Machine Learning Approaches to Query Document Matching in Search

Hang Li
Microsoft Research Asia
Talk Outline

• Semantic Matching in Search
• Learning to Match
  – Similarity Learning
  – Topic Modeling
  – String Matching
• Our Methods
  – Robust Similarity Function Learning Using Kernel Methods
  – Regularized Latent Semantic Indexing
  – Query Generation Using Log Linear Model
  – Query Rewriting Using Conditional Random Fields
Semantic Matching in Search
A Good Web Search Engine

• Must be good at
  – Relevance
  – Freshness
  – Comprehensiveness
  – User interface

• Relevance is particularly important
Query Document Mismatch is Biggest Challenge for Relevance
Same Search Intent Different Query Representations
Example = “Distance between Sun and Earth”

- "how far" earth sun
- "how far" sun
- "how far" sun earth
- average distance earth sun
- average distance from earth to sun
- average distance from the earth to the sun
- distance between earth & sun
- distance between earth and sun
- distance between earth and the sun
- distance from earth to the sun
- distance from sun to earth
- distance from sun to the earth
- distance from the earth to the sun
- distance from the sun to earth
- distance of earth from sun
- distance between earth sun
- how far away is the sun from earth
- how far away is the sun from the earth
- how far earth from sun
- how far earth is from the sun
- how far from earth is the sun
- how far from earth to sun
- how far from the earth to the sun
- distance between sun and earth
Same Search Intent, Different Query Representations

Example = “Youtube”

<table>
<thead>
<tr>
<th>Query</th>
<th>Query</th>
<th>Query</th>
</tr>
</thead>
<tbody>
<tr>
<td>yutube</td>
<td>yuotube</td>
<td>yuo tube</td>
</tr>
<tr>
<td>ytube</td>
<td>youtubr</td>
<td>yu tube</td>
</tr>
<tr>
<td>youtubo</td>
<td>youtuber</td>
<td>youtubecom</td>
</tr>
<tr>
<td>youtube om</td>
<td>youtube music videos</td>
<td>youtube videos</td>
</tr>
<tr>
<td>youtube</td>
<td>youtube com</td>
<td>youtube co</td>
</tr>
<tr>
<td>youtub com</td>
<td>you tube music videos</td>
<td>youtube</td>
</tr>
<tr>
<td>youtub</td>
<td>you tube com yourtube</td>
<td>your tube</td>
</tr>
<tr>
<td>you tube</td>
<td>you tub</td>
<td>you tube video clips</td>
</tr>
<tr>
<td>you tube videos</td>
<td>www you tube com</td>
<td>wwww youtube com</td>
</tr>
<tr>
<td>www youtube</td>
<td>www youtube com</td>
<td>www youtube co</td>
</tr>
<tr>
<td>yotube</td>
<td>www you tube</td>
<td>www utube com</td>
</tr>
<tr>
<td>ww youtube com</td>
<td>www utube</td>
<td>www u tube</td>
</tr>
<tr>
<td>utube videos</td>
<td>utube com</td>
<td>utube</td>
</tr>
<tr>
<td>u tube com</td>
<td>utub</td>
<td>u tube videos</td>
</tr>
<tr>
<td>u tube</td>
<td>my tube</td>
<td>toutube</td>
</tr>
<tr>
<td>outube</td>
<td>our tube</td>
<td>toutube</td>
</tr>
</tbody>
</table>
Examples of Term Mismatch

• Query → Document
  • swimming pool schedule → pool schedule
  • seattle best hotel → seattle best hotels
  • natural logarithm transformation → logarithm transformation
  • china kong → china hong kong
  • why are windows so expensive → why are macs so expensive
Different Levels of Semantic Matching

- **Term**: Match exactly same terms.  
  - NY ➔ New York  
  - disk ➔ disc

- **Word Sense**: Match terms with same meanings.  
  - utube ➔ youtube  
  - NY ➔ New York  
  - motherboard ➔ mainboard

