Machine Learning for Information Retrieval

Yi Zhang
University of California Santa Cruz
What’s Information Retrieval

Data/Information

Access Information

Generate Knowledge

Filtering/Recommendation

Information Retrieval

Categorization

Clustering

Question and Answer

Mining: Sentiment Analysis, Information Extraction, etc.

Search
Related Areas

CS: Modeling
- Statistics, optimization
- Artificial Intelligence
- Machine learning & data mining
- Natural language processing

Information Domain
- Economics
- Marketing
- Phycology
- Healthy
- Legal
- Financial ...

Information Retrieval

Human computer interaction

Computer networks
- Distributed computing
- Storage
- Security

System
Outline

PART I

- Supervised learning and its IR applications
  - Text classification, adaptive filtering, collaborative filtering
  - Search engine: ranking
- Semi-supervised learning and its application to text classification

PART II

- Frontiers of information retrieval
- Example: Proactive information retrieval (recommendation systems)

First part prepared with Rong Jin
Supervised Learning: Basic Setting

- **Supervised learning**
  - Given: input and output variables pairs (training data) \( \{(x_1,y_1)(x_2,y_2)\ldots(x_N,y_N)\} \)
  - Learning: infer a function \( f(X) \) from the training data
  - Inference: predict future outcomes \( y=f(x) \) given \( x \)

classification: \( R^{[|V|]} \rightarrow \{0,1\} \)

regression: \( R^{[|V|]} \rightarrow R \)

[f]

\[ y \]

\[ f \]
How to Learn

- Given evidence (data) E, find hypothesis (model) h
- Maximum likelihood (ML) estimation
  \[ h_{ML} = \arg \max_h P(E \mid h) \]
- Maximum a posteriori (MAP) estimation
  \[ h_{MAP} = \arg \max_h P(h \mid E) \]
  \[ = \arg \max_h \frac{P(E \mid h)P(h)}{P(E)} \]
  \[ = \arg \max_h P(E \mid h)P(h) \]

Bayes’ rule

Posterior probability of hypothesis

Prior probability of hypothesis

Data likelihood
Text Categorization

- **Given:**
  - Predefined categories
  - Documents with class labels (training data)
  - A set of unlabelled new documents (testing data)

- **Task:** classify the new documents to existing classes

- **Applications:**
  - News article classification
  - Email classification, including spam filtering
  - Web page classification
  - Scientific journal articles
  - Word sense disambiguation
  - Language identification, author identification, tagging
  - Bug vs. non Bug
Major Steps for Text Classification

- Gathering a training set (input objects and corresponding outputs) from human or from other measurements
- Determine the input feature of the learned function (What’s X)
  - Typically, the input object is transformed into a feature vector
  - This step influence the final performance of the system greatly
- Determine the functional form of the learned algorithm
  - Logistic regression? Neural network? SVM? Gradient boosted trees?
- Determine the corresponding learning algorithm (ML or MAP or other loss function)
- Learn: run the learning algorithm on the gathered training set
  - Optional: adjust the parameter via cross validation
- Test the performance on a test set
Text Classification Approaches

- **Manual**
  1. Manually assign labels to each document
  2. Manual create rule based expert system

- **Automatic approaches**
  1. Naive Bayes and language models
  2. Decision tree
  3. Neural networks
  4. Support vector machines
  5. Boosting or bagging
  6. Regression (linear, polynomial …)
  7. K nearest neighbors
  8. Rocchio
Two Major ML Approaches

- **Generative models**
  - Model the joint probabilistic distribution $p(c, X)$, and derive the conditional probability
  \[
P(c_j | X) = \frac{P(X | c_j)p(c_j)}{\sum_j P(X | c_j)p(c_j)}
\]
  - Examples: Naive Bayes, language models, HMM

- **Discriminative models**
  - Model the conditional probability $P(c|X)$ directly
  - Examples: decision tree, neural networks, support vector machines, boosting or bagging, regression (linear, polynomial …), k nearest neighbor
Naïve Bayes (Multi-Variate Bernoulli Mixture Models)

- Pick up a class randomly according to a class prior $P(c_i)$
- The person runs through the dictionary, deciding whether to include each word $i$ in that document according to the probability $P(X_j=1|C_i)$
  \[ P(X_j=1|C_i)+P(X_j=0|C_i)=1 \]
- The probability of a document is
  \[ P(X,c_i) = P(c_i) \prod_{j=1}^{[V]} P(x_j | c_i) \]
  \[ X = \{x_1, x_2, ..., x_j, ..., x_{[V]} \}, \ x_j = \{0,1\} \]
Naïve Bayes (Multinomial Mixture Models)

- Pick up a class randomly according to a class prior $P(c_i)$
- Each document $X$ in a class $c$ is generated by a multinominal distribution

$$P(x | c) = P(x_1 | c)P(x_2 | c)...P(x_T | c)$$

where $\sum_{k=1}^{V} P(x_k | c) = 1$

$x = \{x_1, x_2, x_3, ..., x_j, ..., x_T\}, x_j \in \{1, 2, ..., |V|\}$

$T: \text{length of the text}$

$|V|: \text{dimensional document space}$
Learning in Naïve Bayes (Multinomial Model)

- $P(c) = $ the percentage of training documents that belong to class $c$
- Learning the probability of the $k$th word given the class $p(x_k|c)$
  - Maximum likelihood estimation
    $$P_{MLE}(x_i | c) = \frac{tf(x_i,c)}{\sum_{j=1...|V|}tf(x_j,c)}$$
  - Maximum a posteriori (MAP) estimation
    Prior : $(\mu P(x_1), \mu P(x_2),..., \mu P(x_{|V|}))$
    $$P_{MAP}(x_i | c) = \frac{tf(x_i,c) + \mu P(x_i)}{\sum_{j=1...|V|}tf(x_j,c) + \mu}$$

