametric object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30:1958–1970, November 2008.
- [111] Charalampos E. Tsourakakis. MACH: Fast randomized tensor decompositions. In *Proceedings of the 2010 SIAM International Conference on Data Mining (SDM '10)*, pages 689–700, 2010.
- [112] Susan Tweedie, Michael Ashburner, Kathleen Falls, Paul Leyland, Peter McQuilton, Steven Marygold, Gillian Millburn, David Osumi-Sutherland, Andrew Schroeder, Ruth Seal, Haiyan Zhang, and The FlyBase Consortium. FlyBase: enhancing *Drosophila* gene ontology annotations. *Nucleic Acids Research*, 37(suppl 1):D555–D559, 2009.
- [113] Max Welling, Michal Rosen-Zvi, and Geoffrey Hinton. Exponential family harmoniums with an application to information retrieval. In *Advances in Neural Information Processing Systems 17*, pages 1481–1488, 2005.
- [114] Eric P. Xing, Rong Yan, and Alexander G. Hauptmann. Mining associated text and images with dual-wing harmoniums. In *UAI '05: Proceedings of the 21st conference on Uncertainty in artificial intelligence*, pages 633–641, 2005.
- [115] Gui-Rong Xue, Hua-Jun Zeng, Zheng Chen, Yong Yu, Wei-Ying Ma, WenSi Xi, and

- WeiGuo Fan. Optimizing web search using web click-through data. In *CIKM '04: Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pages 118–126, 2004.
- [116] Xifeng Yan and Jiawei Han. gSpan: Graph-based substructure pattern mining. In *Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM '02)*, page 721, 2002.
- [117] Bin Zhang, Meichun Hsu, and Umeshwar Dayal. K-harmonic means - a spatial clustering algorithm with boosting. In *Proceedings of the First International Workshop on Temporal, Spatial, and Spatio-Temporal Data Mining-Revised Papers, TSDM '00*, pages 31–45, 2001.
- [118] Nan Zheng, Qiudan Li, Shengcai Liao, and Leiming Zhang. Flickr group recommendation based on tensor decomposition. In *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval (SIGIR '10)*, pages 737–738, 2010.
- [119] Jie Zhou and Hanchuan Peng. Automatic recognition and annotation of gene expression patterns of fly embryos. *Bioinformatics*, 23(5):589–596, 2007.
- [120] Xiaojin Zhu and Zoubin Ghahramani. Learning from labeled and unlabeled data with label propagation. Technical Report CMU-CALD-02-107, Center for Automated Learning and Discovery, Carnegie Mellon University, June 2002.
- [121] Zeyuan Allen Zhu, Weizhu Chen, Tom Minka, Chenguang Zhu, and Zheng Chen. A novel click model and its applications to online advertising. In *WSDM '10: Proceedings of the third ACM international conference on Web search and data mining*, pages 321–330, 2010.