- **Topic**: Match topics of query and document.  
  - Microsoft Office ➔ ... working for Microsoft ... my office is in ...  
  - Topic: PC Software  
  - Topic: Personal Homepage

- **Structure**: Match structures of query and document.  
  - Microsoft Office home ➔ find homepage of Microsoft Office  
  - 21 movie ➔ find movie named 21  
  - buy laptop less than 1000 ➔ find online dealers to buy laptop with less than 1000 dollars
Michael Jordan is Professor in the Department of Electrical Engineering
Online Semantic Matching

Matching can be conducted at different levels

1. [michael I. jordan: Keyphrase]
2. [berkeley: Attribute]: academic
3. [michael jordan: Keyphrase]
4. [berkeley: Attribute]: academic

Ranking Results

- [Michael Jordan/M. Jordan: Keyphrase] is [Professor] in the [Department/Dept.] of [Electrical Engineering/EE]: academic
Related Work

• Studied in long history of IR
• Query expansion, pseudo relevance feedback
• … …
Learning to Match
Matching Problem

<table>
<thead>
<tr>
<th></th>
<th>d1</th>
<th>d2</th>
<th>d3</th>
<th>dn</th>
</tr>
</thead>
<tbody>
<tr>
<td>q1</td>
<td></td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>q1</td>
<td></td>
<td></td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>q1</td>
<td></td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>q1</td>
<td>1</td>
<td></td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>qm</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Using both features and additional knowledge
Matching between Heterogeneous Data is Everywhere

• Matching between user and product (collaborative filtering)
• Matching between text and image (image annotation)
• Matching between people (dating)
• Matching between languages (machine translation)
Challenges in Matching

• No enough information (mismatch)
• Scale is very large
# Matching vs Ranking

<table>
<thead>
<tr>
<th></th>
<th>Matching</th>
<th>Ranking</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Prediction</strong></td>
<td>Matching score between query and document</td>
<td>List of documents</td>
</tr>
<tr>
<td><strong>Model</strong></td>
<td>$f(q, d)$</td>
<td>$f(q,d1), f(q,d2), \ldots f(q,dn)$</td>
</tr>
<tr>
<td><strong>Loss Function</strong></td>
<td>Single query document pair</td>
<td>List of documents with respect to query</td>
</tr>
<tr>
<td><strong>Challenge</strong></td>
<td>Mismatch</td>
<td>Correct ranking on top</td>
</tr>
</tbody>
</table>
Three Approaches of Learning to Match

• Similarity Learning  →  Word sense level
• Topic Modeling  →  Topic level
• String Matching  →  Token level, structure level
Robust Similarity Function Learning Using Kernel Methods

Wei Wu, Hang Li, Jun Xu, Satoshi Oyama, JMLR 2011
Dealing with Mismatch with Re-Ranking
- Our Approach = Online Learning of Kernel Methods

Michael Jordan

Michael I. Jordan
Michael Jordan NBA
Michael Jordan Berkeley

Input query

Similar queries

Retrieved documents

Re-ranking

Retrieved documents
Introduction to Kernel Methods

- **Kernel** $k: \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$
  
  - Definition: $k(x, x') = \langle \phi(x), \phi(x') \rangle$, 
    where $\phi: \mathcal{X} \rightarrow \mathcal{H}$
  
  - Given $k_1$ and $k_2$ are kernels, create new kernels: $\alpha k$, where $\alpha \geq 0$; $k_1 + k_2$; $k_1 \cdot k_2$

\[
k(x, x') = \langle \phi(x), \phi(x') \rangle
\]
Positive Semi-definite Kernel

• Positive Semi-definite Kernel:
  – Kernel matrix is symmetric and semi definite

\[
\begin{bmatrix}
K(x_1, x_1), \cdots, K(x_1, x_n) \\
\vdots \\
K(x_n, x_1), \cdots, K(x_n, x_n)
\end{bmatrix}
\]

\(\forall n, \quad \begin{bmatrix}
K(x_1, x_1), \cdots, K(x_1, x_n) \\
\vdots \\
K(x_n, x_1), \cdots, K(x_n, x_n)
\end{bmatrix}\]