Avoid zero probability
Example of Documents Generation based on Multinomial Mixture Models

\[ p(x_k | c) \]

- search 0.02
- engine 0.01
- google 0.05
- yahoo 0.05
- orange 0.00001

class 1: Search engine

- food 0.025
- nutrition 0.01
- healthy 0.005
- diet 0.002

class 2: Weight control

Search engine paper

Weight control paper

(Ponte & Croft 98)
Example of ML Learning

\[ p(x_k|c) = ? \]

Google 100
search 50
engine 50
Yahoo 70
algorithm 20
query 10
compete 10

(total #words
occurrences=1000)

Documents related to “search engines”
Binary Classification in Naïve Bayes (Multinomial Model)

Classification Rule: classify a document $X$ to class $c$ if:

$$1 < \frac{P(c | X)}{P(c | X)} = \frac{P(X | c) P(c)}{P(X | c) P(c) / P(X)} = \frac{P(X | c) P(c)}{P(X | c) P(c)} = \prod_{j=1..|V|} P(x_j | c)^{t_{f_j} c} P(c)$$

$$\Leftrightarrow 0 < \sum_{j=1..|V|} t_{f_j} \log \frac{P(x_j | c)}{P(x_j | c) P(c)} + \log \frac{P(c)}{P(c)} = \left( \log \frac{P(c)}{P(c)}, \log \frac{P(x_1 | c)}{P(x_1 | c)}, ..., \log \frac{P(x_{|V|} | c)}{P(x_{|V|} | c)} \right) \times \begin{pmatrix} 1 \\ t_{f_1} \\ \vdots \\ t_{f_{|V|}} \end{pmatrix}$$

$$\Leftrightarrow W^T X > 0$$

linear separator
Naïve Bayes Classifier

- **Pros:**
  - Clear theoretical foundation
  - Relatively effective
  - Very simple
  - Fast: handles 10,000 attributes easily

- **Cons**
  - Wrong assumptions
    - Independence assumptions
    - Multi-Variate Bernoulli or Multinomial models assumption
    - One class per document assumption
  - Classification accuracy is usually worse than many other methods, such as logistic regression, linear regression or support vector machines
  - Bad probabilistic estimation of $P(c|x)$
Discriminative Models

- Focusing on estimating the decision boundary between classes or $P(y|X)$
- No explicit assumption on how the documents are generated
- For text classification task, usually the decision boundary is a linear separator
  - Assign a document to the positive class if $h(X) = W^T X > 0$
Linear Regression Model

- Modeling the conditional probability $p(y|X)$ as a Normal distribution

$$P(y | X, W) = N(y; W^T X, \sigma^2)$$
Logistic Regression

- Modeling the conditional probability $p(y|X)$ as a logistic function

$$P(y|X,W) = g(yW^T X) = \frac{1}{1 + e^{-yW^T X}}$$

- Document vector
- Model parameter vector
- Class label $\{-1,1\}$
Learning Logistic Regression Model

- Maximum likelihood estimation

\[ W_{ML} = \arg \max_{W} \prod_{i=1}^{t} p(y_i | x_i, W) = \arg \max_{W} \sum_{i=1}^{t} \log(p(y_i | x_i, W)) \]

- Maximum a posteriori (MAP) estimation

\[ W_{MAP} = \arg \max_{W} \prod_{i=1}^{t} p(y_i | x_i, W)P(W) \]

\[ = \arg \max_{W} \sum_{i=1}^{t} \log(p(y_i | x_i, W)) + \log(P(W)) \]

Usually a Gaussian prior

Document space (N)

likelihood of training data
If the goal is to minimize classification error, classify a document $X$ to the class if:

$$P(y = yes | X, W) = \frac{1}{1 + e^{-W^T X}} > 0.5 \iff W^T X > 0$$

linear separator
Why Use $P(W)$?

- Controlling model complexity
- Avoiding the pain of zero probability
- Integrating expert/prior knowledge
- Integrating two classification algorithms
- Transfer learning, multi task learning
  - Using one task to help another task
- Bayesian online learning
- Bayesian active learning
Learning as Empirical Loss Minimization

- Given: input and output variables pairs 
  \{ (x_1, y_1)(x_2, y_2) \ldots (x_N, y_N) \}
- Learning: infer W from the training data
- How: minimize the loss on training data

\[ \sum_i \text{Loss}(\hat{y}_i, y_i) = \sum_i \text{Loss}(y_i, f(x_i, w)) \]

- Maximize the posterior probability (Logistic regression, linear regression)
- Minimize the mean square error (Least Square Fit)
- Maximizing the margin between two classes (Support Vector Machines)

- Usually no close form solution. Gradient descent algorithms are often used to find the optimal solution W*
Loss Function for Different Discriminative Classifiers
Some Empirical Performance

From Li and Yang
SIGIR03
# Summary for Text Classification

<table>
<thead>
<tr>
<th>Focus</th>
<th>Generative Models (Naïve Bayes)</th>
<th>Discriminative Models (LR)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple</td>
<td>++</td>
<td>o</td>
</tr>
<tr>
<td>Training efficiency</td>
<td>++</td>
<td>+</td>
</tr>
<tr>
<td>Effectiveness</td>
<td>o</td>
<td>+</td>
</tr>
</tbody>
</table>
Non Parametric Learning

- Also called “instance based learning” or “memory based learning”
- Does not explicitly learn the function $f$
  - $f$ is encoded implicitly
- Examples: k nearest neighbor, kernel regression
- Four majors things
  1. A distance metric
  2. How many nearby neighbors to look at?
  3. A weighting function (optional)
  4. How to fit with the local points?
Instance Based Learning Summary

**Advantages:**
- Can fit low dimensional, very complex, functions very accurately
- Training, adding new data, is almost free.
- Variable resolution.
- Doesn’t forget old training data unless statistics warrant.
- Cross-validation is cheap

**Cons:**
- With large datasets, predictions are slow (although kdtree approximations, and newer cache approximations helps)
- Usually needs many training data to fit complex functions
Example: KNN for Collaborative Filtering

- Collaborative filtering task: will this user like the item?
- Assumption: users have similar tastes on some items may also have similar preferences on other items
- Making filtering decisions for one user based on the feedback from other users that are similar to this user
Basic Collaborative Filtering Setting

- Goal: Making filtering decisions for an individual user based on the judgments of other users
- General idea based on KNN
  - Given a user $u$, find similar users \{u_1, \ldots, u_m\}
  - Predict $u$'s rating based on the ratings of $u_1, \ldots, u_m$

<table>
<thead>
<tr>
<th></th>
<th>$o_1$</th>
<th>$o_2$</th>
<th>$\ldots$</th>
<th>$o_{j-1}$</th>
<th>$o_j$</th>
<th>$\ldots$</th>
<th>$o_K$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$u_1$</td>
<td>1</td>
<td>5</td>
<td>$\ldots$</td>
<td>4</td>
<td>2</td>
<td>$\ldots$</td>
<td>5</td>
</tr>
<tr>
<td>$u_2$</td>
<td>2</td>
<td>4</td>
<td>$\ldots$</td>
<td>4</td>
<td>1</td>
<td>$\ldots$</td>
<td>4</td>
</tr>
<tr>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
</tr>
<tr>
<td>$u_i$</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>$\ldots$</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
</tr>
<tr>
<td>$u$</td>
<td>2</td>
<td>5</td>
<td>5</td>
<td>$?$</td>
<td>$?$</td>
<td>$?$</td>
<td>$?$</td>
</tr>
</tbody>
</table>

Column: item    Row: user
Modified KNN for Collaborative Filtering

- Learn nearest neighbor interpolation weights that better accounts for interactions among neighbors

\[
\hat{r}_{ui} \leftarrow b_{ui} + \sum_{j \in N(i;u)} w_{ij} (r_{uj} - b_{uj})
\]

\[
\min_w \sum_{v \neq u} \left( r_{vi} - b_{vi} - \sum_{j \in N(i;u)} w_{ij} (r_{vj} - b_{vj}) \right)^2
\]
Search Engine & Ranking
Ranking

- Problem:
  - Given a well-formed query, place the most relevant pages in the first few positions

- Issues:
  - Scale: many candidate matches
    - Response in < 100 msecs
  - Evaluation: absolute metric or pair comparison
    - Editorial
    - User Behavior (# of clicks, # of abandoned queries, etc.)
History of Search

- 3rd Century B.C. Library of Alexandrian
  - Catalogs and classifications (controlled vocabulary)
  - Alphabetization
- 1247 First Concordance of the Bible
  - Invention of the inverted list data structure
- 1755 Johnson’s Dictionary
  - Standardize spellings, set standard for dictionary
- 1852 Roget’s Thesaurus
History of IR (continues)

- 1930’s Punch Card
  - Manual retrieval system
  - Satisfy Boolean query
  - Card: keyword
  - Document
  - Retrieval algorithm:
    The documents corresponding to the position where light falls through all “query” cards are the wanted documents.
History of IR (continues)

- 1950 First Citation of “Information Retrieval”
- 1960’s and 1970’s Computer based IR
  - Quantitative aspects of text and the models that were proposed were based on word frequencies and word occurrences
  - Small scale: library
- 1994 Web Search Engine
- 1997 Image and video retrieval
- 1999 Question and answering

Based on Bruce Croft and Ned Fielden
A Typical/Simple Retrieval Process

User’s Information Need (GUI, user models)

Comparison (retrieval models)

Representation (stop, stem, NLP, meta data, expansion, structure, phrase)

Query (query language)

Indexed Objects (inverted index)

Evaluation/ User Feedback

Retrieved Documents
Basic Issues in Retrieval

- How to represent text
- How to represent the information needs of the user
- How to compare representations (rank documents)
- How to evaluate the effectiveness of retrieval
Different Retrieval Models

- **Boolean model**: Rules for Identifying relevant documents
- **Vector space model**: a small relevant document
- **Bayesian inference model**: an expression of the information need
- **Language models**: sample of the relevant documents
- **Link based models (web)**
- **Learning to rank**
Learning to Rank: Problem Setting

- Given: user query \( q \), web pages \( D=\{d_1, d_2, \ldots, d_N\} \)
- Each web page \( d_i \) is represented as a feature vector
  - Text match score with title, anchor text, headings, bold text, body text, . . . , of \( d_i \) as a hypertext document
  - Pagerank, topic-specific Pageranks, personalized Pageranks of \( d_i \) as a node in the Web graph
  - Estimated location of user, commercial intent, . . .
- Goal: a single scoring function on \( (q, D) \) so as to induce a ranking on \( D=\{d_1, d_2, \ldots, d_N\} \)?
Ranking Features for (Document, Query) Pairs

<table>
<thead>
<tr>
<th>Query</th>
<th>Local</th>
<th>Global</th>
</tr>
</thead>
<tbody>
<tr>
<td>Independent</td>
<td>Document Length</td>
<td>Popularity/Pagerank</td>
</tr>
<tr>
<td></td>
<td>Page Quality</td>
<td>Web Connectivity</td>
</tr>
<tr>
<td></td>
<td>…</td>
<td>Document Clusters</td>
</tr>
<tr>
<td>Dependent</td>
<td>Term frequency</td>
<td>Anchor-text</td>
</tr>
<tr>
<td></td>
<td>Proximity</td>
<td>Number of results</td>
</tr>
<tr>
<td></td>
<td>Section matches</td>
<td>Result Set Clusters</td>
</tr>
<tr>
<td></td>
<td>…</td>
<td>…</td>
</tr>
</tbody>
</table>