• Reproducing Kernel Hilbert Space:

Hilbert Space \(H\) generated by \(K(x, x')\)
Kernel Methods (e.g., SVM)

• Positive Semi-definite Kernel:
  \[ k(x, y) = \langle \varphi(x), \varphi(y) \rangle_{\mathcal{H}}, \] generating RKHS \( \mathcal{H} \)

• Input
  - Training data \( S = \{x_i, t_i\}_{1 \leq i \leq N} \)

• Output
  - Prediction function \( f(x) \)

• Optimization
  - \( \min_{f \in \mathcal{H}} \frac{1}{N} \sum_{i=1}^{N} l(f(x_i), t_i) + \Omega(f) \)

• Representer theorem
  - \( f^*(x) = \sum_{i=1}^{N} \alpha_i k(x, x_i) \)
IR Models as Similarity Functions (Similarity Functions) (Xu and Li 2010)

Query Space

Document Space

New Space

VSM, BM25, LM, MRF
IR Models Are Similarity Functions

- **VSM**
  
  
  \[ BM25(q, d) = \langle \phi^V_{q}(q), \phi^V_{d}(d) \rangle, \text{ for all } w \in V \]
  
  \[ \phi^V_{q}(q)_w = \text{tfidf}(w, q) \text{ and } \phi^V_{d}(d)_w = \text{tfidf}(w, d) \]

- **BM25**
  
  \[ BM25(q, d) = \langle \phi^B_{q}(q), \phi^B_{d}(d) \rangle, \text{ for all } w \in V \]
  
  \[ \phi^B_{q}(q)_w = \frac{(k_3+1) \times \text{tf}(w, q)}{k_3 + \text{tf}(w, q)} \]
  
  \[ \phi^B_{d}(d)_w = \text{IDF}(w) \cdot \frac{(k_1+1) \times \text{tf}(w, d)}{k_1(1-b+b \cdot \frac{\text{len}(d)}{\text{avgDocLen}})+\text{tf}(w, d)} \]

- **LMIR**
  
  \[ LMIR(q, d) = \langle \phi^L_{q}(q), \phi^L_{d}(d) \rangle + \text{len}(q) \cdot \log \frac{\mu}{\text{len}(d)+\mu'}, \text{ for all } w \in V \]
  
  \[ \phi^L_{q}(q)_w = \text{tf}(w, q) \]
  
  \[ \phi^L_{d}(d)_w = \log \left(1 + \frac{\text{tf}(w, d)}{\mu \cdot P(w)}\right), \text{ where } P(w) \text{ plays similar role as IDF in BM25} \]
Relevance beyond Unigram

unigram

bigram

2-dependent-terms

co-occur (machine, learning)

co-occur (learning, book)
Extension of IR models

- **BM25**
  - \( BM25(q, d) = \langle \phi_Q^{BM25}(q), \phi_D^{BM25}(d) \rangle \), and for all \( w \in V \)
    \[
    \phi_Q^{BM25}(q)_w = \frac{(k_3+1) \times tf(w, q)}{k_3 + tf(w, q)}
    \]
    \[
    \phi_D^{BM25}(d)_w = IDF(t) \cdot \frac{(k_1+1) \times tf(w, d)}{k_1 \left(1 - b + b \cdot \frac{\text{len}(d)}{\text{avgDocLen}}\right) + tf(w, d)}
    \]

- **BM25\_Kernel**
  - \( BM25\_Kernel(q, d) = \sum_t BM25\_Kernel_t(q, d) \) where \( t \) is dependence type
  - \( BM25\_Kernel_t(q, d) = \langle \phi_Q^{BM25}(q), \phi_D^{BM25}(d) \rangle \), and for all \( x \in V_t \)
    \[
    \phi_{Q,t}^{BM25}(q)_x = \frac{(k_3+1) \times f_t(x, q)}{k_3 + f_t(x, q)}
    \]
    \[
    \phi_{D,t}^{BM25}(d)_x = IDF_t(x) \cdot \frac{(k_1+1) \times f_t(x, d)}{k_1 \left(1 - b + b \cdot \frac{f_t(d)}{\text{avgDocLen}_t}\right) + f_t(x, d)}
    \]
Similarity Function

- Kernel $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$
  - Definition: $k(x, x') = \langle \phi(x), \phi(x') \rangle$, where $\phi : \mathcal{X} \rightarrow \mathcal{H}$
  - Given $k_1$ and $k_2$ are kernels, create new kernels: $\alpha k$, where $\alpha \geq 0$; $k_1 + k_2$; $k_1 \cdot k_2$

- Similarity function: $k : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$
  - Definition: $k(x, y) = \langle \phi(x), \phi'(y) \rangle$, where $\phi : \mathcal{X} \rightarrow \mathcal{H}$ and $\phi' : \mathcal{Y} \rightarrow \mathcal{H}$
  - Given $k_1$ and $k_2$ are similarity functions, create new similarity functions: $\alpha k$, where $\alpha \in \mathbb{R}$; $k_1 + k_2$; $k_1 \cdot k_2$
Kernel vs Similarity Function

\[ k(x, x') = \langle \phi(x), \phi(x') \rangle \]

\[ k(x, y) = \langle \phi(x), \phi'(y) \rangle \]
Mapping to Space of Query Document Pairs
- Using Kernel Methods

Query-document pair space

Hilbert space $\tilde{k}((q, d), (q', d'))$

Query space

Hilbert space $k_Q(q, q')$

Document space

Hilbert space $k_D(d, d')$

Matching

$\tilde{k}(q, d)$

$\tilde{k}(q', d')$

$\tilde{k}(d, d')$

Similarity Functions
Similarity Learning

- Similarity Function: \( k(x, y) = \langle \varphi(x), \varphi(y) \rangle_H \)
- Input
  - Training data \( S = \{(x_i, y_i), t_i\} \) \( 1 \leq i \leq N \)
- Output
  - Similarity Function
- Optimization
  \[
  \min_{k \in \mathcal{K} \subseteq \mathcal{A}} \frac{1}{N} \sum_{i=1}^{N} l(k(x_i, y_i), t_i) + \Omega(k)
  \]
Similarity Learning Using Kernel Methods

- **Assumption**
  - Space of similarity functions is RKHS generated by positive-definite kernel \( k: (\mathcal{X} \times \mathcal{Y}) \times (\mathcal{X} \times \mathcal{Y}) \).

- **Optimization**
  \[
  \min_{k \in \mathcal{K}} \frac{1}{N} \sum_{i=1}^{N} l(k(x_i, y_i), t_i) + \frac{\lambda}{2} \| k \|_{\mathcal{K}}^2
  \]

- **Solution**
  - By representer theorem \( k^*(x, y) = \sum_{i=1}^{N} \alpha_i \bar{k}((x_i, y_i), (x, y)) \)
  - \( \bar{k}((x, y), (x', y')) = g(x, y) k_{\mathcal{X}}(x, x') k_{\mathcal{Y}}(y, y') g(x', y') \)
Learning Robust BM25

- BM25 =
  - Kernel
    $$\bar{k}((q, d), (q', d')) = k_{BM25}(q, d)k_Q(q, q')k_D(d, d')k_{BM25}(q', d')$$
- Solution (called Robust BM25)
  $$k_{RBM25}(q, d) = k_{BM25}(q, d) \cdot \sum_{i=1}^{N} \alpha_i k_Q(q, q_i)k_D(d, d_i)k_{BM25}(q_i, d_i)$$
- Deal with term mismatch
Regularized Latent Semantic Indexing

Quan Wang, Jun Xu, Hang Li, Nick Craswell, SIGIR 2011
Topic Modeling
Our Approach = Regularized Latent Semantic Indexing

\[ d_1, d_2, \ldots, d_N \]

Learning System

Model

\[ d_m \]