From Ahmed
Learning to Rank Approaches

- **Pointwise learning**
  - **Input**: single document  
  - **Output**: relevance score  
  - As a traditional classification/regression problem  
  - Loss function defined over each document (Evaluation: difference between the score assigned by the algorithm and the “true score”)

- **Pairwise learning**
  - **Input**: document pairs \((d_i,d_j)\)  
  - **Output**: partial order preference  
  - As a special classification/regression problem to classify whether \(d_i\) is prefered to \(d_j\) or not.  
  - Loss function defined over each document pair (Evaluation: count the number of reversed pairs)

- **Listwise learning**
  - **Input**: document collections  
  - **Output**: ranked document list \(y\)  
  - Lost function defined over each ranking list
Collecting Explicit Document Relevance Judgments for A Query

LOOP

1. Query is issued to various search systems
2. Top N documents retrieved from different search systems are selected and merged into a pool for human assessment
3. Query is modified manually or automatically based relevant and irrelevant documents identified by human assessors

Until no new relevant documents is found

TREC Pooling Strategy: trec.nist.gov
Evaluation of Ranking Algorithms

- Rank correlation
- Precision@k
- Mean Reciprocal Rank
- Mean Average Precision
- NDCG, etc.

\[
\text{DCG}_p = \sum_{i=1}^{p} \frac{2^{\text{rel}_i} - 1}{\log_2 (1 + i)}
\]
Learning to Rank: Compare with Traditional Classification based Machine Learning

- Evaluation is usually rank list based instead of document based
  - NDCG, MAP, diversity of the list, etc.
  - Hard to optimize: non smooth and non differential able objective functions
  - Order: relative order of documents are important
  - Position: top ranked documents are more important

- Two major solutions
  - Introducing smooth and differentiable functions (AdaRank, SVM-MAP, SoftRank)
  - Using optimization techniques for non-smooth and non differentiable problems (Rank Genetic Programming, directly define gradient LambdaRank)
Online Learning with Implicit Feedback

Online learning: a generalized linear model that gives the utility of a ranking $y$ for a query $x$: $U(x,y)=W^T f_{x,y}$

LOOP FOREVER

Present a ranking $y$

Observe user feedback (clicks)

Training data: $(2,4,1,3,5) > (1,2,3,4,5)$

Update the model

Theoretical guarantee: average regret is bounded
Other Practical Concerns

- Controlling model complexity
  - Feature selection, smoothing (coefficient shrinkage): Ridge regression, lasso regression
- Over fitting (cross-validation, leave one out)
- Cost sensitive learning
  - Classify a ham as spam is more costly than the other way around
- Unbalanced samples and rare classes: 0.01% positive vs. 99.99% negative samples
- Biased samples
  - User only provides feedback on documents she reads
  - While she may not read randomly
- Noisy label
Semi-supervised learning and its application to text classification
Why Semi-supervised Learning

- Labeling could be expensive, and difficult
- Labeling could be unreliable
  - Ex. Segmentation applications
  - Need for multiple experts
- Unlabeled data
  - Easy to obtain in large numbers
  - Ex. Web pages, text documents, etc.
Semi-supervised Learning Problems

- Classification
  - Transductive – predict labels of unlabeled data
  - Inductive – learn a classification function
- Clustering (constrained clustering)
- Ranking (semi-supervised ranking)
- Almost every learning problem has a semi-supervised counterpart.
Why Unlabeled Data Might Help? 
Clustering Assumption
Points with same label are connected through high density regions, thereby defining a cluster

Clusters are separated through low-density regions
Statistical View

- Generative model for classification

\[ \Pr(X, Y | \theta, \eta) = \Pr(X | Y; \theta) \Pr(Y | \eta) \]

- Unlabeled data help estimate \( \Pr(X | Y; \theta) \)
  → Clustering assumption
Why Unlabeled Data Might Help?

Manifold Assumption

- Graph representation
  - Vertex: training example (labeled and unlabeled)
  - Edge: similar examples

- **Manifold assumption**
  - Data lies on a low-dimensional manifold
  - Classification function $f(x)$ should “follow” the data manifold
Statistical View

- Discriminative model for classification
  \[ \Pr(X, Y | \theta, \eta) = \Pr(X | \mu) \Pr(Y | X; \theta) \]

- Unlabeled data help regularize \( \theta \) via a prior \( \Pr(\theta | X) \)
  \( \rightarrow \) Manifold assumption
Semi-supervised Text Classification/Clustering

- Label propagation
- Graph partitioning based approaches
- Transductive Support Vector Machine (TSVM)
- Co-training
- Distance metric learning
Part II: Information Retrieval Frontiers
Conversational Answer Retrieval

- Current IR: open domain, keyword queries
- Current Question Answering system: limited domain, specific answers to natural language questions
- IR meet QA: rich user-system dialogue for understanding the question and refining the answer
- Conversation: natural language, pointing and licking
- Learning from heterogeneous data, interactive learning, active learning, mixed objectives/loss functions.
Empowering Users

- Current search engine users are becoming passive information receiver instead of active information seeker
- Future search engine let user interact with information so that user can do deeper learning
  - Empower users to be more proactive and critical thinkers while search
  - Suboptimal for many types of search tasks
- Learning task models
- Learning content models to represent different types of content and interaction model
- How to evaluate (utility/loss function)

WE LIVE IN THE ERA OF SMART MACHINE AND STUPID PEOPLE?
Zero or Less Query Terms

- Physical limitation of the mobile device limits a user’s ability/intention to search
  - Proactive information retrieval is needed
- Large amount of personal and contextual information available
  - Proactive information retrieval is likely to do well
An Example: Proactive Information Retrieval