Prediction System

\[ t_1, t_2, \ldots, t_k \]
Introduction to Topic Modeling

<table>
<thead>
<tr>
<th></th>
<th>estimation</th>
<th>prediction</th>
<th>translation</th>
<th>parsing</th>
</tr>
</thead>
<tbody>
<tr>
<td>document 1</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>document 2</td>
<td>5</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>document 3</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>document 4</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>
### Probabilistic Approach

<table>
<thead>
<tr>
<th></th>
<th>d1</th>
<th>dn</th>
</tr>
</thead>
<tbody>
<tr>
<td>w1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>wm</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

**Probabilistic Latent Semantic Indexing**

**Latent Dirichlet Allocation**
Non-Probabilistic Approach

Linear Projection to Lower Dimensional Space

Latent Semantic Indexing
Regularized Latent Semantic Indexing

\[
\min_{U, \{v_n\}} \sum_{n=1}^{N} \|d_n - Uv_n\|_2^2 + \lambda_1 \sum_{k=1}^{K} \|u_k\|_1 + \lambda_2 \sum_{n=1}^{N} \|v_n\|_2^2
\]

Topics are sparse
Regularized Latent Semantic Indexing

- L1 on topics and L2 on documents
- L1 leads to sparse topics, a topic only contains a small number of words
- L2 leads to accurate modeling
- Formulation is simple
- Easy to scale up
## Scalability Comparison

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Max Dataset Applied (#docs; #words)</th>
<th># Topics</th>
<th># Processors Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>PLDA and PLDA+ (by Google)</td>
<td>Wiki-200T (2,112,618; 200,000)</td>
<td>1000</td>
<td>2,048</td>
</tr>
<tr>
<td>AD-LDA (by UCI)</td>
<td>NY Times (300,000; 102,660)</td>
<td>200</td>
<td>16</td>
</tr>
<tr>
<td>RLSI</td>
<td>B01 (1,562,807; 7,014,881) Wikipedia (3,239,884; 1,689,193) Bing News (1,028,070; 940,702)</td>
<td>500 ~ 1000</td>
<td>16 single machine!</td>
</tr>
</tbody>
</table>
## Regularized Topics

AP dataset, topic compactness: 0.0075

<table>
<thead>
<tr>
<th>OPEC</th>
<th>Africa</th>
<th>contra</th>
<th>school</th>
<th>Noriega</th>
<th>firefight</th>
<th>plane</th>
<th>Saturday</th>
<th>Iran</th>
<th>senate</th>
</tr>
</thead>
<tbody>
<tr>
<td>oil</td>
<td>South</td>
<td>Sandinista</td>
<td>student</td>
<td>Panama</td>
<td>ACR</td>
<td>crash</td>
<td>coastal</td>
<td>Iranian</td>
<td>Reagan</td>
</tr>
<tr>
<td>cent</td>
<td>African</td>
<td>rebel</td>
<td>teacher</td>
<td>Panamanian</td>
<td>forest</td>
<td>flight</td>
<td>estimate</td>
<td>Iraq</td>
<td>billion</td>
</tr>
<tr>
<td>barrel</td>
<td>Angola</td>
<td>Nicaragua</td>
<td>education</td>
<td>Delval</td>
<td>park</td>
<td>air</td>
<td>western</td>
<td>hostage</td>
<td>budget</td>
</tr>
<tr>
<td>price</td>
<td>apartheid</td>
<td>Nicaraguan</td>
<td>college</td>
<td>canal</td>
<td>blaze</td>
<td>airline</td>
<td>Minsch</td>
<td>Iraqi</td>
<td>trade</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>drug</th>
<th>soviet</th>
<th>aid</th>
<th>court</th>
<th>Jackson</th>
<th>percent</th>
<th>student</th>
<th>nuclear</th>
<th>Bush</th>
<th>Israel</th>
</tr>
</thead>
<tbody>
<tr>
<td>cocaine</td>
<td>Afghanistan</td>
<td>virus</td>
<td>senate</td>
<td>Dukaki</td>
<td>billion</td>
<td>Korea</td>
<td>soviet</td>
<td>Dukaki</td>
<td>Palestinian</td>
</tr>
<tr>
<td>traffick</td>
<td>Afghan</td>
<td>infect</td>
<td>Reagan</td>
<td>democrat</td>
<td>rate</td>
<td>protest</td>
<td>treaty</td>
<td>campaign</td>
<td>Israeli</td>
</tr>
<tr>
<td>test</td>
<td>Gorbachev</td>
<td>test</td>
<td>house</td>
<td>delegate</td>
<td>0</td>
<td>Korean</td>
<td>missile</td>
<td>Quayle</td>
<td>Arab</td>
</tr>
<tr>
<td>enforce</td>
<td>Pakistan</td>
<td>patient</td>
<td>state</td>
<td>percent</td>
<td>trade</td>
<td>Chun</td>
<td>weapon</td>
<td>Bentsen</td>
<td>PLO</td>
</tr>
</tbody>
</table>
Optimization