Joint work with: Jian Wang, Lanbo Zhang, Jamie Callan, Wei Xu, Philip Zigoris, Yize Li, Jiazhong Nie, Jonathan Koren, Yun Chi, Wei Xu and more.
Flight From San Francisco to Home

Here are the papers you should read and cite in paper
If You Have Stock

Need to read news/blog/message related to your stock portfolio
Proactive IR Every Where

- You want to find a good restaurant in Santa Cruz
- A job for you after taking MLSS
- A app for you to install on your smart phone
- Shopping, movie, music, book …
Proactive IR in Mobile

- Physical limitation of the mobile device limits a user’s ability/intention to search
  - Proactive information retrieval is needed
- Large amount of personal and contextual information available
  - Proactive information retrieval is likely to do well
Beyond Siri: Personal Assistant

- Open domain
- Don’t suddenly switching modes when can not handle a hard environment

Augmented reality through electric contact lens or glasses

Whisper
Architecture: An Example
Major Challenges

- What to recommend
  - Will this user like the item (document, movie, products, restaurant, etc.)?
  - Use rich user interaction sequences, multi-model sensor data
  - Context: time and location
  - Consider the amount of time or information required
  - Open domain

- When to recommend: interruptibility; value of the information changes over time

- How to recommend: trust, educate the user
What to Recommend: Learn about the User

Forrest Gump:

"There is an awful lot you can tell about a person by their shoes ... where they're going, where they've been."
Find Your Soul Mate in Less than 10 Minutes

You can tell a lot about people by their reactions and opinions about other people.
-8 Minute Dating
How to Learn User Preferences?

- Approach 1: collaborative filtering
- Approach 2: content based adaptive filtering
- Hybrid filtering
Our Approach: Developing System with Desirable Characteristics

What can a person do? (desirable characteristics)
- Use heuristics
- Ask good questions
- Use context and implicit feedback
- Social learning

Our solution for a computer
- Bayesian Prior
- Bayesian Active Learning
- Graphical Models

Unified Framework
- Learning: Bayesian Graphical Models
- Bayesian Hierarchical Models, Probabilistic Relational learning
Challenge 1: Integrate Heterogeneous Information

How to use multiple evidence from the user, such as social networks, implicit user feedback, explicit user feedback, location, time, demographic information?
### Basic Collaborative Filtering Setting: User-Item Rating Matrix

- **User-Item Rating Matrix**

<table>
<thead>
<tr>
<th></th>
<th>o₁</th>
<th>o₂</th>
<th>…</th>
<th>oₘ₋₁</th>
<th>oₘ</th>
<th>…</th>
<th>oₖ</th>
</tr>
</thead>
<tbody>
<tr>
<td>u₁</td>
<td>1</td>
<td>5</td>
<td>…</td>
<td>4</td>
<td>2</td>
<td>..</td>
<td>5</td>
</tr>
<tr>
<td>u₂</td>
<td>2</td>
<td>4</td>
<td>…</td>
<td>4</td>
<td>1</td>
<td>…</td>
<td>4</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
</tr>
<tr>
<td>uᵢ</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>…</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td></td>
</tr>
<tr>
<td>u</td>
<td>2</td>
<td>5</td>
<td>5</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
</tbody>
</table>

**Column:** item  **Row:** user
Go Beyond User-Item Matrix: Two General Frameworks/Scenarios

- Scenario 1: A generative model for polyadic data
  Data: $\{(i, j, k, l, \ldots)\}$
  Example: (author, paper, keyword)

- Scenario 2: A discriminative model for multiple types of objects with multiple types of relationships
  Data: $\{(r, i, j, \ldots y)\}$
  Example: $(\text{rating}, \text{Tom}, \text{"The Matrix Reloaded"}, 5)$
  $(\text{trust}, \text{Tom}, \text{John}, 1)\ldots$
Scenario 1: A Generative Model for Polyadic Data

- **Probabilistic Polyadic Factor Model**
  - for generating an author-keyword-reference triple \((i,j,k)\)
    - A factor for author, a factor for keyword, and a factor for reference are selected with \(P(c_i^l, c_m^m, c_n^n)\)
    - Select author, keyword, and reference independently given the hidden factors

\[
P(c_i^l, c_m^m, c_n^n)\prod_{i,j,k \in l,m,n} P(c_i^l | c_j^l) P(j | c_m^m) P(k | c_n^n)
\]

- **Data likelihood for \(A=\{(a_{i,j,k})\}\)**
  \[
  \prod_{i,j,k \in l,m,n} (\sum_{c_i^l, c_m^m, c_n^n} P(c_i^l, c_m^m, c_n^n) P(c_i^l | c_j^l) P(j | c_m^m) P(k | c_n^n))^{a_{i,j,k}}
  \]
Learning with EM

\[
\arg \max_{\Theta} \sum_{i,j,k} a_{ijk} \log \left[ \sum_{l,m,n} P(c_{lmn})P(i|l)P(j|m)P(k|n) \right]
\]

where \( \Theta = \{ P(c_{lmn}), P(i|l), P(j|m), P(k|n) \} \)

\[ E - Step : \]
\[ \hat{p}_{lmn|ijk} = \frac{P(c_{lmn})P(i|l)P(j|m)P(k|n)}{\sum_{l',m',n'} P(c_{l'm'n'})P(i|l')P(j|m')P(k|n')} , \]
\[ M - Step : \]
\[ P(c_{lmn}) = \sum_{i,j,k} a_{ijk} \hat{p}_{lmn|ijk}, \]
\[ P(i|l) = \sum_{j,k,m,n} a_{ijk} \hat{p}_{lmn|ijk} / \sum_{j,k,l,m,n} a_{ijk} \hat{p}_{lmn|ijk}, \]
\[ P(j|m) = \sum_{i,k,l,n} a_{ijk} \hat{p}_{lmn|ijk} / \sum_{i,k,l,m,n} a_{ijk} \hat{p}_{lmn|ijk}, \]
\[ P(k|n) = \sum_{i,j,l,m} a_{ijk} \hat{p}_{lmn|ijk} / \sum_{i,j,l,m,n} a_{ijk} \hat{p}_{lmn|ijk}. \]