Algorithm 2 Update $U$

Require: $D \in \mathbb{R}^{M \times N}$, $V \in \mathbb{R}^{K \times N}$
1: $S \leftarrow VV^T$
2: $R \leftarrow DV^T$
3: for $m = 1 : M$ do
4: $\bar{u}_m \leftarrow 0$
5: repeat
6: for $k = 1 : K$ do
7: $w_{mk} \leftarrow r_{mk} - \sum_{l \neq k} S_{kl} u_{ml}$
8: $u_{mk} \leftarrow \frac{(|w_{mk}| - \frac{1}{2} \alpha N)_+ \text{sign}(w_{mk})}{s_{kk}}$
9: end for
10: until convergence
11: end for
12: return $U$

Algorithm 3 Update $V$

Require: $D \in \mathbb{R}^{M \times N}$, $U \in \mathbb{R}^{M \times K}$
1: $\Sigma \leftarrow (U^TU + \theta I)^{-1}$
2: $\Phi \leftarrow U^TD$
3: for $n = 1 : N$ do
4: $v_n \leftarrow \Sigma \phi_n$, where $\phi_n$ is the $n^{th}$ column
5: end for
6: return $V$
Scaling up on MapReduce
Query Generation Using Log Linear Model

Ziqi Wang, Gu Xu, Hang Li, Ming Zhang

ACL 2011
Candidate Generation in Spelling Correction

Our Approach = Log Linear Model

\[(W_m^1, W_c^1), (W_m^2, W_c^2), \ldots, (W_m^N, W_c^N)\]

Learning System

Model

Dictionary

Retrieval System

\[W_m, W_{c,1}, W_{c,2}, \ldots, W_{c,k}\]
Introduction to Log Linear Model

• Exponential Model (Log Linear Model)

\[ P(y \mid x) = \frac{\exp\left(\sum_{i} w_{i}f_{i}(x, y)\right)}{Z(x)} \]

\[ Z(x) = \sum_{y} \exp\left(\sum_{i} w_{i}f_{i}(x, y)\right) \]

• Logistic Regression
• Maximum Entropy
Logistic Regression

\[ P(y \mid x) = \frac{\exp(w \cdot x + b)}{1 + \exp(w \cdot x + b)} \]

\[ = \frac{1}{1 + \exp(-(w \cdot x + b))} \]

\[ \ln \frac{P(y \mid x)}{1 - P(y \mid x)} = w \cdot x + b \]
Maximum Likelihood Estimation

• Maximizing Log Likelihood

\[
\max_\lambda L(\lambda) = \sum_{x,y} \tilde{P}(x, y) \log P(y \mid x)
\]

\[
P(y \mid x) = \frac{\exp(\sum_i w_i f_i(x, y))}{Z(x)}
\]

\[
Z(x) = \sum_y \exp(\sum_i w_i f_i(x, y))
\]

• Improved iterative scaling, Quasi Newton, gradient descent
Candidate Generation in Spelling Error Correction

officier

model
dictionary

officer
office
official
offices
Learning

Training Data

\((w^1_m, w^1_c)\)

\((w^2_m, w^2_c)\)

\((w^3_m, w^3_c)\)

\(\ldots\)

Rule Extraction

\(\alpha_1 \rightarrow \beta_1\)

\(\alpha_2 \rightarrow \beta_2\)

\(\alpha_3 \rightarrow \beta_3\)

\(\ldots\)

Model Learning

\(P(w_c, R(w_m, w_c) \mid w_m)\)

\(\alpha_1 \rightarrow \beta_1, \lambda_1\)

\(\alpha_2 \rightarrow \beta_2, \lambda_2\)

\(\alpha_3 \rightarrow \beta_3, \lambda_3\)

\(\ldots\)

Model

Log linear model

Weight
Rule Extraction

• Edit-distance based alignment:

Misspelled: \[ ^n \ i \ c \ o \ s \ o \ f \ t \ $ \]

Correct: \[ ^m \ i \ c \ r \ o \ s \ o \ f \ t \ $ \]

• Basic substitution rules:

\[ n \rightarrow m, \phi \rightarrow r \]

• Contextual substitution rules

\[ ^n \rightarrow ^m, ni \rightarrow mi, ^ni \rightarrow ^mi, c \rightarrow cr, \ldots \]
Log Linear Model

- **Model**
  \[
P(w_c, R(w_m, w_c) | w_m) = \frac{\exp(\sum_{r \in R(w_m, w_c)} \lambda_r)}{\sum_{(w'_c, R(w_m, w'_c)) \in Z(w_m)} \exp(\sum_{o \in R(w_m, w'_c)} \lambda_o)}
\]

  - Set of rules rewrite \( w_m \) to \( w_c \)
  - Weight of rule \( \lambda_r \) for each rule \( r \)
  - All pairs of word \( w'_c \) and rule set \( R(w_m, w'_c) \)

- **Candidate Generation**
  \[
  rank(w_c | w_m) = \max_{R(w_m, w_c)} \left( \sum_{r \in R(w_m, w_c)} \lambda_r \right)
  \]

  - Non-positive constraint, to improve efficiency in retrieval,
  - Natural assumption

\[\forall \lambda_r \leq 0\]
Model Learning

• Objective function

\[ \lambda^* = \arg \max_{\lambda} \sum_i \max_{R(w_m^i, w_c^i)} \log P(w_c^i, R(w_m^i, w_c^i) \mid w_m^i) \]

Take max over transformations

• Algorithm
  – Constrained Quasi Newton Method (BFGS)
Retrieval

Misspelled word $W_m$ → Finding all matching rules $\alpha_1 \rightarrow \beta_1, \lambda_1$
$\alpha_2 \rightarrow \beta_2, \lambda_2$
$\alpha_3 \rightarrow \beta_3, \lambda_3$
$\ldots$

Aho Corasick Tree (rule set)

Find best candidates

Trie Tree (dictionary)

Top k candidates

$W_{c,1}$
$W_{c,2}$
$W_{c,3}$
$\ldots$
Aho Corasick Tree

Index all the α ‘s in the rules on the AC tree

β are stored in an associated list
Retrieval with Dynamic Programming

• Traverse trie tree
  – Match the next position of $w_m$
  – Apply a rule at the current position of $w_m$
• Two pruning strategies
  – If the sum of weights is smaller than the smallest weight in the top $k$ list, prune the branch
  – two search branches merge, prune the smaller branch
Query Rewriting Using Conditional Random Fields

Jiafeng Guo, Gu Xu, Hang Li, Xueqi Cheng
SIGIR 2008
Candidate Selection in Spelling Error Correction
Our Approach = Conditional Random Fields

\[(q_m^1, q_c^1), (q_m^2, q_c^2), \ldots, (q_m^N, q_c^N)\]

- Learning System
- Model
- Prediction System

\[q_m, q_{c,1}, q_{c,2}, \ldots, q_{c,k}\]
Introduction to Conditional Random Fields

- $x: \{1,2,\ldots,N\}$ are values of states
- $y: \{1,2,\ldots,M\}$ are values of observations