Extensions:
1. Dirichlet priors
2. Equivalence to non-negative tensor factorization with KL loss => other loss functions
## Experimental Results

**Del.io.us: Task:** For a user, given a tag, recommend top urls

<table>
<thead>
<tr>
<th>NDCG</th>
<th>Top 1</th>
<th>Top 3</th>
<th>Top 5</th>
<th>Top 10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline 1</td>
<td>0.091</td>
<td>0.149</td>
<td>0.175</td>
<td>0.208</td>
</tr>
<tr>
<td>Baseline 2</td>
<td>0.046</td>
<td>0.076</td>
<td>0.09</td>
<td>0.11</td>
</tr>
<tr>
<td><strong>Polyadic Factorization</strong></td>
<td><strong>0.102</strong></td>
<td><strong>0.171</strong></td>
<td><strong>0.204</strong></td>
<td><strong>0.251</strong></td>
</tr>
</tbody>
</table>

**Citeseer: Task:** For an author, given a keyword, recommend top references

<table>
<thead>
<tr>
<th>NDCG</th>
<th>Top 1</th>
<th>Top 3</th>
<th>Top 5</th>
<th>Top 10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline 1</td>
<td>0.025</td>
<td>0.045</td>
<td>0.057</td>
<td>0.076</td>
</tr>
<tr>
<td>Baseline 2</td>
<td>0.021</td>
<td>0.037</td>
<td>0.047</td>
<td>0.062</td>
</tr>
<tr>
<td><strong>Polyadic Factorization</strong></td>
<td><strong>0.038</strong></td>
<td><strong>0.070</strong></td>
<td><strong>0.090</strong></td>
<td><strong>0.120</strong></td>
</tr>
</tbody>
</table>

Yun Chi, Shenghuo Zhu, Yihong Gong, Yi Zhang
## Factors of Authors

<table>
<thead>
<tr>
<th>DB &amp; DM</th>
<th>Networks</th>
<th>AI &amp; ML</th>
</tr>
</thead>
<tbody>
<tr>
<td>T Eiter</td>
<td>J L Boudec</td>
<td>S Thrun</td>
</tr>
<tr>
<td>G Karypis</td>
<td>D Estrin</td>
<td>W Burgard</td>
</tr>
<tr>
<td>G D Giacomo</td>
<td>Z Zhang</td>
<td>H Zhang</td>
</tr>
<tr>
<td>W Faber</td>
<td>J A Stankovic</td>
<td>J Malik</td>
</tr>
<tr>
<td>N Leone</td>
<td>V Firoiu</td>
<td>M J Black</td>
</tr>
<tr>
<td>E Keogh</td>
<td>K G Shin</td>
<td>J A Fessler</td>
</tr>
<tr>
<td>J Yang</td>
<td>G B Giannakis</td>
<td>M J Mataric</td>
</tr>
<tr>
<td>J Han</td>
<td>J Liebeherr</td>
<td>C Sanderson</td>
</tr>
<tr>
<td>V J Tsotras</td>
<td>T Abdelzaher</td>
<td>R Deriche</td>
</tr>
<tr>
<td>D Calvanese</td>
<td>C Lu</td>
<td>D Fox</td>
</tr>
<tr>
<td>D Zhang</td>
<td>D Towsley</td>
<td>S Belongie</td>
</tr>
<tr>
<td>W Wang</td>
<td>I Stojmenovic</td>
<td>J Huang</td>
</tr>
<tr>
<td>M Hanus</td>
<td>T He</td>
<td>S Z Li</td>
</tr>
<tr>
<td>H Garcia-Molina</td>
<td>G Manimaran</td>
<td>I Cohen</td>
</tr>
<tr>
<td>S Mehrotra</td>
<td>E W Knightly</td>
<td>G Sapiro</td>
</tr>
<tr>
<td>D Gunopulos</td>
<td>M Reisslein</td>
<td>S Soatto</td>
</tr>
<tr>
<td>C Zaniolo</td>
<td>J Heidemann</td>
<td>A M Tekalp</td>
</tr>
</tbody>
</table>

Yun Chi, Shenghuo Zhu, Yihong Gong, Yi Zhang
## Factors of Keywords

<table>
<thead>
<tr>
<th>DB &amp; DM</th>
<th>Networks</th>
<th>AI &amp; ML</th>
</tr>
</thead>
<tbody>
<tr>
<td>database</td>
<td>network</td>
<td>recognition</td>
</tr>
<tr>
<td>documents</td>
<td>scheduling</td>
<td>segmentation</td>
</tr>
<tr>
<td>information</td>
<td>protocol</td>
<td>bayesian</td>
</tr>
<tr>
<td>document</td>
<td>differentiated</td>
<td>wavelet</td>
</tr>
<tr>
<td>language</td>
<td>mobility</td>
<td>feature</td>
</tr>
<tr>
<td>semantic</td>
<td>location</td>
<td>reconstruction</td>
</tr>
<tr>
<td>approach</td>
<td>packets</td>
<td>localization</td>
</tr>
<tr>
<td>semi-structured</td>
<td>capacity</td>
<td>approach</td>
</tr>
<tr>
<td>disjunctive</td>
<td>channels</td>
<td>statistical</td>
</tr>
<tr>
<td>declarative</td>
<td>communication</td>
<td>learning</td>
</tr>
<tr>
<td>extraction</td>
<td>schedulers</td>
<td>matching</td>
</tr>
<tr>
<td>programs</td>
<td>differentiation</td>
<td>correspondences</td>
</tr>
<tr>
<td>representation</td>
<td>multiplexing</td>
<td>estimate</td>
</tr>
<tr>
<td>dimensionality</td>
<td>queueing</td>
<td>registration</td>
</tr>
<tr>
<td>datasets</td>
<td>support</td>
<td>classification</td>
</tr>
</tbody>
</table>

Yun Chi, Shenghuo Zhu, Yihong Gong, Yi Zhang
## Factors of References

1. Fast algorithms for mining association rules
2. Mining association rules between sets of items in large databases
3. The anatomy of a large-scale hypertextual Web search engine
4. Authoritative sources in a hyperlinked environment
5. The stable model semantics for logic programming
6. Information retrieval

<table>
<thead>
<tr>
<th>1. Dynamic source routing in ad hoc wireless networks</th>
</tr>
</thead>
<tbody>
<tr>
<td>2. A performance comparison of multi-hop wireless ad hoc network routing protocols</td>
</tr>
<tr>
<td>3. Ad-hoc on-demand distance vector routing</td>
</tr>
<tr>
<td>4. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers</td>
</tr>
<tr>
<td>5. Next century challenges: scalable coordination in sensor networks</td>
</tr>
<tr>
<td>6. A highly adaptive distributed routing algorithm for mobile wireless networks</td>
</tr>
</tbody>
</table>

1. Reinforcement learning I: introduction
2. Bagging predictors
3. Support-vector networks
4. Optimization by simulated annealing
5. A decision-theoretic generalization of on-line learning and an application to boosting
6. Experiments with a new boosting algorithm
Scenario 2: How to Learn from Heterogeneous Information

Different types of objects &
Different types of relationships between objects
Scenario 2: A Discriminative Model for Multiple Types of Objects with Multiple Types of Relationships

Data: \{(relationship type, object 1, object 2, ..., value)\}
\{(r, i, j, ..., y)\}

Example: \{(rating, Tom, "Kill Bill", 5), (trust, Tom, John, 1)\}

\[ y_{r,i,j} \sim N(h_i^T A_r h_j + (W_r,1 f_i)^T W_r,2 f_j, 1/\lambda_r) \]
Prediction Based on Bayesian Inference

\[ D = \{(r, i, j, y)\}: \text{the training/labeled data set.} \]

\[ D' = \{(r, i, j)\}: \text{the testing data set.} \]

\[ \hat{y}_{r, i, j} = \int_{h_i, h_j} P(h_i)(h_i^T A_r h_j + (W_{r,1} f_i)W_{r,2} f_j) dh_i \]
Parameter Learning: EM Algorithm

\[ D = \{(r, i, j, y)\}: \text{the training/labeled data set.} \rightarrow \{h_i, A_r, W_{r,1}, W_{r,2}\} \]

The Joint likelihood of random variables in the model:

\[
P(H, A, W, D) = \prod_{(r, i, j, y) \in D} P(y_{r,i,j} | h_i, h_j, f_i, f_j, A_r, W_{r,1}, W_{r,w}, \lambda_r) \]
\[
\prod_i P(h_i | \lambda^{(h)}) \prod_r P(A_r | \lambda^{(A)}) P(W_{r,1} | \lambda^{(w)}) P(W_{r,2} | \lambda^{(w)})
\]

E step: the Variational Bayesian approach is used to approximate the posterior estimation of H, assuming (A;W) are known.

\[
P(H) \sim Q(H) = \prod_{i=1}^{N} Q(h_i)
\]

M step: we find the max a posterior estimation of A and W based on the approximate posterior estimation of Q(H) found at E step.
Experiments: Epinion.com

- Task: Integrating implicit, explicit user feedback, social network, and user meta data for product recommendation

More Information => Better Ranking Performance

R: Rating
RP: Rating + Purchasing
RPT: Rating + Purchasing + Trust Network
RPTF: Rating + Purchasing + Trust Network + Features

Zhang et.al. COLINGS 2010
## Similar Products

<table>
<thead>
<tr>
<th>Query</th>
<th>Neighbors</th>
<th>Name</th>
<th>ratings</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Little Smart Sort ‘n Go Car</td>
<td>92</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Little Smart Tiny Tot Driver</td>
<td>69</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Matchbox Special 5-Pack Vehicles</td>
<td>44</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Primo - Choo Choo Train</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Enfamil With Iron</td>
<td>202</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Activity Walker</td>
<td>42</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Similac With Irons</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Munchkin White Hot Basic Spoons</td>
<td>22</td>
<td></td>
</tr>
</tbody>
</table>

Zhang et.al. COLINGS 2010
Experiments: Yelp.com

Task: Context Based Restaurant Recommendation
Context features extracted from Yelp.com restaurant reviews using text mining techniques (time, location, occasion, companion)

Zhang et.al. COLINGS 2010
More about Probabilistic Latent Relational Models

- Two human brain areas divide up language by semantics and syntax.
  - Left posterior area: the meaning of words
  - Left frontal cortex: grammatical process
- PLRM: a simple Bayesian graphical model that approximates how the brain processes
  - Learning the hidden representation of objects ($h_i$)
  - Learning the relationship between objects ($A_r, W_r$)
Problem: Recommendation Based on Predicted Rating is Not Good for Consumers

- Will you like it?
  - Traditional recommender systems recommend item with the highest predicted rating

- Will you purchase it?
Basic Economics: Utility, Total Utility, Marginal Utility

Consumer purchases product to get satisfaction (utility)
Recommendation based on Marginal Net Utility