$P(Y \mid X)$
Model of Conditional Random Fields

\[
P(Y \mid X) = \frac{\exp\left(\sum_{k} \lambda_k \cdot F_k(Y, X)\right)}{Z(X)}
\]

\[
Z(X) = \sum_{Y} \exp\left(\sum_{k} \lambda_k \cdot F_k(Y, X)\right)
\]

\[
F(Y, X) = \sum_{t=1}^{T} f(y_t, x_t)
\]
Tagging Using CRF

- Viterbi algorithm

\[
\text{arg max}_Y P(Y \mid X) = \text{arg max}_Y \log \frac{\exp \sum_k \lambda_k \cdot F_k (Y, X)}{Z(X)}
\]

\[
= \text{arg max}_Y \sum_k \lambda_k \cdot F_k (Y, X)
\]
Learning of CRF

\[(X_1, Y_1), (X_2, Y_2), \cdots (X_n, Y_n)\]

\[
P(Y \mid X) = \frac{\exp\left(\sum_k \lambda_k \cdot F_k(Y, X)\right)}{Z(X)}
\]

\[
Z(X) = \sum_Y \exp\left(\sum_k \lambda_k \cdot F_k(Y, X)\right)
\]

\[
\arg \max \sum_{i=1}^{n} \log \frac{\exp\left(\sum_k \lambda_k \cdot F_k(Y, X)\right)}{Z(X)}
\]

Maximum Likelihood
IIS, Quasi Newton
Candidate Selection in Spelling Error Correction

officier website -> model

officer website
office website
official website
Candidate Selection Problem

Observed "noisy" word sequence

"Ideal" word sequence

$y^* = \arg \max_y \Pr(y|x)$

"ideal" query word sequence  original query word sequence
Conditional Random Fields for Candidate Selection

Introducing Refinement Operations

Operations

Spelling: insertion, deletion, substitution, transposition, …

Word Stemming: +s/-s, +es/-es, +ed/-ed, +ing/-ing, …
Query Refinement Using Conditional Random Fields

\[
\Pr(y, \bar{o}, \bar{z}|x) = \frac{1}{Z(x)} \prod_{i=1}^{n} \phi(y_{i-1}, y_i) \prod_{j=1}^{m_i} \phi(z_{ij_i}, o_{ij_i}, z_{ij_i-1})
\]
Summary
## IR Matching (Relevance) Models

<table>
<thead>
<tr>
<th>Basic Model (unigram)</th>
<th>Probabilistic Approach</th>
<th>Non Probabilistic Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BM25[Robertson], LM4IR [Zhai ][Ponte &amp; Croft]</td>
<td>Vector Space Model [Salton]</td>
</tr>
<tr>
<td>Extended Model (n-gram)</td>
<td>MRF[Metzler &amp; Croft]</td>
<td>Similarity Function [Xu &amp; Li]</td>
</tr>
<tr>
<td>Robust Model</td>
<td></td>
<td>Rocchio [Rocchio], Kernel Method [Wu et al]</td>
</tr>
</tbody>
</table>
## IR Matching (Relevance) Models

<table>
<thead>
<tr>
<th>Model Type</th>
<th>Probabilistic Approach</th>
<th>Non Probabilistic Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>Topic Model</td>
<td>PLSI [Hoffman], LDA [Blei et al]</td>
<td>LSI [et al], RLSI [Xu et al]</td>
</tr>
<tr>
<td>String Matching Model (token level)</td>
<td>Generative model [Brill &amp; Moore], Log linear model [Wang et al], CRF [Guo et al]</td>
<td></td>
</tr>
<tr>
<td>String Matching Model (structure level)</td>
<td>Translation Model [Berger &amp; Lafferty]</td>
<td></td>
</tr>
</tbody>
</table>
Summary

• Semantic Matching in Search
• Learning to Match
  – Similarity Learning
  – Topic Modeling
  – String Matching
• Our Methods
  – Robust Similarity Function Learning Using Kernel Methods
  – Regularized Latent Semantic Indexing
  – Query Generation Using Log Linear Model
  – Query Rewriting Using Conditional Random Fields
Thank You!

hangli@microsoft.com