- Design: Design the utility functional form
- Learn: Learn the utility function parameters from user history
- Predict: Predict the marginal net utility of a product for a user using the function learned and user history
- Rank: Rank products based on predicted marginal net utility
Problem Setting

$u = 1, \ldots, M$

$i$ or $j = 1, \ldots, N$

$C_i$

day 1  day 3  day 7  day 10

Time $t$
Proposed Utility Functional Form

\[ \Delta U(X, i) = \alpha_i \left( \log(x_i + 1) - \log(x_i) \right) \]

New:

\[ \Delta U_{u,t}(X, i) = \alpha_{u,t} \left( \frac{Y_i}{x_{u,t,i} + 1} \right) \]

- **U(X)**: utility for the existing purchase history X
- **\Delta U(X, i)**: marginal utility for the addition purchase of product \( p_i \)
- **\alpha_i**: product’s basic utility
- **x_i**: purchase quantity of product \( p_i \) in the history
Revamping Factorization Models with Utility

Marginal net utility

\[ v_{u,i,t} = f(u,i) \left[ (x_{u,i,t}+1)^{\gamma_i} - (x_{u,i,t})^{\gamma_i} \right] - c_i \]

Take SVD as an example

\[ P_{u,i} = q_i^T p_u \]

\[ v_{u,i,t} = q_i^T p_u \left[ (x_{u,i,t}+1)^{\gamma_i} - (x_{u,i,t})^{\gamma_i} \right] - c_i \]
Results on E-Commerce: Shop.com

conversion rate@K

![Bar chart showing conversion rate@K with bars for different K values and conversion rates for Top Popular, SVD_{matrix}, SVD_{util} with θ = 1, and SVD_{util} with θ = 0.7.]

Zhang et.al. SIGIR 2011
What the Model can Learn?

- Examples of Repurchases
  - diaper, pet food, etc.
  - Tend to be consumable products

- Examples of New purchases
  - computer, cell phone, bed frame, etc.
  - Tend to be durable product
  - law of diminishing marginal utility
Extensions: When to Recommend

- **Different diminishing of return rate for a product**
  - For different users
  - For different user segments

- **Beyond diminishing of return**
  - A customer might buy a new camera 2 years later
  - A customer either will buy a new camera immediately (if it fails) or several years later (for upgrade)
  - After buying a camera, buy lens 3 weeks later, and buy battery 1 year later

- A user interest change over time
A Problem: User is Misled

Potential buyer: I want a silver Toyota Camry

Sales: come here, we have Toyota Camry

ANGRY
This is a Common Problem in Information Retrieval Systems

- IR system: present summarization of each document by selecting words that match the "query" or "user profile"
- User: click irrelevant documents (misled by the summary)
  - Waste of time
- Our solution: provide information (summary, explanation, faceted navigation) based on user utility optimization

How to generate summary?
- Learn from human: select relevant meta data (facet value pairs)
- Gradient boosted trees
Example: Summarizing Movie in Search Results

The utility function

<table>
<thead>
<tr>
<th></th>
<th>Document: +</th>
<th>Document: -</th>
</tr>
</thead>
<tbody>
<tr>
<td>User: clicked (guess +)</td>
<td>A</td>
<td>−B</td>
</tr>
<tr>
<td>User: didn’t click (guess -)</td>
<td>−C</td>
<td>D</td>
</tr>
</tbody>
</table>

Evaluation: An Mechanical Turk-based game

Each turks/subject needs to make a decision based on the summary

Each turk/subject receives credits or penalty based on whether the decision is good or bad. and is paid accordingly.

Table 4: Performances of Different Summarization Approaches. * and ◊ denote a significant improvement over MFS and MMR respectively (p-value < 0.05).

<table>
<thead>
<tr>
<th>Approach</th>
<th>MANU</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>MFS</td>
<td>0.813</td>
<td>0.871</td>
<td>0.708</td>
</tr>
<tr>
<td>MMR</td>
<td>0.887</td>
<td>0.867</td>
<td>0.859*</td>
</tr>
<tr>
<td>QSFS</td>
<td>0.923◊</td>
<td>0.921◊</td>
<td>0.887*</td>
</tr>
<tr>
<td>MMR-QSFS</td>
<td>0.929*◊</td>
<td>0.933*◊</td>
<td>0.886*</td>
</tr>
</tbody>
</table>

Our approaches

Zhang et.al. SIGIR 2012
How to Recommend 2: Gain User Trust and Confidence

Diagram:
- User Perceived Quality:
  - Explanation
  - Recommendation Accuracy
  - Recommendation Novelty
  - Recommendation Diversity

- User Beliefs:
  - Transparency

- User Attitudes:
  - Trust & Confidence
  - Overall Satisfaction
  - Perceived Usefulness

- Behavioral Quality:
  - Use Intention
  - Context Compatibility
Example: FMVilla, Social Music Recommendation with Explanation
Summary: Proactive IR without Query

- What to recommend: predict which product a user likes
  - Integrate homogeneous information: social, contextual, meta data, user actions, user ratings, demographic, etc.
  - Explore user interest while making recommendation
  - Integrate heuristics into a learning model
- When to recommend
  - Economics and marketing: better model consumer purchase decision process => lead to better recommendation results
- How to recommend
  - Provide information and interaction mechanism to help a user make decision
  - Transparency: gain user trust, help users understand how the system works and help them become better users
How to Scale
Applying Machine Learning to IR

- Set up a simple baseline (such as the average, random)
- Set up a well known baseline (such as tf*idf)
- Debug the learning algorithm (system) with good understanding of why and when it works and does not work
  - Getting more training data?
  - Less/more features?
  - Change the features?
  - It converges?
  - A bug?
QA

Yi Zhang
University of California, Santa Cruz
yiz@soe.ucsc.edu
http://users.soe.ucsc.edu/~yiz